LEARNING-BASED MODEL REDUCTION AND CONTROL OF UNDERACTUATED BALANCE ROBOTS

by

KUO CHEN

A dissertation submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Doctor of Philosophy

Graduate Program in Mechanical and Aerospace Engineering

Written under the direction of

Jingang Yi

And approved by

____

____

____

____

New Brunswick, New Jersey

JANUARY, 2019

ABSTRACT OF THE DISSERTATION

Learning-based Model Reduction and Control of Underactuated Balance Robots

By KUO CHEN

Dissertation Director:

Jingang Yi

Underactuated robots are mechanical systems with fewer control inputs than their degrees of freedom (DOF). Inverted pendulums, bicycles and walking robots are a few examples of such systems. Underactuated balance robots are underactuated robots that must perform the balancing and tracking tasks simultaneously. The balancing task requires the robot to maintain its balance around possibly unstable equilibrium points, while the tracking task requires it to track desired trajectories. For these competing tasks, a common guideline to design controllers is to identify a low dimensional subspace of the state space, called a latent manifold, on which the balancing and tracking tasks are consistent and compatible. The approach of latent manifold identification is called model reduction. Previous works apply model reduction to physical-principled models with well understood dynamics structures. This dissertation proposes machine

learning-based model reduction approaches for modeling high dimensional robots, extracting balancing skills from demonstration data and controlling robots with data-driven models.

Several aspects of machine learning make it attractive for use in model reduction and control applications for underactuated robots. First, the system dynamics learned from collected data can be more accurate than analytical models derived from physical laws. For high dimensional motion, it is also easier to build learning-based latent space models than any physical models. Second, a latent manifold that encodes balancing skill can be learned from the demonstrated trajectories. One application example is to transfer human walking skills to humanoid robots by enforcing humanoid robots onto the latent manifold identified from human trajectories. Finally, optimization-based controllers such as model predictive control (MPC) and reinforcement learning can be integrated with the learned model and applied to stabilize the learned open-loop dynamics onto the desired latent manifolds.

In this dissertation, we introduce a framework that integrates the physical-based robot model with the learning-based latent manifold model for high dimensional human limbs motion to achieve pose estimation of human robot interactions. Human-bikebot interaction is used as an example to demonstrate the proposed approach. We extend the physical latent manifold-based controller to achieve biped slip recovery. We reveal the relationship between learning-based model reduction and physical-based model reduction for high dimensional dynamics such as human legged locomotion. One of our final goals is to design learning-based controllers to achieve biped walking and slip recovery. The balancing while tracking problem has been successfully solved by designing physical model-based controllers to stabilize the system state onto the balance equilibrium manifold (BEM). However, its application has been restricted to systems with well understood dynamics structures. In the last part of this dissertation, we adopt the BEM concept to design a learning model-based control framework. The

system dynamics is identified without prior physical knowledge nor successful balancing demonstrations. The proposed framework achieves superior control performance compared to the physical model-based approach, and provides analytical performance guarantees. The works in this dissertation are demonstrated using multiple robotic platforms such as an inverted pendulum, a bikebot and a biped robot.

Acknowledgements

I am grateful to my Ph.D. adviser Jingang Yi for his support and guidence. Besides the knowledge I learned from him, I have always been encouraged by his enthusiasm about our work and enlightened by his vision on research topics. Without him, my work can never go as far as this.

I would like to thank my committee members: Professor Qingze Zou, Professor Haym Benaroya and Professor Ahmed Elgammal for their valuable feedback to my work.

During these years of study, I work extensively with my fellow lab mates. I am grateful to Yizhai Zhang. In the beginning of my PhD, he gave me a lot help and advisement both on life and research. I have been lucky to work with Mitja Trkov. We collaborate in the bipedal walking project and face the challenges from our adviser together. I admire his communication skill and his effort to balance family and work. I am thankful to Kaiyan Yu. We started our PhD in the same year and took courses together. I would like to thank Pengcheng Wang. A lot of my knowledge of underactuated system control is learned from him. Through our disccusion, I setup the framework of my learning based controller. I would like to thank Yoshitaka Abe. Together, we designed controller for the biped robot from Japan. I am grateful to Fei Liu for his tremendous help in my job hunting. I am grateful to Merrill Edmonds for proofreading my dissertation and giving invaluable feedback. I would like to thank Jiangbo Liu, Siyu Chen, Chaoke Guo, Yongbin Gong and Marko Mihalec for their discussion and help on multiple research works.

There are a lot of friends at Rutgers who are not directly related to my research but their accompany made my journey much more enjoyable. I am grateful to Wei Song. Wei is full of interesting ideas and always shares them with me. I would like to thank Longle Ma and Libing Yu for many bicycle riding and hiking events. I am grateful to Hao Peng for many discussions on research, job hunting and future planning. I am thankful to Han Sun, Qiming Guan and Xiaosheng Zhong for many casual talks and dinner eating. I would like to thank all friends with whom we play pingpong every Friday night, play tennis in the summer, skate in the winter, play board games, run marathon, go hiking and have meals together. They give me supports and encouragements more than they thought.

Last, I am grateful to my parents. They always give me unconditional love and support and have more confidence on me than myself.

Dedication

To my parents

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

1.1 Background and motivations

Underactuated robots are systems with fewer control inputs than their configuration states. Underactuated balance robots are underactuated robots that perform the balancing and tracking tasks simultaneously. Because of the limited actuation and multiple tasks, it is challenging to design feedback controllers for underactuated balance robots. For example, autonomous bicycle driving is an underactuated balance control problem which requires the bicycle to both track desired output trajectory and maintain internal states, i.e. bicycle roll angle, bounded around the unstable equilibrium points. Bipedal walker is another example of high-dimensional underactuated balance robot. The control inputs not only actuate the robot joints to form desired configuration trajectories such as bending the knee and swinging the foot but also enforce the floating base or equivalently the center of mass (COM) onto a periodical orbit to achieve periodical walking.

Various approaches have been proposed to control such underactuated mechanical systems. Energy-based method was applied to the swing-up control of inverted pendulum [3] [4]. This approach has been extended to balance a stationary bicycle in [5]. In [6], a reduction based controller has been proposed for general underactuated systems. Through a closed form global coordinate transformation, the original underactuated system model is transformed into a normal form. Underactuated systems with symmetry properties can be reduced to the normal forms that are structured cascade nonlinear systems. Controller for the reduced normal form is designed in [6]. Both energy-based and reduction-based control design require well structured analytical models of the robotic systems. Domain knowledge of the model structure and physical insights of the control task are critical to the successful design of these controllers.

Bipedal walking systems, as a special example of underactuated systems, have drawn particular attention from the research community because of their great potential applications in humanoid robots and human assistive devices. The difficulty of bipedal walking control lies in the balancing while tracking control tasks under the limited actuations. During walking, joint angles are controlled to follow desired trajectories so that the swinging leg moves forward to serve as the stance base for the next step. In the meantime, the floating base or equivalently the trunk should be propelled by the resultant ground reaction force (GRF) properly so that the robot does not fall forward or backward. The joint torque actuations are responsible for not only joint angles tracking but also the generation of GRF which propels the floating base. Optimization-based control is a natural solution to this multi-task problem. In [7] [8] [9], humanoid robot control is formulated as a quadratic programming problem where control inputs are found to maximize the objective function that weights each key trajectory tracking performance indicator differently. The trajectory that prescribes the motion of the floating base can be generated by assuming the robot is a simple linear inverted pendulum [10] [9]. On the other branch of bipedal walking research, hybrid zero dynamics (HZD) approach is proposed to provide a trade-off between joint angles tracking and floating base propelling tasks [11]. Under the HZD controller, the desired joint angles are not prescribed with respect to time but to a phase variable measuring the percentage of completion of one step. When each joint of the robot follows the desired joint angle, the system states travel only a low dimensional subspace of the state space. Within this subspace, the floating base trajectory also achieves periodical orbit. Therefore, the balance and tracking tasks are consistent within the low dimensional subspace and the only effort is to control the state onto this subspace.

Another example of the challenge of underactuated balance robots with limited actuations is the output tracking problem of unstable systems. In the cart-pole system example, the cart moves to maintain the balance of the pendulum while tracking the desired output trajectory. For such systems, no analytical causal compensator can achieve exact output tracking while maintaining the internal stability [12]. In [13], an external/internal convertible (EIC) dynamic structure is explored to separate the system into an external subsystem for the trajectory tracking task and an internal subsystem for the balancing task. A causal controller design is proposed in [13] to achieve both trajectory tracking and balancing tasks by enforcing the system state onto balance equilibrium manifold (BEM). On the BEM, the tracking task and the balancing task are no longer competing but consistent.

This dissertation takes a unifying view of underactuated balance robots by using latent manifold approach. The latent manifold is designed to encode the multiple control tasks for the robot so that these tasks can be achieved simultaneously once the state is on the manifold. For example, in the biped walking problem, the latent manifold is embodied as the virtual constraints which define how different joint angles coordinate and synchronize with each other [11]. On these virtual constraints, not only does each joint angle follow its desired trajectory, but the COM also follows periodical motion. In the bicycle balancing while tracking problem, the latent manifold is embodied as the BEM which defines how the bicycle roll angle equilibrium point depends on the desired bicycle position trajectory. Once the system state is stabilized on the BEM, the bicycle roll angle is stabilized around its equilibrium point while the bicycle position tracks desired output trajectory [13].

Despite the advantages of latent manifolds in underactuated balance system control, their applications have been limited to controller design for well structured analytical models. In many cases, learning models can predict the behavior of the system with higher accuracy than analytical models while requiring limited domain knowledge. This motivates the study on learning model-based controller design for latent manifold stabilization. In other cases, the latent manifold is not clearly defined but collected demonstration data indicates possible unknown latent manifold. This motivates the study on identifying latent manifolds from data. The fact that the learning-based dynamics reduction approach [14] and the analytical-based dynamics reduction approach [11] both reduce the system dimensions and simplify the model motivates the study on their relation and correspondence. For the modeling and control of high dimensional underactuated balance systems such as the human-bikebot interaction system, neither a physical-based model nor a learning-based model alone achieves the desired modeling and control performance. Learning-based latent manifold models provide accurate prediction for high dimensional systems, while physical-principled models provide physical interpretation. This motivates the study on integrating physical and learning models for human-robot interaction.

In the rest of this chapter, we first give an introduction to the latent manifold concept and then present the dissertation outline and a summery of the contributions.

1.2 Latent manifold concept

Mathematically, during certain motion, given the robot state $\boldsymbol{x} \in \mathbb{R}^n$, there exists a coordinate transformation $\tilde{\boldsymbol{x}} = \mathcal{T}(\boldsymbol{x})$ such that $\tilde{\boldsymbol{x}}_i \equiv 0$ constantly for $i = m+1, ..., n$, where $m < n$. The m-dimensional space $\tilde{\boldsymbol{x}}_i = \mathcal{T}_i(\boldsymbol{x})$, $i = 1, ..., m$, is the latent manifold while the $(n-m)$-dimensional equation $\mathcal{T}_i(\boldsymbol{x}) \equiv \mathbf{0}$ for $i = m+1, ..., n$, forms $n - m$ constraints. The n-dimensional dynamics also reduce to the m-dimensional dynamics. The latent manifold encodes the coordination of state variables and serves as a description of robot skill [15]. Therefore, identifying the latent manifold is a critical step towards understanding and replicating certain motion skill. In this dissertation, several approaches are discussed for constructing or identifying latent manifolds.

1.2.1 Virtual constraints

In the area of geometric control, control inputs are applied to force the system state onto the designed virtual constraints. Virtual constraints are not consequences of physical constraints but consequences of feedback control. For example, we want to control the single-input single-ouput system

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ 6 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u \\ y &= \begin{bmatrix} \alpha & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \end{aligned} \tag{1.1}$$

to have desired output $y = 0$. By differentiating the output once,

$$\dot{y} = \alpha\dot{x}_1 + \dot{x}_2 = 6x_1 + (1+\alpha)x_2 + u,$$

and designing the input as $u = -6x_1 - (1+\alpha)x_2 + v$, the output dynamics is forced to be $\dot{y} = v$. The choice of $v = -y$ results in output dynamics converging to 0. Here, the steady state $y = \alpha x_1 + x_2 = 0$ is the virtual constraint. Under this virtual constraint, the original second order dynamics is reduced to first order dynamics $\dot{x}_1 = x_2 = -\alpha x_1$, which is called zero dynamics. The coordinate transformation that transforms the original coordinates into zero dynamics state variable and output variable is

$$\tilde{\boldsymbol{x}} = \mathcal{T}\boldsymbol{x} = \begin{bmatrix} 1 & 0 \\ \alpha & 1 \end{bmatrix} \boldsymbol{x}. \tag{1.2}$$

Note that the choice of the zero dynamics state variable and therefore $\mathcal{T}$ is not unique. For example, we can choose $\tilde{x}_1 = -x_1 + \alpha x_2$ so that in the new coordinates $\tilde{x}_1 \perp \tilde{x}_2$. However, different coordinate systems are equivalent because they are connected through diffeomorphic transformations.

The choice of the virtual constraints is critical to the stability of the zero dynamics. The zero dynamics $\dot{x}_1 = -\alpha x_1$ is stable if and only if $\alpha > 0$, which is determined by the design of virtual constraints. If the zero dynamics is unstable, the zero dynamics state variable will go to infinity and the feedback control is infeasible.

1.2.2 Learning-based reduction approach

For high-dimensional robot motion, it is difficult to identify the latent manifold from physical-principled analysis. However, the robot state trajectory data can be easily collected. The rationale of the learning-based latent manifold modeling approach is that the motion dynamics is actually embedded in a low-dimensional subspace and this low-dimensional latent space can be identified with machine learning approaches.

Principal component analysis (PCA) is the most widely applied linear dimensional reduction approach. The fundamental idea of PCA is to find an orthonormal linear coordinate transformation $\mathcal{T} = \{\boldsymbol{u}_i\}, i = 1, ..., n$, so that the new coordinates $\tilde{\boldsymbol{x}}_i = \boldsymbol{u}_i^T \boldsymbol{x}$ has sampling variance λ_i in decreasing order. $\tilde{\boldsymbol{x}}_i = \boldsymbol{u}_i^T \boldsymbol{x}$ with small variance indicates that the values in the new coordinates are around zero. The first m coordinates are picked as the coordinates in latent manifold, while the last $n - m$ coordinates are neglected.

PCA can be modified into nonlinear dimension reduction algorithms by using the kernel concept. In [16], Gaussian process latent variable model is developed as a generalization of PCA in the probabilistic frame with nonlinear kernel. Instead of obtaining the explicit form of coordinate transformation, the algorithm outputs the corresponding coordinates in the latent space and hyperparameters of the mapping from the latent space to the original space.

One interesting point of view is to apply PCA to learn the state trajectory of linear system (1.1) under the virtual constraint based control. Because the motion data converges to the designed virtual constraint $y = \alpha x_1 + x_2 = 0$, the motion data has minimum variance in the direction of the virtual constraint and has maximum variance in its perpendicular direction. In Chapter 4, we prove that PCA can identify the linear virtual constraints, and the nonlinear dimension reduction approach can identify nonlinear virtual constraints.

Locally linear embedding (LLE) is a nonlinear dimension reduction approach that preserves the reconstruction of one point by its neighbors [17]. For N points $\boldsymbol{y}_i \in \mathbb{R}^D$, $i = 1, ..., N$, the LLE finds the d-dimensional latent space coordinates $\boldsymbol{x}_i \in \mathbb{R}^d$, $d \ll D$, which preserves the same reconstruction relation in the original space. The LLE consists of a single pass of three steps. First, the neighbors of each data point $\boldsymbol{y}_i$ are computed. Second, each point is reconstructed by its neighbors by minimizing $\mathcal{E}(W) = \Sigma_i |\boldsymbol{y}_i - \Sigma_j W_{ij} \boldsymbol{y}_j|^2$ with respect to W_{ij} under the constraints $\Sigma_j W_{ij} = 1$. Third, the reconstruction relations W_{ij} are preserved in the latent space. The latent coordinates are solved by minimizing $\Phi(\boldsymbol{x}) = \Sigma_i |\boldsymbol{x}_i - \Sigma_j W_{ij} \boldsymbol{x}_j|^2$ with respect to $\boldsymbol{x}_i$.

Nonlinear dimension reduction methods have been successfully applied to model, track and synthesize high-dimensional human motion. In [18], inverse kinematics solution is searched on the constructed latent space of human motion. The latent space of a particular human motion is trained from motion data. The algorithm outputs a human pose that not only satisfies prescribed constraints but also lies on the latent manifold. This guarantees that the found pose satisfies the constraints and is human-like. In [14], human walking joint angles are modeled by a latent space dynamics and a mapping from the latent space to the joint angle space. The latent dynamics approach significantly simplifies the model complexity without dramatically losing modeling accuracy. The work in [19] further applies this model to human tracking. In [20] [21], the prior knowledge of the latent space topology is enforced so that the learned performance is enhanced. For example, periodical walking motion should have closed circles as its latent space. Other physical knowledge and assumptions can be incorporated into the learning of latent space as in [22]. In [23], the labels of latent coordinates are used to further enhance the modeling accuracy. It is shown that incorporating prior knowledge into the nonlinear dimension reduction approach can increase both the model accuracy and its interpretability. In Chapter 2, a new dimension reduction algorithm is proposed by taking advantage of the prior information about human-robot interactions. In Chapter 4, we build a connection between the learning-based latent space model and the physical-based zero dynamics approach.

1.3 Dissertation outline and contributions

This dissertation consists of six chapters. Chapter 1 is the introduction. In Chapter 2, we integrate the physical-principled model with the learning-based model for high dimensional human-robot interaction modeling and state estimation. Human-bikebot interactions are taken as an example. In Chapter 3, physical-principled latent manifold approach is applied to biped slip recovery control. In Chapter 4, the relationship between the physical-principled latent manifold approach and the learning-based latent manifold approach is discussed with bipedal walking as an example. The ultimate goal of Chapters 3 and 4 is to quantify walking and slip recovery skills with latent manifolds and transfer skills between human and biped robots. In Chapter 5, a planning and control framework is proposed for underactuated balance robots modeled by the learning-based approach. Chapter 6 concludes the dissertation and discusses the future research directions.

The modeling and control of physical human-robot interactions (pHRI) is challenging because of the high dimensionality of the system and the various ways of interactions. However, for many pHRI activities, high dimensional human joint angles are coordinated to achieve simple tasks. This promises an opportunity to model it in the latent manifold. In Chapter 2, an integrated physical-learning modeling framework for pHRI is proposed. A latent manifold-based learning model is applied to the modeling of high-dimensional human joints, while the physical-principled model provides a trustworthy and physically interpretable model for the robot and human trunk. The learning-approach modeled subsystem is coupled with the physical-approach modeled subsystem through system inputs and constraints. Existing dimension reduction approaches such as PCA and LLE do not endow physical meaning to the constructed latent manifold. We present the axial linear embedding (ALE) algorithm to construct latent variables with physical meaning. The proposed framework is applied to the bikebot riding pose estimation problem. The contribution of this chapter lies in the novel approach to integrate the physical-principled model with the leaning-based latent manifold model through inputs and constraints. The proposed dimensional reduction algorithm ALE is also a contribution.

The hybrid zero dynamics (HZD) approach has been proposed to control biped robot to achieve periodical walking. In Chapter 3, we propose an HZD based control design for biped slip recovery. During the slip phase, the robot has more degrees of freedom, i.e. the additional slipping displacement of the foot, making the system even harder to control. However, through a sequence of gaits under different contact property, the robot can be controlled back to periodical walking gait. Each gait is described by virtual constraints that reduce the system state onto the zero dynamics space. The recovery sequence consists of multiple gaits that are connected smoothly in sequence. The system state is driven back to normal walking periodical orbit under this recovery sequence. The contribution of this chapter is the proposed criteria of designing virtual constraints for slip recovery.

Since both the learning-based and physical-based latent manifold models are applied to model high-dimensional systems, it is natural to reveal their relationship and characteristics. In Chapter 4, we present the correspondence between the learning-based latent space model and the analytical virtual constraints. It is shown that the learned latent dynamics is equivalent to the zero dynamics. The mapping from the latent manifold to the high dimensional state space is equivalent to the virtual constraints. Human walking experiments are used to validate the findings. This discovery can be applied to classifying gaits and potentially transferring human walking skill to bipedal robots.

Despite the mathematical elegance of the latent manifold concept, identifying the latent manifold and using it in controller design are only manageable for well understood physical-principled models. In Chapter 5, we propose a learning model-based control framework for underactuated balance robots. The system model is obtained using Gaussian processes (GPs) model learned from experimental data. GPs provide a way to quantify modeling uncertainty. Model predictive control (MPC) is adopted for the trajectory tracking task to compute the latent manifold. A learned inverse dynamics controller then stabilizes the system state onto the latent manifold. The planning and control framework takes advantage of the modeling uncertainty information to achieve robust control performance. Hardware experiments on a rotary base inverted pendulum show that without prior physical-principled knowledge of the system nor success balancing demonstration, the learning model-based control performance is superior to that of the analytical model-based controller. The contribution of this chapter lies in three aspects. First, the proposed control framework is based on Gaussian process learning model instead of physical-principled models. The fact that this framework does not depend on the understanding of the robot dynamics structure broaden the application of latent manifold concept. Second, the model learning is efficient and does not require successful balance demonstration. Finally, the proposed framework is robust to modeling error due to the novel way of exploiting model uncertainties provided by Gaussian processes models.

Chapter 2

Integrated Physical-Learning Model for Human-Robot Interaction

2.1 Introduction

Human with trained motor skills can fluidly and flexibly interact with machines while smart machines or robots can also provide motor assistance and enhancement to facilitate human's motor skills learning [24]. Modeling of physical human-robot interactions (pHRI) is important to understand the role of the human sensorimotor control in the trained motor skills with machines or robots and to design human assistive and rehabilitation systems. One of the main challenges for modeling pHRI is the high dimensionality and complexity of human motion and its interactions with machines [25]. The goal of this chapter is to present an integrated physical-learning pHRI model with applications to the bikebot (i.e., bicycle-based robot) riding example.

The proposed pHRI modeling framework uses the physical principles to model the dynamic motions of the robot and the human trunk while a machine-learning-based method is employed to capture the human limb motion in a low-dimensional latent space [26]. The physical model and the learning model are interconnected and integrated to describe the pHRI. The rationales of using the integrated physical-learning pHRI modeling approach are twofold. First, one main challenge of modeling and control of pHRI is the high-dimensional human motion and anatomical redundancy of human body segments. Using physical principles for rigid body dynamics, such as

Lagrangian or Newtonian mechanics, generates high-dimensional models that are difficult to be used for control systems and estimation design. For many human activities, high-dimensional motions in physical joint space are highly coordinated and can thus be represented by low-dimensional latent dynamics [27]. Second, the physical modeling approach is commonly used to describe the robot dynamics. The physical interactions between the human and the robot, such as forceful contacts and coordinated movements, provide additional properties and constraints for both the human and the robot dynamics. The two modeling methods are complementary and the integration of the physical model for the robot and the learning model for human motion naturally provides a means to capture and incorporate the interaction characteristics.

Dimensional reduction and manifold learning methods are used to capture human motion characteristics for applications such as humanoid robots control [28–30] and human tracking and activity recognition [14, 21, 31, 32]. Low-dimensional manifolds are learned and used to capture the high-dimensional human movements. In [15], motion planning is conducted on the learned skill manifolds without need to build the analytical robotic models. Embedded skill manifolds are also presented in [33, 34] to represent and encode human hand motion and compare the motion capability between the robotic hands with the human hands. Nonlinear dimensionality reduction and Gaussian process latent variable models (GP-LVM) are used in [33, 34] to obtain the skilled manifolds.

One potential drawback of using machine learning or data-driven modeling approach is that commonly used dimensionality reduction algorithms (e.g., principal component analysis (PCA) or locally linear embedding (LLE)) do not preserve the physical meaning for the obtained low-dimensional latent variables. It is difficult to interpret these variables and build physical connections with joint angles. To overcome this shortcoming, we present an axial linear embedding (ALE) algorithm to conduct the dimensional reduction that preserves the physical interpretation of the latent variables.

Besides the physical interpretation, the use of ALE also demonstrates superior performance than other commonly used dimensional reduction approaches, such as PCA or LLE [27].

In [14], a Gaussian process dynamic model (GPDM) is proposed to capture high-dimensional time series data with application to human motion animation. In [23], a GPBF-LEARN model is introduced to improve GPDM for robotics applications by incorporating control input and initialization of latent coordinates. A predictive model in the low-dimensional latent space and an observation model mapping from the latent space to the high-dimensional data space are presented. Because of the non-convex optimization in the learning problem, the learned model is subject to local optimum and hence the initialization (labels) of the latent coordinates is critical for performance. The GPBF-LEARN approach shows the advantage of labeling latent coordinates with their observed values. In this chapter, we adopt GPBF-LEARN's structure and incorporate the ALE method for dimensionality reduction to construct physically meaningful labels for latent coordinates corresponding to the high-dimensional joint angles. Bikebot riding experiments show that this method outperforms other algorithms such as PCA and LLE for latent coordinates initialization.

The work in [35] try to use both the physical model and the learning model. The model integration in [35] is different from what we present here in this chapter. Instead of establishing the dynamic relationship between the inputs and the outputs, the learning model in [35] captures the difference between the experiments data and the predictions by the physical model. Therefore, the learning model serves as a correction to the physical model. Instead, we consider physical and learning modeling approaches for different types of coupled physical systems (i.e., robot and human body segments, respectively) and use the physical interactions and constraints to enhance the modeling accuracy. Such treatment take advantages of both models' attractive, complementary properties, such as simple, low-dimensional representation of the latent models and the high-fidelity, physically interpretable dynamics models.

Bikebot riding is used as an example to test and validate our modeling method. Bikebot is an actively controlled bicycle-based robot that was developed for studying human balancing motor skills [5]. We use bicycle-like riding as the testbed primarily for two reasons. First, riding the bicycle requires the coordinated control of multi-limb and body movements and thus, offers an attractive platform for studying human postural balance motor skills. Secondly, the interactions between the rider and the bikebot are through multiple forceful contacts, such as steering handlebar, seat and pedaling etc. These interactions provide dynamic and geometric constraints between the rider and the bikebot. In [36], a pose estimation scheme is presented to real-time predict trunk and bicycle orientations from the inertial measurement units (IMU) and seat force sensor measurements. Unlike the work in [36], the main advantage of using the physical-learning model is that no IMU sensors are needed to attach to rider's limb segments.

The remainder of this chapter is organized as follows. The physical-learning modeling approach is presented in Section 2.2. We present the modeling application to the rider-bikebot system in Section 2.3. The extended Kalman filter (EKF)-based pose estimation is discussed in Section 2.4 and experiments are presented in Section 2.5. Finally, we conclude the chapter in Section 2.6.

2.2 Physical-learning modeling framework

2.2.1 Model overview

The integrated physical-learning model uses the physical principles to capture the dynamic motion of the robot and the human trunk and the machine learning-based model to describe the limb dynamic motion. The human-robot interaction forces and geometric constraints are used to integrate the physical and learning models. Figure 2.1

illustrates the physical-learning pHRI modeling framework.

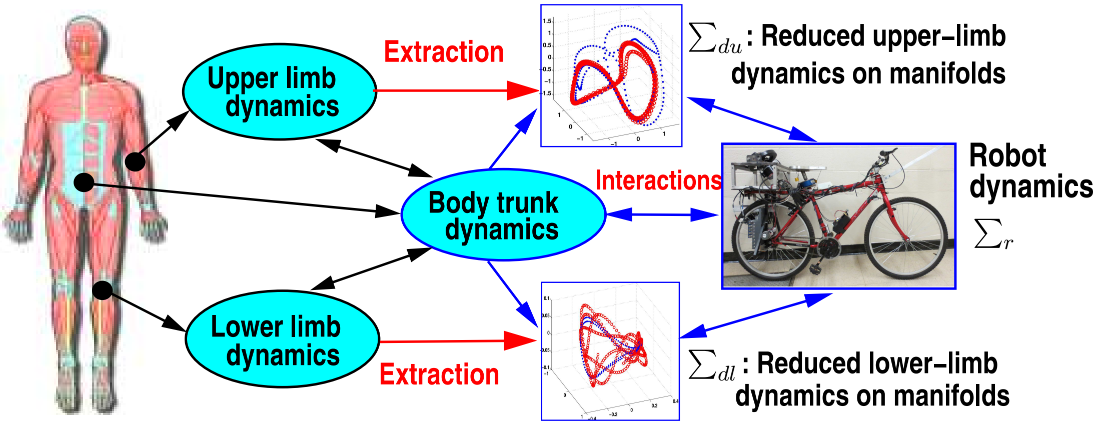


Figure 2.1: Concept of the integrated physical-learning pHRI modeling framework.

The physical-learning pHRI model is written as $\Sigma_{pl} = \Sigma_r \times \Sigma_t \times \Sigma_{du} \times \Sigma_{dl}$, where Σ_r denotes robot dynamics, Σ_t is human trunk dynamics and Σ_{du} and Σ_{dl} are the upper- and lower-limb dynamics in the low-dimensional latent space. Note that Σ_r and Σ_t are obtained through physical principles (e.g., Lagrangian mechanics) while Σ_{du} and Σ_{dl} are obtained through dimensionality reduction and machine learning techniques. Let $\boldsymbol{q}_r \in \mathbb{R}^{n_r}$ and $\boldsymbol{q}_t \in \mathbb{R}^{n_t}$ denote the generalized coordinates (with dimensions n_r and n_t) for the robot and the human trunk motions, respectively. Let $\boldsymbol{q}_{rt} = [\boldsymbol{q}_r^T \; \boldsymbol{q}_t^T]^T \in \mathbb{R}^{n_t+n_r}$ and the robot-trunk dynamics are obtained as

$$\boldsymbol{M}(\boldsymbol{q}_{rt})\ddot{\boldsymbol{q}}_{rt} + \boldsymbol{C}(\boldsymbol{q}_{rt}, \dot{\boldsymbol{q}}_{rt}) + \boldsymbol{G}(\boldsymbol{q}_{rt}) = \boldsymbol{u}_t\,, \tag{2.1}$$

where $\boldsymbol{M}(\boldsymbol{q}_{rt})$, $\boldsymbol{C}(\boldsymbol{q}_{rt}, \dot{\boldsymbol{q}}_{rt})$, and $\boldsymbol{G}(\boldsymbol{q}_{rt})$ are the inertia matrix, Coriolis and gravity vectors, respectively, and $\boldsymbol{u}_t$ is the torque input for human trunk and the robot. In Section 2.3, we will illustrate how to obtain (2.1) for the rider-bikebot system.

Our choice to use physical models to capture human trunk motion and learning models for limb motion primarily lies in several considerations. The first consideration is the dimensionality of the trunk and the limb motions in human locomotions and activities. For many human activities such as riding a bicycle (bikebot), the trunk movement can be captured as a rigid 3 degree-of-the-freedom (DOF) inverted pendulum [36], while the motions of each limb have more than 5 DOFs. It is undesirable to use physical models to capture the constrained high-DOF limb motions due to their complexity. Another motivation to use learning models to capture the limb motion lies in the fact of the coordinated motion pattern among limbs in many human activities, such as bicycling, walking, running, etc. It is of advantages to use learning models to capture these coordination characteristics on low-dimensional manifolds rather than using any constrained physical models.

The dynamics of the bikebot-trunk given by (2.1) are used as the physical models because the motions are resulted from forces and torques in the pHRI. Moreover, for the pose estimation applications, we take advantages of the non-drifting property of the measured human-robot interactions forces/torques to reduce or eliminate the drifts due to inertial sensor noises. Integration of the force and inertial measurements provides a robust means to completely eliminate the estimation drifts due to inertial sensor noises [36]. Therefore, using the bikebot-trunk dynamic constraints can improve the accuracy of the pose estimation as shown in the experiments in Section 2.5. Although the learning models presented in this work are built on conveniently obtained joint angles, the physical-learning modeling framework is not restricted to the kinematics information and the dynamics of limb motions can also be used and integrated into the learning models.

2.2.2 Learning model for human limb motion

We adopt machine-learning-based latent dynamic model to represent the limb motion. The latent dynamic model consists of two parts: a predictive latent state dynamics and an observation model that maps low-dimensional latent variables to high-dimensional joint angles. We denote the latent state variable as $\boldsymbol{x} \in \mathbb{R}^d$ and the limb joint angles as $\boldsymbol{y} \in \mathbb{R}^D$, where d and D ($d \ll D$) are the dimensions of the latent space and the joint angels space, respectively. Figure 2.2 illustrates the latent dynamics model structure. For presentation clarity, we use discrete-time representation for the learning model. To capture the physical human-robot interaction, a control input $\boldsymbol{u}_h \in \mathbb{R}^{n_h}$ and a geometric constraint $\boldsymbol{z} \in \mathbb{R}^{n_c}$ are used in the latent dynamic model, where n_h and n_c are the dimensions of the control inputs and the constraints, respectively.

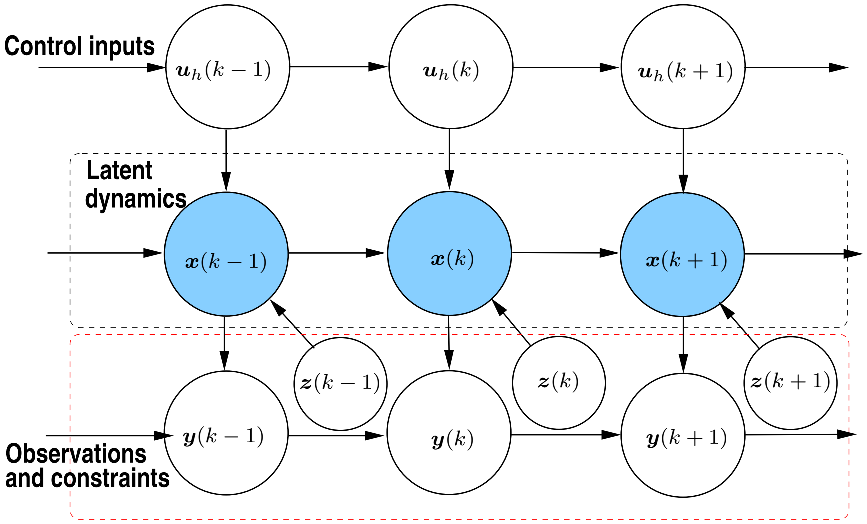


Figure 2.2: A graphical model of the latent dynamics structure.

The latent dynamics Σ_l for limb motion are formulated as

$$\delta \boldsymbol{x}(k) = \boldsymbol{f}(\boldsymbol{x}(k-1), \boldsymbol{u}_h(k-1), \boldsymbol{\alpha}) + \boldsymbol{w}_p, \tag{2.2a}$$

$$\boldsymbol{y}(k) = \boldsymbol{g}(\boldsymbol{x}(k), \boldsymbol{\beta}) + \boldsymbol{w}_o, \tag{2.2b}$$

where $\delta \boldsymbol{x}(k) := \boldsymbol{x}(k) - \boldsymbol{x}(k-1)$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are system parameters, $\boldsymbol{w}_p$ and $\boldsymbol{w}_o$ model the noises. For training data set $\{\boldsymbol{y}(k)\}^N$ and control input set $\{\boldsymbol{u}_h(k)\}^N$, we estimate maps $\boldsymbol{f}(\cdot,\cdot,\cdot)$ and $\boldsymbol{g}(\cdot,\cdot)$ in (2.2) by identifying $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. Denoting $\boldsymbol{X} = \{\boldsymbol{x}(k)\}^N$, $\boldsymbol{Y} = \{\boldsymbol{y}(k)\}^N$ and $\boldsymbol{U} = \{\boldsymbol{u}_h(k)\}^N$, the system identification problem is formulated as maximizing a-posterior distribution $P(\boldsymbol{X}, \boldsymbol{\alpha}, \boldsymbol{\beta} | \boldsymbol{Y}, \boldsymbol{U}, \hat{\boldsymbol{X}})$. Here $\hat{\boldsymbol{X}} = \{\hat{\boldsymbol{x}}(k)\}^N$ (i.e., label of $\boldsymbol{X}$) is used to initialize $\boldsymbol{X}$ in the optimization process.

We adopt the GPBF-LEARN structure to factorize the objective function as

$$P(\boldsymbol{X}, \boldsymbol{\alpha}, \boldsymbol{\beta} | \boldsymbol{Y}, \boldsymbol{U}, \hat{\boldsymbol{X}}) \propto P(\boldsymbol{Y} | \boldsymbol{X}, \boldsymbol{\beta}) P(\boldsymbol{X} | \boldsymbol{U}, \boldsymbol{\alpha}) P(\boldsymbol{X} | \hat{\boldsymbol{X}}) P(\boldsymbol{\alpha}) P(\boldsymbol{\beta}). \tag{2.3}$$

The first term $P(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{\beta})$ is factorized as the product of D Gaussian process regression models with each one corresponding to the regression of the ith dimension of $\boldsymbol{y} \in \mathbb{R}^D$. Denoting $\mathrm{y}_i = \{\boldsymbol{y}_i(k)\}^N$, we have

$$P(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{\beta}) = \prod_{i=1}^{D} P(\mathrm{y_i}|\boldsymbol{X},\boldsymbol{\beta}_\mathrm{i}) = \prod_{\mathrm{i=1}}^{\mathrm{D}} \mathcal{N}(\mathrm{y_i}|\mathbf{0},\boldsymbol{K}_{\mathrm{y_i}} + \sigma^2_{\mathrm{n_i}}\mathrm{I}),$$

where $\boldsymbol{K}_{y_i}$ is an $N \times N$ kernel matrix with $\boldsymbol{K}_{y_i}[p,q] = k_{y_i}(\boldsymbol{x}_p,\boldsymbol{x}_q)$. A squared exponential kernel is chosen as $k_{y_i}(\boldsymbol{x}_p,\boldsymbol{x}_q) = \sigma^2_{f_i} e^{-\frac{1}{2}(\boldsymbol{x}_p-\boldsymbol{x}_q)^T \boldsymbol{W}_i(\boldsymbol{x}_p-\boldsymbol{x}_q)}$, where $\boldsymbol{W}_i$ is diagonal matrix for weighting different inputs. $\boldsymbol{\beta}_i = \{\boldsymbol{\sigma}_{f_i}, \boldsymbol{W}_i, \boldsymbol{\sigma}_{n_i}\}$ is the hyper-parameter set trained for the ith dimension Gaussian process regression. To model observation function $\boldsymbol{g}$ in (2.2b), D-dim GPDM hyper-parameter sets $\boldsymbol{\beta} = \{\boldsymbol{\beta}_i\}^D$ are learned and obtained.

To estimate the latent dynamics (2.2a), we use the Gaussian process model to learn the regression relationship between $\delta x_i(k) = x_i(k) - x_i(k-1)$ and $\boldsymbol{s}(k-1) = [\boldsymbol{x}^T(k-1)\;\; \boldsymbol{u}_h^T(k-1)]^T$, $i = 1, 2, \cdots, d$. Therefore, the second term in (2.3) is factorized as

$$P(\boldsymbol{X}|\boldsymbol{U},\boldsymbol{\alpha}) = \prod_{i=1}^{d} P(\boldsymbol{x}_i|\boldsymbol{U},\boldsymbol{\alpha}_i) = \prod_{i=1}^{d} \mathcal{N}(\delta\boldsymbol{x}_i|\mathbf{0},\boldsymbol{K}_{x_i} + \sigma^2_{x_{n_i}}\boldsymbol{I}),$$

where $\boldsymbol{x}_i = \{\boldsymbol{x}_i(k)\}^N$, $\delta\boldsymbol{x}_i = \{\delta x_i(k)\}^N$ and $\boldsymbol{K}_{x_i}[p,q] = k_{x_i}(\boldsymbol{s}_p,\boldsymbol{s}_q)$ is the kernel function for the ith dimension of the predictive function (2.2a) and $k_{x_i}(\boldsymbol{s}_p,\boldsymbol{s}_q) = \sigma^2_{x_{f_i}} e^{-\frac{1}{2}(\boldsymbol{s}_p-\boldsymbol{s}_q)^T \boldsymbol{W}_{x_i}(\boldsymbol{s}_p-\boldsymbol{s}_q)}$. Terms $\boldsymbol{\alpha}_i = \{\sigma_{x_{f_i}}, \boldsymbol{W}_{x_i}, \sigma_{x_{n_i}}\}$ are the hyper-parameters learned for the ith dimension of the predictive function mapping $\boldsymbol{f}$. Up to d-dim GPDM regression hyper-parameter sets $\boldsymbol{\alpha} = \{\boldsymbol{\alpha}_i\}^d$ are learned.

The third term in (2.3) is expressed as an identically independent Gaussian distribution with preset observation noise variance $\sigma^2_{\hat{\boldsymbol{x}}_t}\boldsymbol{I}$, namely,

$$P(\boldsymbol{X}|\hat{\boldsymbol{X}}) = \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{x}(k)|\hat{\boldsymbol{x}}(k), \sigma^2_{\hat{\boldsymbol{x}}(k)}\boldsymbol{I}).$$

This term expresses the confidence of the label of the latent coordinates. With each term in (2.3) specified, the learning algorithm takes the training data $\boldsymbol{Y}$ and $\boldsymbol{U}$ and initial values $\hat{\boldsymbol{X}}$, $\boldsymbol{\alpha}_{ini}$ and $\boldsymbol{\beta}_{ini}$ to optimize (2.3) with respect to $\boldsymbol{X}$, $\boldsymbol{\alpha}$, and $\boldsymbol{\beta}$. The optimization process is implemented with a scale conjugate gradient algorithm [23].

Constraints $\boldsymbol{z}$ include the geometric and dynamics relationships and commonly exist in the physical human-robot interactions. These constraints can be integrated and incorporated during the stage to obtain the learning models such as the approach discussed in [37]. In this work, we will use the EKF to fuse various sensing information for the pose estimation application and the integration of constraints $\boldsymbol{z}$ into the EKF design will be presented and discussed in Section 2.4.

2.2.3 Learning model initialization

Because of the non-convexity of the above optimizing process, the initialization of $\boldsymbol{X}$ is critical to avoid the local minimums and obtain the correct results. We propose a novel latent coordinates labeling approach, termed as axial linear embedding (ALE). The construction of ALE reduction algorithm is inspired by observation that the limb motion heavily depends on the trunk motion and the limb-robot interactions. This observation comes from the fact that the two endpoints of each limb are connected to the trunk (limb-shoulder connection) and the robot (hand- or foot-robot interaction). For example, when riding a bicycle, two hands are always holding the handlebar and two feet are always in contact with the pedals. Therefore, the limb segment poses are mainly determined by the trunk and the bicycle motion.

The ALE algorithm is illustrated in Algorithm 2.1. We construct a set of motion primitives in latent space. We define an equilibrium point $\boldsymbol{q}_{rt}^{e}$ at which the pHRI is at either stable or comfortable locations. For example, for riding a bicycle, $\boldsymbol{q}_{rt}^{e}$ is defined when rider sits on the seat with two arms at comfortable, natural position on handlebars and bicycle at vertical position and zero steering angle. At $\boldsymbol{q}_{rt}^{e}$, the limb joint angles are at $\boldsymbol{y}^{e}$. To learn the model effectively, we subtract all the limbs joint angles by $\boldsymbol{y}^{e}$ so that the center of the joint angles space is at zero. We abuse the notation slightly and still use $\boldsymbol{y}$ to denote the de-centered joint angles without causing any confusion. Then, we perturb human's motion around $\boldsymbol{q}_{rt}^{e}$ by only moving along jth coordinate direction of $\boldsymbol{q}_{rt}^{j}$, $j = 1, 2, \cdots, n_{rt}$, $n_{rt} = d$ is the dimension of the latent space, and record the limb joint angles set $\{\boldsymbol{y}_l\}_{\boldsymbol{q}_j}$. We call the set of this perturbed motion as the *template* of $\boldsymbol{q}_{rt}$, denoted as $\mathcal{T}_{rt} = \mathrm{span}\{\boldsymbol{q}_{rt}^{j}\}$. Using the PCA method, we factorize the first principal component in latent space to obtain $\{x_l\}_{\boldsymbol{q}_j} = \mathtt{PCA}_1(\{\boldsymbol{y}_l\}_{\boldsymbol{q}_j})$. After n_{rt} experiment runs, $\{x_l\}_{\boldsymbol{q}_j}$ are obtained for the d-dimensional latent space. This process is shown in lines 1-3 in Algorithm 2.1.

For a training set $\{\boldsymbol{y}(k)\}^{N}$, ALE finds the latent label $\boldsymbol{x}(k)$ for $\boldsymbol{y}(k)$ by first finding M closest points around $\boldsymbol{y}(k)$ that are in the basic movements sets $\cup_{j}^{n_{rt}}\{\boldsymbol{y}_l\}_{\boldsymbol{q}_j}$ (line 4), approximating $\boldsymbol{y}(k)$ as a linear regression of these M points (line 5), and then keeping this approximation relation in latent space (line 6). This projecting approach is similar to LLE, which preserves the reconstruction relation of a point relative to its neighbors in high dimension into the latent space. However, our approach preserves the high dimensional space reconstruction relation of a point relative to neighboring points on the axes into the latent space.

Algorithm 2.1: Axial Linear Embedding (ALE)

```
   for $j = 1$ to $n_{rt}$ do
1      $\boldsymbol{q} = \boldsymbol{q}_{rt}^{e}$; $\boldsymbol{y} = \boldsymbol{y}^{e}$;
2      Perturb $\boldsymbol{q}$ along the $j$th template of $\mathcal{T}_{rt}$ and obtain limb motion $\{\boldsymbol{y}_l\}_{\boldsymbol{q}_j}$;
3      $\{x_l\}_{\boldsymbol{q}_j} = \mathtt{PCA}_1(\{\boldsymbol{y}_l\}_{\boldsymbol{q}_j})$; $\{\boldsymbol{x}_l\}_{\boldsymbol{q}_j} = \boldsymbol{0}$; $\{\boldsymbol{x}_l^{j}\}_{\boldsymbol{q}_j} = \{x_l\}_{\boldsymbol{q}_j}$;
   end
   for $k = 1$ to $N$ do
4      Find $M$ points $\boldsymbol{y}_{k_i}$ s.t. $\|\boldsymbol{y}_{k_i} - \boldsymbol{y}(k)\|_2$ are the $M$ smallest value for
         $\boldsymbol{y}_{k_i} \in \cup_{j=1,\cdots,n_{rt}}\{\boldsymbol{y}_l\}_{\boldsymbol{q}_j}$;
5      $\boldsymbol{w}_{k_i} = \arg\min_{\boldsymbol{w}_{k_i}} \|\boldsymbol{y}(k) - \sum_{i=1}^{M} \boldsymbol{w}_{k_i}\boldsymbol{y}_{k_i}\|_2^2$, $\sum_{i=1}^{M} \boldsymbol{w}_{k_i} = 1$;
6      $\hat{\boldsymbol{x}}(k) = \Sigma_{i=1}^{M}\boldsymbol{w}_{k_i}\boldsymbol{x}_{k_i}$, $\boldsymbol{x}_{k_i}$ is the latent label of $\boldsymbol{y}_{k_i}$;
   end
```

Note that the equilibrium point $\boldsymbol{q}_{rt}^{e}$ of the template $\mathcal{T}_{rt}$ indeed corresponds to the origin of the latent space $\boldsymbol{x}$ by the above construction. The ALE constructs the latent space with the following properties: (1) origin $\mathbf{0} \in \mathbb{R}^d$ in the latent space maps to the equilibrium $\boldsymbol{y}^e \in \mathbb{R}^D$ in the limb joint angles space; (2) points in one latent space axis $\boldsymbol{x}_i$, $i = 1, \cdots, d$, map to the limb poses corresponding to one motion primitive in template $\mathcal{T}_{rt}$; and (3) any limb movements can be decomposed onto motion primitives $\cup_j^{n_{rt}} \{\boldsymbol{y}_l\}_{\boldsymbol{q}_j}$, namely, an arbitrary point in the limb joint angles space is approximated by a linear combination of motion primitives. By intentionally choosing motion primitives, the latent axes preserves the physical meaning, namely, the coordinates along motion primitive directions.

2.2.4 Mapping from the latent space to the physical space

Once the label $\hat{\boldsymbol{X}} = \{\hat{\boldsymbol{x}}_i\}$ is obtained from ALE, we apply GPDM to identify the latent dynamics model (2.2a) as discussed in Section 2.2.2. High-dimensional joint angles $\boldsymbol{y}(k)$ are estimated from $\boldsymbol{x}(k)$. We denote the training data sets for prediction (2.2a) and observation (2.2b) as $\boldsymbol{T}_p = \{\{\boldsymbol{x}(i-1)\}^N, \{\delta\boldsymbol{x}(i)\}^N\}$ and $\boldsymbol{T}_o = \{\{\boldsymbol{x}(i)\}^N, \{\boldsymbol{y}(i)\}^N\}$, respectively. The latent coordinates are given from the model as

$$P(\delta\boldsymbol{x}(k)|\boldsymbol{s}(k-1), \boldsymbol{T}_p) \propto \quad \mathcal{N}(\delta\boldsymbol{x}(k)|GP_\mu(\boldsymbol{s}(k-1), \boldsymbol{T}_p), GP_\Sigma(\boldsymbol{s}(k-1), \boldsymbol{T}_p)), \tag{2.4}$$

where $GP_\mu(\boldsymbol{s}(k-1), \boldsymbol{T}_p) = \boldsymbol{k}_*^T(\boldsymbol{K}+\sigma_n^2\boldsymbol{I})^{-1}([\delta\boldsymbol{x}(i)])$, $GP_\Sigma(\boldsymbol{s}(k-1), \boldsymbol{T}_p) = \boldsymbol{k}(\boldsymbol{s}(k-1), \boldsymbol{s}(k-1)) - \boldsymbol{k}_*^T(\boldsymbol{K}+\sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{k}_*$, and $[\delta\boldsymbol{x}(i)]$ denotes a column vector obtained by stacking $\delta\boldsymbol{x}(i)$. $\boldsymbol{K}$ is implemented with $K[p,q] = \sigma_f^2 e^{-\frac{1}{2}(\boldsymbol{s}_p-\boldsymbol{s}_q)^T W(\boldsymbol{s}_p-\boldsymbol{s}_q)}$, where $\boldsymbol{s}_p$ and $\boldsymbol{s}_q$ are from training data sets. $\boldsymbol{k}_*$ is realized with $\boldsymbol{k}_*[p] = \sigma_f^2 e^{-\frac{1}{2}(\boldsymbol{s}(k-1)-\boldsymbol{s}_q)^T W(\boldsymbol{s}^*-\boldsymbol{s}_p)}$. The mean value of this distribution is added by $\boldsymbol{y}^e$ to obtain the estimates of the joint angles.

In the EKF design in Section 2.4, we need to calculate $\frac{\partial\delta x(k)}{\partial x(k-1)}$ to obtain the Jacobian matrix of (2.2a). Noting that in (2.4) only $\boldsymbol{k}_*$ in $GP_\Sigma(\boldsymbol{s}(k-1), \boldsymbol{T}_p))$ is a function of

$\boldsymbol{x}(k-1)$, we obtain

$$\frac{\partial \delta x(k)}{\partial x(k-1)} = \frac{\partial GP_\mu(\boldsymbol{s}(k-1), \boldsymbol{T_p})}{\partial \boldsymbol{k_*}} \frac{\partial \boldsymbol{k_*}}{\partial x(k-1)}. \tag{2.5}$$

Similarly, the joint angles are obtained from (2.2b) as

$$P(\boldsymbol{y}(k)|\boldsymbol{x}(k), \boldsymbol{T}_o) \propto \mathcal{N}(\boldsymbol{y}(k)|GP_\mu(\boldsymbol{x}(k), \boldsymbol{T}_o), GP_\Sigma(\boldsymbol{x}(k), \boldsymbol{T}_o)), \tag{2.6}$$

where $GP_\mu(\boldsymbol{x}(k), \boldsymbol{T}_o) = \boldsymbol{k}_*^T(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}\mathbf{y}$ and $GP_\Sigma(\boldsymbol{x}(k), \boldsymbol{T}_o) = \boldsymbol{k}(\boldsymbol{x}(k), \boldsymbol{x}(k)) - \boldsymbol{k}_*^T(\boldsymbol{K} + \sigma_n^2\boldsymbol{I})^{-1}\boldsymbol{k}_*$. The Jacobian of (2.2b) is the obtained

$$\frac{\partial \boldsymbol{y}(k)}{\partial \boldsymbol{x}(k)} = \frac{\partial GP_\mu(\boldsymbol{x}(k), \boldsymbol{T_o})}{\partial \boldsymbol{k_*}} \frac{\partial \boldsymbol{k_*}}{\partial \boldsymbol{x}(k)}. \tag{2.7}$$

2.3 Rider-bikebot systems

In this section, we use rider-bikebot interactions as an example to illustrate the physical-learning modeling framework.

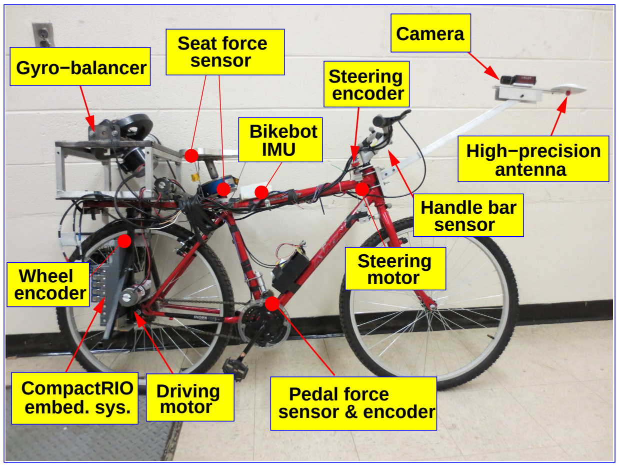
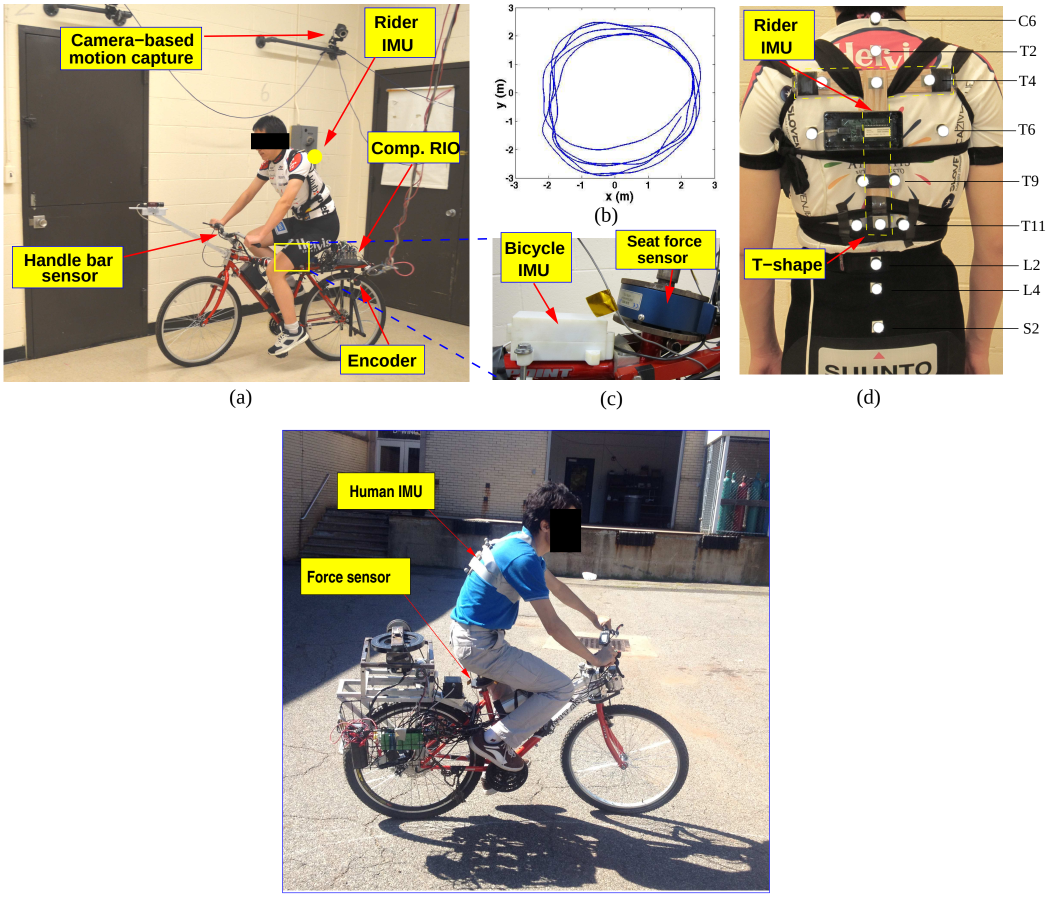


Figure 2.3: Bikebot systems configuration.

2.3.1 Bikebot system configuration

Figure 2.3 shows the bikebot system configuration. The bikebot is a modified bicycle with augmented steering, pedaling and balancing actuation to understand and study human sensorimotor balancing skills through unstable rider-bicycle interactions [5]. The



(a) (b) (c) (d)

(e)

Figure 2.4: (a) Bikebot indoor riding experiment. (b) Bikebot trajectory in one test riding. (c) Bikebot IMU and seat force sensor. (d) Locations of the IMU and optical markers mounted on rider trunk. (e) Bikebot outdoor riding experiment.

bikebot is equipped with various sensors, such as IMU and seat force sensor, as shown in the figure. When the actuators are not powered, the bikebot functions the same as a regular bicycle. Figure 2.4(a) shows the indoor riding experiments and Figure 2.4(e) shows the outdoor riding experiments.

Figure 2.5 illustrates the kinematic schematic of the rider-bikebot interactions. The rider's upper-body is considered as an inverted pendulum in three-dimensional (3D) space with its length, mass, and mass mass moment of inertia denoted as h_h, m_h, and J_h, respectively. A ground-fixed inertial frame $\mathcal{I}\,(X, Y, Z)$ is defined with the Z-axis downwards. A moving frame $\mathcal{R}\,(x, y, z)$ is defined with the x-axis along wheel-ground contact points C_1 and C_2, the z-axis along the Z-axis, and the origin at C_2. The bikebot roll and yaw angles and steering angle are denoted as φ_b, ψ, and γ, respectively. The gyroscope on the bikebot frame is tilted by angle α with respect to the x-axis. Let $\mathcal{I}_h$ and $\mathcal{I}_b$ denote the rider and bikebot gyroscope frames, respectively. The orientation of the trunk is defined by three Euler angles with the X-Y-X ordered rotation from frames $\mathcal{R}$ to $\mathcal{I}_h$: roll angle φ_h around the x-axis, angle θ around the y-axis, and finally self-spinning angle ϕ around the x-axis [36]. The zero-lateral velocity nonholonomic constraint of C_2 is considered and the bikebot's velocity is denoted as v_{rx}.

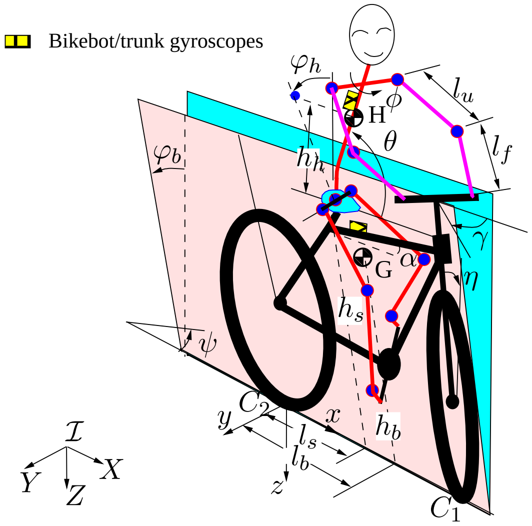


Figure 2.5: Schematic of the rider-bikebot interaction.

We mainly focus on and present the modeling results for the upper-limb motion and the lower-limb motion can be similarly obtained. The rider's upper-limb poses are specified from the trunk orientation. We define the right and the left upper-arm poses are given by Y-Z-X rotation from the trunk frame (i.e., 3 degree-of-the-freedom (3-DOF) shoulder joint) with joint angle sets (y_1, y_2, y_3) (left) and (y_6, y_7, y_8) (right), respectively. The elbow joints are assumed as a 2-DOF joint and the right and the left fore-arms are obtained by Y-X rotations with joint angle pairs of (y_4, y_5) (left) and (y_9, y_{10}) (right), respectively. The wrist joint is assumed fixed with the handlebar. Thus, the upper-limb is modeled as a 10-DOF multi-link. Similarly, the joint angles for the lower-limb are denoted as y_{11} to y_{20}.

2.3.2 The physical-learning model

Physical model for the bikebot and the rider trunk

We define the generalized coordinate for the trunk as $\boldsymbol{q}_t = [\varphi_h \; \theta \; \phi]^T$. Considering v_{rx} and ψ as time-varying model parameters, the bikebot's generalized coordinate is $\boldsymbol{q}_r = \varphi_b$. The bikebot is considered as a rigid body with mass m_b and mass moment of inertia J_b around the X-axis. The bicycle mass center G and the seat position are located at $[l_b \;\; 0 \;\; -h_b]^T$ and $[l_s \;\; 0 \;\; -h_s]^T$, respectively, in $\mathcal{R}$. The bicycle wheelbase is denoted as l and the caster angle as η. From [38], the bikebot yaw rate is calculated as

$$\dot{\psi} = \frac{v_{rx} \tan\gamma \, \mathrm{c}_\eta}{l \, \mathrm{c}_{\varphi_b}} = \frac{v_{rx} \, \mathrm{c}_\eta}{l \, \mathrm{c}_{\varphi_b}} u_s, \tag{2.8}$$

where $u_s = \tan\gamma$ is the steering control input and notation $\mathrm{c}_x := \cos x$ ($\mathrm{s}_x := \sin x$) is used for angle x in the above equation and throughout the rest of this chapter. From (2.8), it is straightforward to approximate and obtain $\ddot{\psi} \approx \frac{v_{rx} \, \mathrm{c}_\xi}{l} \dot{u}_s$.

Following the similar development in [36, 39], we take a constrained Lagrange equation to obtain the motion equation of the rider trunk and the bikebot in (2.1) with $\boldsymbol{q}_{rt} = [\varphi_b \; \varphi_h \; \theta]^T$, where matrices $\boldsymbol{M}(\boldsymbol{q}_{rt})$, $\boldsymbol{G}(\boldsymbol{q}_{rt})$, and $\boldsymbol{C}(\boldsymbol{q}_{rt}, \dot{\boldsymbol{q}}_{rt})$ are given in (2.9) with simplified trunk self-spinning dynamics $\dot{\phi} = 0$. The control input $\boldsymbol{u}_t$ is

$$\boldsymbol{u}_t = \begin{bmatrix} -\tau_h & \tau_h & \tau_\theta \end{bmatrix}^T,$$

where τ_h and τ_θ are the torques applied by the rider to the trunk in the roll and the pitch directions, respectively.

$$\boldsymbol{M}(\boldsymbol{q}_{rt}) = \begin{bmatrix} J_b + m_b h_b^2 + m_h h_s^2 & m_h h_h h_s \,\mathrm{c}_{\varphi_b - \varphi_h} \mathrm{s}_\theta & m_h h_h h_s \,\mathrm{s}_{\theta-\varphi_b} \mathrm{c}_\theta \\ m_h h_h h_s \,\mathrm{c}_{\varphi_b-\varphi_h} \mathrm{s}_\theta & J_h + m_h h_h^2 \mathrm{s}_\theta^2 & m_h h_h^2 \mathrm{s}_\theta \mathrm{c}_\theta \mathrm{s}_{\theta-\varphi_h} \\ m_h h_h h_s \,\mathrm{s}_{\theta-\varphi_b} \mathrm{c}_\theta & m_h h_h^2 \mathrm{s}_\theta \mathrm{c}_\theta \mathrm{s}_{\theta-\varphi_h} & J_h + m_h h_h^2 (1 - \mathrm{s}_\theta^2 \mathrm{s}_{\theta-\varphi_h}^2) \end{bmatrix},$$

$$\boldsymbol{C}(\boldsymbol{q}_{rt}, \dot{\boldsymbol{q}}_{rt}) = \begin{bmatrix} \ddot{\psi}\,\mathrm{c}_{\varphi_b}(m_b h_b l_b + m_h h_s l_s + m_h h_h h_s \mathrm{c}_\theta) + m_b h_b l_b \dot{\varphi}_b \dot{\psi}\,\mathrm{s}_{\varphi_b} + \\ m_h h_s h_h \dot{\varphi}_b (\dot{\theta}\,\mathrm{c}_\theta \mathrm{c}_{\theta-\varphi_b} + \dot{\varphi}_h \mathrm{s}_\theta \mathrm{s}_{\varphi_b-\varphi_h}) + m_b h_b \dot{\psi}\,\mathrm{c}_{\varphi_b}(v_{rx} - h_b \dot{\psi}\,\mathrm{s}_{\varphi_b}) - \\ m_h h_s \dot{\psi}\,\mathrm{c}_{\varphi_b}[\dot{\psi}(h_s \mathrm{s}_{\varphi_b} + h_h \mathrm{s}_\theta \mathrm{s}_{\varphi_h}) - v_{rx} + h_h \dot{\theta}\,\mathrm{s}_\theta \mathrm{c}_{\theta-\varphi_h}] + \\ m_h h_s \dot{\varphi}_b \dot{\psi}\,\mathrm{s}_{\varphi_b}(l_s + h_h \mathrm{c}_\theta) \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ m_h h_h \ddot{\psi}\,\mathrm{s}_\theta \mathrm{c}_{\varphi_h}(l_s/2 + h_h \mathrm{c}_\theta) - \\ m_h h_h \mathrm{s}_\theta(\dot{\psi}\,\mathrm{c}_{\varphi_h} + \dot{\theta}\,\mathrm{s}_{\theta-\varphi_h})[\dot{\psi}(h_s \mathrm{s}_{\varphi_b} + h_h \mathrm{s}_\theta \mathrm{s}_{\varphi_h}) - v_{rx} + h_h \dot{\theta}\,\mathrm{s}_\theta \mathrm{c}_{\theta-\varphi_h}] + \\ m_h h_h^2 \dot{\varphi}_h \dot{\theta}\,\mathrm{s}_\theta \mathrm{c}_\theta \mathrm{c}_{\theta-\varphi_h} + m_h h_h h_s \dot{\varphi}_h \dot{\varphi}_b \mathrm{s}_\theta \mathrm{s}_{\varphi_h-\varphi_b} + \\ m_h h_h \dot{\varphi}_h \dot{\psi}\,\mathrm{s}_\theta \mathrm{s}_{\varphi_h}(l_s + h_h \mathrm{c}_\theta) \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \ddot{\psi}[m_h h_h (h_s \mathrm{s}_{\varphi_b} + h_h \mathrm{s}_\theta \mathrm{s}_{\varphi_h}) \mathrm{s}_\theta \mathrm{c}_{\theta-\varphi_h} + m_h h_h \mathrm{c}_\theta \mathrm{s}_\theta (l_s + h_h \mathrm{c}_\theta)] - \\ m_h h_h \dot{v}_{rx} \mathrm{s}_\theta \mathrm{c}_{\theta-\varphi_h} + m_h h_h (\dot{\psi}\,\mathrm{s}_\theta - \dot{\theta}\,\mathrm{c}_{2\theta} - \dot{\varphi}_h \mathrm{c}_\theta \mathrm{c}_{\varphi_h})[\dot{\psi}(l_s + h_h \mathrm{c}_\theta) + \\ \dot{\varphi}_b h_s \mathrm{c}_{\varphi_b} + \dot{\theta} h_h \mathrm{c}_\theta \mathrm{s}_\theta + \dot{\varphi}_h h_h \mathrm{c}_{\varphi_h} \mathrm{s}_\theta] - \\ m_h h_h (\dot{\theta}\,\mathrm{c}_{2\theta-\varphi_h} + \dot{\psi}\,\mathrm{s}_{\varphi_h} \mathrm{c}_\theta)[\dot{\psi}(h_s \mathrm{s}_{\varphi_b} + h_h \mathrm{s}_\theta \mathrm{s}_{\varphi_h}) - \\ v_{rx} + \dot{\theta}(h_h \mathrm{s}_\theta^2 \mathrm{s}_{\varphi_h} + h_h \mathrm{c}_\theta \mathrm{c}_{\varphi_h} \mathrm{s}_\theta)] - \\ m_h h_h \mathrm{c}_\theta (2\dot{\theta}\,\mathrm{s}_\theta + \dot{\varphi}_h \mathrm{s}_{\varphi_h})(h_s \dot{\varphi}_b \mathrm{s}_{\varphi_b} - h_h \dot{\theta}\,\mathrm{c}_\theta^2 + h_h \dot{\varphi}_h \mathrm{s}_\theta \mathrm{s}_{\varphi_h}) \end{bmatrix},$$

$$\boldsymbol{G}(\boldsymbol{q}_{rt}) = -\begin{bmatrix} (m_b h_b + m_h h_s) g\,\mathrm{s}_{\varphi_b} \\ m_h g h_h \mathrm{s}_{\varphi_h} \\ 0 \end{bmatrix}. \tag{2.9}$$

Learning model for the upper-limb movement

We capture the 10-DOF upper-limb motion in a low-dimensional latent space. The rider's upper-limb orientation and motion during bikebot riding are primarily influenced by three motion templates of the bikebot-trunk system: trunk roll and pitch motions and the bikebot steering angle motion and therefore, $D = 10$ and $d = 3$. For the same reason, we build the three motion primitives by perturbing trunk roll (φ_h) and pitch (θ) motions and bikebot steering (γ) motion. For the input $\boldsymbol{u}_h$ in (2.2), we have

$$\boldsymbol{u}_h = \begin{bmatrix} \gamma(t) & \dot{\gamma}(t) & \varphi_h(t) & \dot{\varphi}_h(t) & \theta(t) & \dot{\theta}(t) & \phi(t) & \dot{\phi}(t) \end{bmatrix}^T.$$

The interactions between the physical and learning model lie in two aspects. First, the orientation angles of the rider trunk and the bikebot and their derivatives are taken as the input to the latent dynamics. The second interaction is from the geometric constraint and the human anatomical properties that will be presented in the next section.

2.4 EKF-based rider-bikebot pose estimation

In this section, we present an application of the physical-learning model for rider-bikebot pose estimation. Comparing with the previous studies in [1, 36], one main advantage of using the physical-learning model is the no need of wearable sensors attached to the limb segments. Both the bikebot and the rider trunk have one tri-axial gyroscope installed and similar to [36], we obtain the kinematic equations of the gyroscope measurements respectively as

$$\dot{\varphi}_b = \begin{bmatrix} \mathrm{c}_\alpha & 0 & s_\alpha \end{bmatrix} \boldsymbol{\omega}_r =: f_1(\boldsymbol{q}_{rt}; \boldsymbol{\omega}_r), \tag{2.10a}$$

$$\begin{aligned} \begin{bmatrix} \dot{\varphi}_h \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} &= \begin{bmatrix} 0 & \frac{\mathrm{s}_\phi}{\mathrm{s}_\theta} & \frac{\mathrm{c}_\phi}{\mathrm{s}_\theta} \\ 0 & \mathrm{c}_\phi & -\mathrm{s}_\phi \\ 1 & -\frac{\mathrm{c}_\theta}{\mathrm{s}_\theta}\mathrm{s}_\phi & -\frac{\mathrm{c}_\theta\,\mathrm{c}_\phi}{\mathrm{s}_\theta} \end{bmatrix} \boldsymbol{\omega}_t + \begin{bmatrix} \frac{\mathrm{s}_\alpha\,\mathrm{c}_\theta\,\mathrm{c}_{\varphi_h}}{\mathrm{c}_{\varphi_b} s_\theta} & 0 & -\frac{\mathrm{c}_\alpha\,\mathrm{c}_\theta\,\mathrm{c}_{\varphi_h}}{\mathrm{c}_{\varphi_b}\,\mathrm{s}_\theta} \\ \frac{\mathrm{s}_\alpha\,\mathrm{s}_{\varphi_h}}{\mathrm{c}_{\varphi_b}} & 0 & -\frac{\mathrm{c}_\alpha\,\mathrm{s}_{\varphi_h}}{\mathrm{c}_{\varphi_b}} \\ -\frac{\mathrm{s}_\alpha\,\mathrm{c}_{\varphi_h}}{\mathrm{c}_{\varphi_b}\,\mathrm{s}_\theta} & 0 & \frac{\mathrm{c}_\alpha\,\mathrm{c}_{\varphi_h}}{\mathrm{c}_{\varphi_b}\,\mathrm{s}_\theta} \end{bmatrix} \boldsymbol{\omega}_r \\ &=: \boldsymbol{f}_2(\boldsymbol{q}_{rt}; \boldsymbol{\omega}_r, \boldsymbol{\omega}_t). \end{aligned} \tag{2.10b}$$

where $\boldsymbol{\omega}_t$ and $\boldsymbol{\omega}_r$ are the trunk and bikebot gyroscope measurements, respectively. Due to the noise in gyroscope measurements, it is notoriously known that directly integration of (2.10a) and (2.10b) cannot produce acceptably accurate pose estimates. We need to look for other constraints to reduce or eliminate the integration drifts.

The latent model (2.2a) is considered as predictive dynamics for pose estimation. With estimated $\hat{\boldsymbol{x}}$, using prediction model (2.2b), the joint angle estimation is obtained. We use the dynamic and geometric constraints to fuse the information to enhance the estimation accuracy. An EKF design is used to fuse the measurements and the model. Figure 2.6 illustrates the information flow of the EKF design.

The dynamic constraint is obtained by adding the first two equations in (2.1) and (2.9)

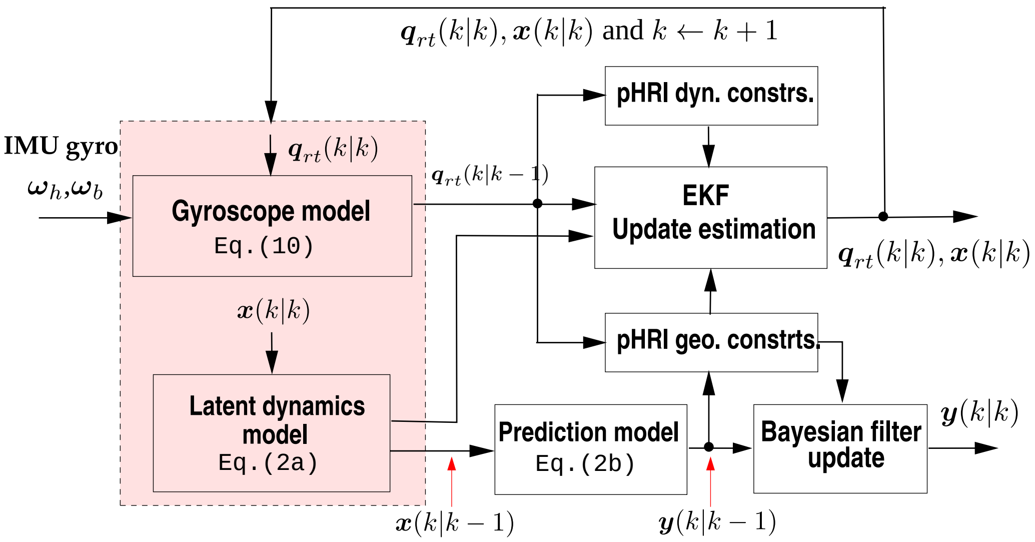


Figure 2.6: The structural and information flow diagram of the physical-learning model-based EKF design for pose estimation.

$$\begin{aligned}
z_1 \;=\;& (J_b + m_b h_b^2 + m_h h_s^2 + m_h h_h h_s \,\mathrm{c}_{\varphi_b - \varphi_h}\,\mathrm{s}_\theta)\ddot{\varphi}_b \\
& +(m_h h_h h_s \,\mathrm{c}_{\varphi_b-\varphi_h}\,\mathrm{s}_\theta + J_h + m_h h_h^2\,\mathrm{s}_\theta^2)\ddot{\varphi}_h + \\
& m_h h_h \,\mathrm{s}_{\theta-\varphi_b}(h_s\,\mathrm{c}_\theta + h_h\,\mathrm{s}_\theta\,\mathrm{c}_\theta)\ddot{\theta} + [(m_b h_b l_b + m_h h_s l_s + m_h h_h h_s\,\mathrm{c}_\theta)\,\mathrm{c}_{\varphi_b} + \\
& m_h h_h\,\mathrm{s}_\theta\,\mathrm{c}_{\varphi_h}(l_s/2 + h_h\,\mathrm{c}_\theta)]\ddot{\psi} + m_b h_b l_b\,\mathrm{s}_{\varphi_b}\,\dot{\varphi}_b\dot{\psi} + m_h h_s h_h\,\mathrm{c}_\theta\,\mathrm{c}_{\theta-\varphi_b}\,\dot{\varphi}_b\dot{\theta} + \\
& (m_b h_b\,\mathrm{c}_{\varphi_b} + m_h h_s\,\mathrm{c}_{\varphi_b} + m_h h_h\,\mathrm{s}_\theta\,\mathrm{c}_{\varphi_h})\dot{\psi} v_{rx} - (m_b h_b^2\,\mathrm{s}_{\varphi_b}\,\mathrm{c}_{\varphi_b} + m_h h_s^2\,\mathrm{s}_{\varphi_b}\,\mathrm{c}_{\varphi_b} + \\
& m_h h_s h_h\,\mathrm{c}_{\varphi_b}\,\mathrm{s}_\theta\,\mathrm{s}_{\varphi_h} + m_h h_h h_s\,\mathrm{s}_\theta\,\mathrm{s}_{\varphi_b}\,\mathrm{c}_{\varphi_h})\dot{\psi}^2 - m_h h_h\,\mathrm{s}_\theta[h_s\,\mathrm{c}_{\varphi_b}\,\mathrm{c}_{\theta-\varphi_h} + \\
& \mathrm{s}_{\theta-\varphi_h}(h_s\,\mathrm{s}_{\varphi_b} + h_h\,\mathrm{s}_{\varphi_h}) + h_h\,\mathrm{s}_\theta\,\mathrm{c}_{\varphi_h}\,\mathrm{c}_{\theta-\varphi_h}]\dot{\theta}\dot{\psi} + m_h h_s\,\mathrm{s}_{\varphi_b}(l_s + h_h\,\mathrm{c}_\theta)\dot{\varphi}_b\dot{\psi} + \\
& m_h h_h^2\,\mathrm{s}_\theta\,\mathrm{c}_\theta\,\mathrm{c}_{\theta-\varphi_h}\,\dot{\varphi}_h\dot{\theta} + m_h h_h\,\mathrm{s}_\theta\,\mathrm{s}_{\varphi_h}(l_s + h_h\,\mathrm{c}_\theta)\dot{\varphi}_h\dot{\psi} - (m_b h_b + m_h h_s)g\,\mathrm{s}_{\varphi_b} - \\
& m_h g h_h\,\mathrm{s}_{\varphi_h} = 0.
\end{aligned} \tag{2.11}$$

$$\begin{aligned}
\boldsymbol{z}_{2l} \;=\;& \boldsymbol{R}_x(\varphi_h)\boldsymbol{R}_y(\theta)\boldsymbol{R}_x(\phi)\begin{bmatrix} l_{sd} \\ h_{sd} \\ 0 \end{bmatrix} + \boldsymbol{R}_x(\varphi_h)\boldsymbol{R}_y(\theta)\boldsymbol{R}_x(\phi)\boldsymbol{R}_y(y_1)\boldsymbol{R}_z(y_2)\boldsymbol{R}_x(y_3)\begin{bmatrix} -l_u \\ 0 \\ 0 \end{bmatrix} \\
& +\boldsymbol{R}_x(\varphi_h)\boldsymbol{R}_y(\theta)\boldsymbol{R}_x(\phi)\boldsymbol{R}_y(y_1)\boldsymbol{R}_z(y_2)\boldsymbol{R}_x(y_3)\boldsymbol{R}_y(y_4)\boldsymbol{R}_x(y_5)\begin{bmatrix} -l_f \\ 0 \\ 0 \end{bmatrix} \\
& -\boldsymbol{r}_l(\gamma) = \boldsymbol{0}.
\end{aligned} \tag{2.12}$$

---

to eliminate τ_h as shown in (2.11). In this constraint, $\dot{\varphi}_h$, $\dot{\varphi}_b$, $\dot{\psi}$, $\dot{\theta}$ are obtained from (2.10a) and (2.10b), and $\ddot{\varphi}_h$, $\ddot{\psi}$ $\ddot{\theta}$ and $\ddot{\varphi}_b$ are approximated by numerically differentiation with smoothing actions. Calculation of constraint $z_1 = 0$ depends on variables φ_b, φ_h, and θ and we compute the output Jacobian matrix from (2.11).

The geometric constraints are from the human anatomical property and the rider-bikebot interactions. When riding the bikebot, the rider's hip sits on the seat and the hands always hold the handlebar. Therefore, the trunk, the upper-limb, and the bikebot frame and steering mechanism form a closed linkage structure. With the known upper-limb poses, we have one vector equation for the left upper-limb formulated as (2.12). In (2.12), l_{sd}, h_{sd}, l_u and l_f are the shoulder width and height, upper-arm and forearm lengths, respectively, $\boldsymbol{r}_l(\gamma)$ is the position vector from bikebot seat to the left handlebar position in $\mathcal{R}$ and $\boldsymbol{R}_i(\beta)$ represents the 3D rotational matrix around the i-axis with angle β, $i = x, y, z$. We obtain the similar constraint for the right upper-limb $\boldsymbol{z}_{2r} = \boldsymbol{0}$ and thus, we have constraints

$$\boldsymbol{z}_2(\varphi_h, \theta, \phi, \boldsymbol{y}) = \begin{bmatrix} \boldsymbol{z}_{2r} \\ \boldsymbol{z}_{2l} \end{bmatrix} = \boldsymbol{0} \in \mathbb{R}^6. \tag{2.13}$$

With the state dynamics (2.10a), (2.10b) and (2.2a) and observations (2.11) and (2.13), we define the state variable $\boldsymbol{\xi} = [\varphi_b \ \varphi_h \ \theta \ \phi \ \boldsymbol{x}^T]^T \in \mathbb{R}^7$ and the EKF design is illustrated in Algorithm 2.2. In the algorithm, matrices $\boldsymbol{Q}(k)$ and $\boldsymbol{R}(k)$ are the covariances of the prediction and observation errors at the kth step. Instead of using constant covariances, we follow the same treatment in [35] to update $\boldsymbol{Q}(k)$ and $\boldsymbol{R}(k)$ by (2.4). Particularly, since the last d dimensions of $\boldsymbol{\xi}$ are given by the latent variables, only a block matrix of $\boldsymbol{Q}(k)$ is updated as shown in line 3 in the algorithm. Because the observation equations are functions of $\boldsymbol{y}$, as shown in line 7, a Gaussian process prediction covariance $\boldsymbol{M}(k)$ of $\boldsymbol{y}(k)$ is obtained to calculate the gain $\boldsymbol{K}(k)$. We implement a Bayesian filter to obtain joint angle prediction $\hat{\boldsymbol{y}}$.

Remark 2.1. *The pose estimation for the lower-limb segments is obtained similar to these of the upper-limb. To include the lower-limb pose estimation, we augment the latent variable $\boldsymbol{x} \in \mathbb{R}^6$ with additional three elements x_4-x_6 and output variable $\boldsymbol{y} \in \mathbb{R}^{20}$ with y_{11}-y_{20}. The input to the lower-limb latent dynamics is the pedal crank angle*

Algorithm 2.2: EKF implementation

```
Input: $\boldsymbol{q}_{rt}(0|0)$, $\boldsymbol{x}(0|0)$, $\boldsymbol{\omega}_b$ and $\boldsymbol{\omega}_h$
Output: Estimates $\boldsymbol{q}_{rt}(k)$, $\boldsymbol{x}(k)$, and $\boldsymbol{y}(k)$
Initialize variance matrices $\boldsymbol{Q}(0)$, $\boldsymbol{R}(0)$, $\Sigma(0)$;
while $k \leq N$ do
    Update $\boldsymbol{\xi}(k|k-1) = [\boldsymbol{q}_{rt}^T \ \boldsymbol{x}^T]^T$ by (2.10) and (2.2a);
    $\boldsymbol{Q}(k)_{[5:7,5:7]} = \mathtt{GP}_{\Sigma}(\boldsymbol{s}_{t-1}, \boldsymbol{T}_p)$;
    $\boldsymbol{F} = [\boldsymbol{f}^T \ f_1 \ \boldsymbol{f}_2^T]^T$; $\boldsymbol{G}(k) = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{\xi}}\big|_{\boldsymbol{\xi}(k-1)}$ with (2.5);
    $\Sigma(k|k-1) = \boldsymbol{G}(k)\Sigma(k-1)\boldsymbol{G}^T(k) + \boldsymbol{Q}(k)$;
    Update $\hat{\boldsymbol{z}}(k)$ with $\boldsymbol{\xi}(k|k-1)$, $\boldsymbol{y}(k-1)$ by (2.11)-(2.13) ;
    $\boldsymbol{M}(k) = \mathtt{GP}_{\Sigma}(\boldsymbol{x}(k|k-1), \boldsymbol{T}_o)$ ;
    $\boldsymbol{H}(k) = \frac{\partial \boldsymbol{Z}}{\partial \boldsymbol{\xi}}\big|_{\boldsymbol{\xi}(k|k-1)}$; $\boldsymbol{N}(k) = \frac{\partial \boldsymbol{Z}}{\partial \boldsymbol{y}}\big|_{\boldsymbol{\xi}(k|k-1)}$;
    $\boldsymbol{K}(k) = \Sigma(k|k-1)\boldsymbol{H}^T(k)\big[\boldsymbol{H}(k)\Sigma(k|k-1)\boldsymbol{H}^T(k) +$
      $\boldsymbol{N}(k)\boldsymbol{M}(k)\boldsymbol{N}^T(k) + \boldsymbol{R}(k)\big]^{-1}$;
    $\boldsymbol{\xi}(k) = \boldsymbol{\xi}(k|k-1) + \boldsymbol{K}(k)(\boldsymbol{0} - \hat{\boldsymbol{z}}(k))$;
    $\Sigma(k) = (\boldsymbol{I} - \boldsymbol{K}(k)\boldsymbol{H}(k))\Sigma(k|k-1)$;
    $\boldsymbol{y}$(k)=BayesianFilter ($\boldsymbol{\xi}(k), \Sigma(k)$,$\hat{\boldsymbol{z}}(k)$);
end
function $\boldsymbol{y}$=BayesianFilter ($\boldsymbol{\xi}$,$\Sigma$,$\hat{\boldsymbol{z}}$) ;
$\hat{\boldsymbol{y}} = \boldsymbol{W}(\boldsymbol{\xi})$; $\boldsymbol{G}_w = \frac{\partial \boldsymbol{W}}{\partial \boldsymbol{\xi}}\big|_{\boldsymbol{\xi}}$ with (2.7); $\boldsymbol{Q}_y = \mathtt{GP}_{\mu}(\boldsymbol{x}, \boldsymbol{T}_o)$;
$\hat{\Sigma}_w = \boldsymbol{G}_w\Sigma\boldsymbol{G}_w^T + \boldsymbol{Q}_y$; $\boldsymbol{H}_w = \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{y}}$; $\boldsymbol{K} = \hat{\Sigma}_w\boldsymbol{H}_w^T(\boldsymbol{H}_w\hat{\Sigma}_w\boldsymbol{H}_w^T + \boldsymbol{R})^{-1}$;
$\boldsymbol{y} = \hat{\boldsymbol{y}} + \boldsymbol{K}(\boldsymbol{0} - \hat{\boldsymbol{z}})$;
```

and angular rate. The geometric constraint for the lower-limb is constructed as the linkage formed by the lower-limb, the seat and the pedal [37]. We here omit the detailed discussions.

2.5 Experiments

2.5.1 Experimental setup

Five healthy and experienced bicycle riders (four male and one female with ages: 27 ± 3 years old, heights: 176 ± 5 cm, and weights: 68 ± 10 kg) were recruited to conduct both the indoor and outdoor experiments. The duration for each riding experiment run was around 2 minutes. The subjects were first asked to get familiar with the bikebot riding before actual experimental data were taken. All subjects signed their informed consent using a protocol approved by the Institutional Review Board (IRB) at Rutgers University. Before experiments, all subjects' biomechanic parameters and the wearable sensor locations are measured. In experiments, we used three load cells mounted under the seat to roughly measure the trunk mass. The heights of the center of the mass position are assumed to be proportional to trunk length and their values are also estimated through OpenSim, a musculoskeletal modeling and dynamic simulation software package [1]. The model parameters (limb length, trunk mass, etc.) for each human subject are used in his/her pose estimation designs.

For the bikebot system, steering angle γ and velocity v_{rx} are measured by encoders. One IMU (model 605 from Motion Sense Inc.) is mounted on bikebot and another one on the human trunk. We only use the tri-axial gyroscope measurements from these IMUs. A vision-based motion capture system (8 Bonita cameras from Vicon Inc.) is used to provide ground truth in indoor experiments. For outdoor experiments, a camera is mounted on the bikebot handlebar and a set of gyroscope/marker pairs are mounted on the upper- and lower-limb to provide the ground truth through vision-inertial fusion algorithms. A real-time embedded system (CompactRIO 9082 from National Instruments Inc.) is use to sample and process all sensor measurements at the frequency of 100 Hz. The gyro-balancer actuation is used only in the outdoor experiments to generate perturbation torques to excite the rider's responses.

2.5.2 Experimental results

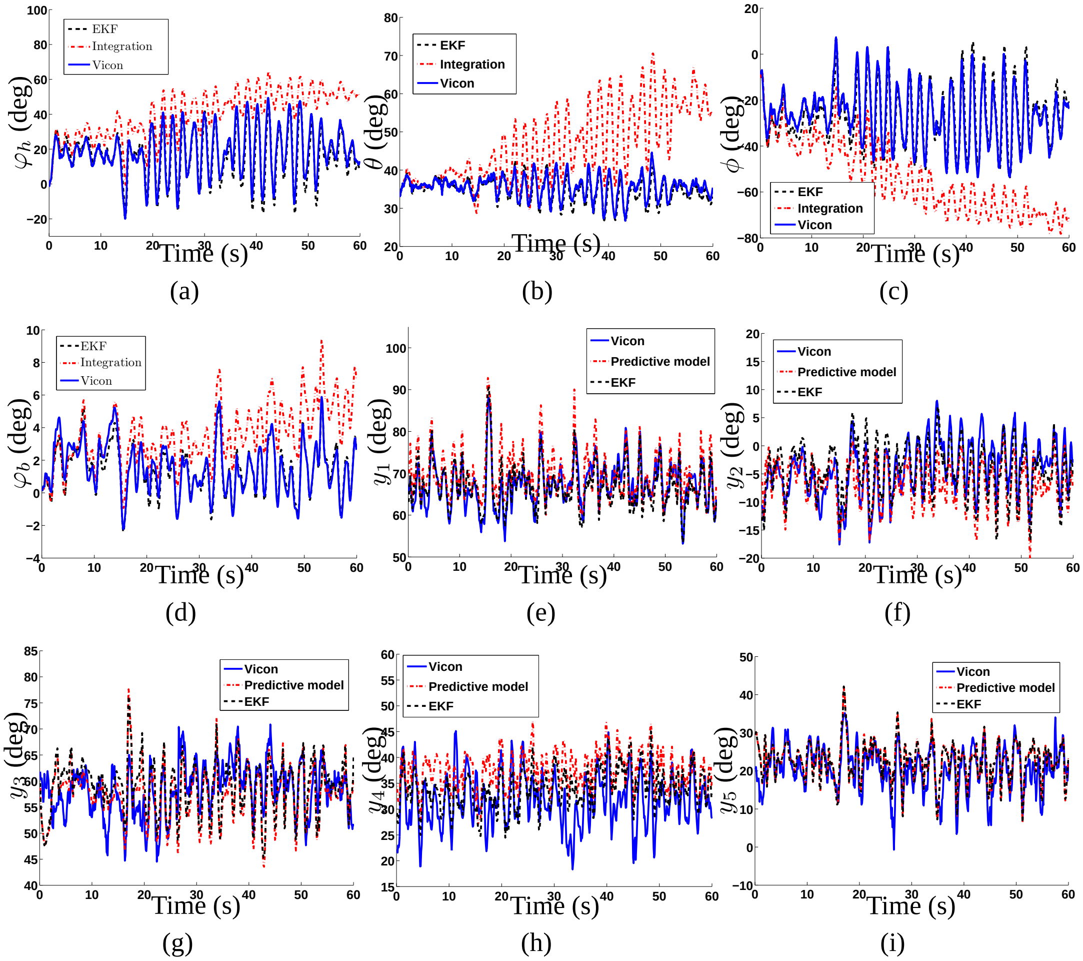
We conduct indoor bikebot riding experiments to demonstrate the performance of the EKF-based pose estimation scheme. To construct isolated latent space coordinate axes in the ALE algorithm, we used the human movement data sets collected by riding

[1] http://opensim.stanford.edu.

a stationary bikebot and assume that the riders use the similar upper-limb and trunk movements and strategies when riding the moving and stationary bikebots. We fixed the bikebot roll angles at a set of values and the subjects were asked to perturb their trunk motions in the roll and pitch directions and move their steering angles independently in separate experiment runs. We collected the motion data in these experiments to construct the ALE reduction axes. By doing so, we avoid challenges to perturb the subjects and excite the isolated body segment motions while riding and balancing a moving bikebot. The joint-angle prediction performance presented later in this section confirms the feasibility to use this method to construct the ALE axes.

We collect camera-based ground truth measurements of two rounds of bikebot riding experiment to obtain the learning model and the physical model parameters. The training data set contains a total of 400 pairs of input and output points. For testing and validation, we use the separate experiments to compute the EKF estimation and compare them with the ground truth. Figure 2.7 shows a one-minute pose estimation of the bikebot and the upper-limb. We chose a group of five joint angles (y_1-y_5) for the left upper-limb as a representative group of all 20 limb joint angles. The right upper-limb is similar and the lower-limb results are more regular than those of the upper-limbs due to the periodic pedaling motion. The EKF estimates follow closely with the ground truth for the bikebot and human body segments orientations. Figure 2.8 further shows the estimation errors of y_1, y_2 and y_4 by the EKF scheme and by only the machine-learning-based prediction model. Clearly, the use of the integrated physical-learning model improves the estimation performance.

To demonstrate the performance for all ten upper-limb joint angles, Table 2.1 illustrates the estimation errors obtained by EKF scheme with three model reduction approaches, including ALE, PCA and LLE. Each element in the table is the mean value with one standard deviation of the root mean square (RMS) errors obtained by 6 rounds of upper-limb motion over a 20-second period. The input data to the model are



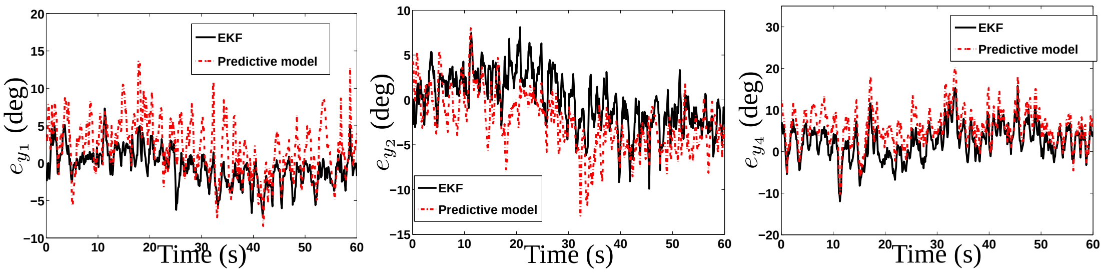
(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 2.7: Experimental comparison results among the physical-learning model-based EKF, learning predictive model, IMU direct integration and the ground truth by Vicon system. (a) φ_h. (b) θ. (c) ϕ. (d) φ_b. (e) y_1. (f) y_2. (g) y_3. (h) y_4. (i) y_5.

obtained from vision-based motion capture measurements. For most of the upper-limb joint angles (except y_3), the mean values of the RMS errors by the ALE are smaller than the corresponding results by the PCA method. The sum of the RMS errors of all the upper-limb joint angles (i.e., y_1-y_{10}) by the ALE algorithm (sum RMS: 36.6) is also less than those of the other two approaches (sum RMS: PCA: 40.9 and LLE: 57.2). From the statistical significance viewpoint, the t-test of the mean value of the RMS errors over the ten joint angles confirms significant smaller errors under the ALE than those under the PCA predictions ($p = 0.01$) although the t-test of the standard



(a) (b) (c)

Figure 2.8: Prediction errors for upper-limb poses. (a) y_1. (b) y_2. (c) y_4.

deviation of the RMS errors does not show significantly different between these two methods ($p = 0.43$). For comparison purpose, we also list the estimation performance reported in [1] by using the wearable IMUs attached to each of the upper-limb segment in the same experiment. The performance by the ALE-GPDM model-based prediction is comparable to the results by using the wearable sensors. The performance of the estimation results for outdoor is similar to these of the indoor experiments and we omit the details.

Table 2.1: Root mean square (RMS) errors (in deg) of the EKF-based estimated trunk-bikebot-trunk poses angles over a 20-second period (in-door experiments) and comparison with wearable IMU results from [1].

GPDM	φ_h	θ	ϕ	φ_b	y_1	y_2	y_3
ALE	5.0 ± 1.4	1.4 ± 0.5	5.1 ± 0.3	0.5 ± 0.1	2.2 ± 0.3	2.8 ± 0.1	4.9 ± 1.1
PCA	5.2 ± 1.9	1.5 ± 0.5	5.1 ± 0.6	0.5 ± 0.1	2.3 ± 0.4	2.9 ± 0.5	4.6 ± 0.5
LLE	5.4 ± 0.1	1.5 ± 0.3	5.8 ± 1.2	0.5 ± 0.2	3.3 ± 1.2	3.3 ± 0.8	6.1 ± 1.8
IMU	3.2	4.6	3.5	0.7	4.9	4.7	3.0
GPDM	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}
ALE	4.5 ± 0.4	4.2 ± 0.8	1.9 ± 0.7	3.9 ± 0.3	4.1 ± 0.5	5.0 ± 1.2	3.1 ± 0.3
PCA	5.0 ± 0.2	5.3 ± 0.9	2.4 ± 0.8	4.5 ± 0.9	4.7 ± 0.5	5.1 ± 0.8	4.1 ± 0.4
LLE	9.1 ± 0.3	7.3 ± 0.5	2.8 ± 0.6	5.6 ± 2.5	8.5 ± 3.1	6.2 ± 1.3	5.0 ± 1.5
IMU	4.7	4.8	4.7	5.2	4.3	4.8	5.2

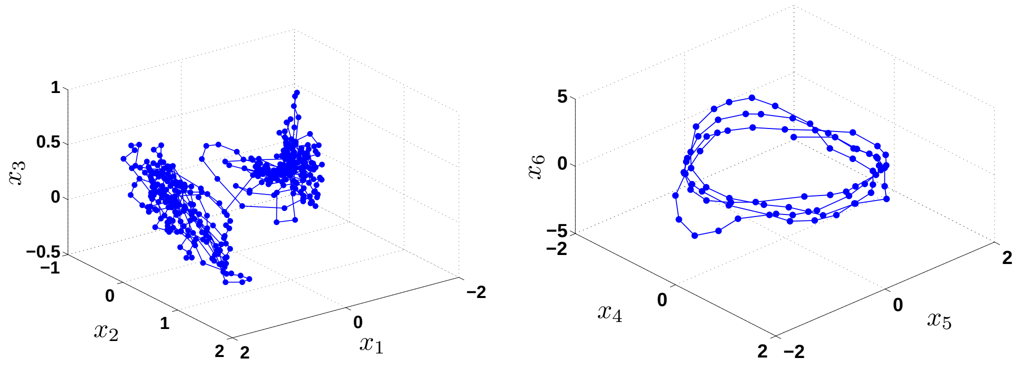
Figure 2.9 shows the latent dynamics trajectory in the latent space. We plot both the latent dynamics for the upper-limb motion (Figure 2.9(a)) and the lower-limb motion

Table 2.2: RMS errors (in deg) of the estimated upper-limb angles over a 25-second perturbed period in the outdoor experiments.

φ_h	θ	ϕ	φ_b	y_1	y_2	y_3
3.7 ± 1.1	2.6 ± 0.7	5.1 ± 1.4	0.3 ± 0.1	3.7 ± 1.6	4.5 ± 1.9	6.3 ± 1.1

y_4	y_5	y_6	y_7	y_8	y_9	y_{10}
7.2 ± 2.0	4.5 ± 0.8	4.2 ± 1.2	4.9 ± 2.0	6.2 ± 2.1	6.0 ± 1.7	4.4 ± 1.0

(Figure 2.9(b)) during the indoor experiments. In these plots, we used a constant value $\sigma_{\hat{\boldsymbol{x}}} = 0.1$ for the preset observation noice variance. Because of the similar anatomical structures, like the upper-limb, a three-dimensional ($d = 3$) latent space is chosen for the lower-limb pedaling motion. Due to the periodic pedaling movement, the trajectory of the latent dynamics (x_4-x_6) of the lower-limb is periodic, while the upper-limb trajectory in the latent space (x_1-x_3) is non-periodic. Therefore, the pose estimation scheme using the learning model can predict both the periodic and non-periodic human movements. The learning model such as GPDM has also been used and reported to capture other non-periodic human motion such as dancing and playing golf, etc. [14].



(a) (b)

Figure 2.9: Trajectory of the latent dynamics for indoor riding experiments. (a) Upper-limb motion. (b) Lower-limb motion.

The rider-bikebot interaction is through a set of geometric constraints and these interaction constraints are shared by different human subjects. Built on this observation, we use the latent dynamic model built from one subject to estimate the joint angles of the other subjects during bikebot riding experiments. Tables 2.3 and 2.4 list the mean and one-standard deviation of the RMS errors of the estimated joint angles for the upper-limb (y_1 to y_{10}) and the lower-limb (y_{11} to y_{20}), respectively, for five human subjects in indoor and outdoor experiments. From these results, we clearly see that the latent dynamic model captures the rider-bikebot interactions and predicts other riders' riding motion although the model is built on one rider.

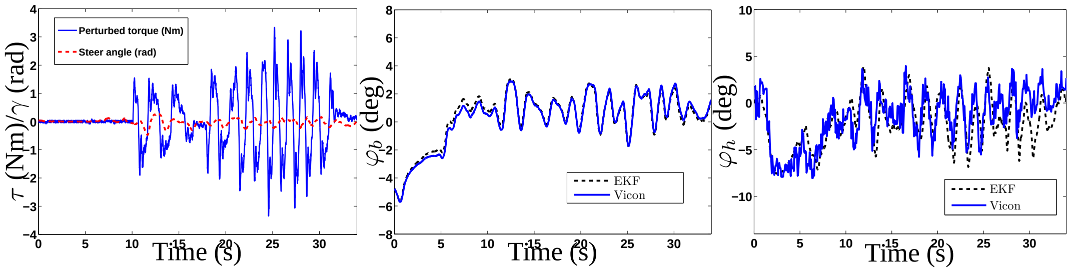
Table 2.3: RMS errors (in deg) of the estimated upper-limb angles over a 1-minute period in the indoor and outdoor experiments.

	φ_h	θ	ϕ	φ_b	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}
Indoor	6.4 ± 1.8	1.5 ± 0.2	7.6 ± 3.1	0.6 ± 0.2	2.6 ± 0.3	4.2 ± 0.9	5.9 ± 0.9	5.3 ± 0.8	5.1 ± 0.7	2.8 ± 0.5	4.4 ± 1.2	3.9 ± 0.2	5.6 ± 0.1	3.0 ± 0.4
Outdoor	7.7 ± 1.0	2.0 ± 0.7	9.6 ± 0.4	0.8 ± 0.2	2.9 ± 0.3	4.9 ± 0.7	6.4 ± 0.9	6.0 ± 2.0	5.2 ± 0.5	2.9 ± 0.4	4.7 ± 0.8	4.5 ± 1.3	6.3 ± 1.1	3.1 ± 0.5

Table 2.4: RMS errors (in deg) of the estimated lower-limb joint angles over a 1-minute period in the indoor and outdoor experiments.

	y_{11}	y_{12}	y_{13}	y_{14}	y_{15}	y_{16}	y_{17}	y_{18}	y_{19}	y_{20}
Indoor	2.8 ± 0.1	4.4 ± 1.0	3.2 ± 0.2	2.8 ± 0.5	3.1 ± 1.8	2.4 ± 0.4	5.3 ± 0.7	3.8 ± 0.4	2.6 ± 0.7	2.8 ± 0.3
Outdoor	3.2 ± 0.8	4.8 ± 1.4	3.7 ± 1.0	3.2 ± 0.8	4.2 ± 2.5	3.6 ± 1.8	5.8 ± 0.9	4.6 ± 1.5	3.4 ± 1.0	3.1 ± 0.7

We further demonstrate the estimation performance for bikebot riding under perturbation. In experiments, the human rider was not informed when and how the perturbation torques would be generated and applied by the gyro-balancer. Under the perturbed external torque, the rider apply recovering strategies (e.g., trunk movements and steering) to keep the bikebot stable. Figure 2.10 shows the experimental results of the applied perturbation torque and rider's reactions. The perturbation starts around 10 seconds and clearly the rider uses steering to react and keep balancing the platform. Figure 2.10(a) shows the perturbed external torque by the gyro-balancer and the rider's reacted steering angle, Figure 2.10(b) shows the estimated and actual bikebot roll angles, while Figure 2.10(c) shows the estimated and actual rider's trunk roll angles. The estimated bikebot and human roll angles follow the true values closely. The RMS errors of the estimated upper-limb joint angles are listed in Table 2.2 and the estimation accuracy is similar to that in Table 2.3 obtained in normal riding experiments without perturbation. These results confirm that the estimation scheme predicts pose estimation under perturbed, dynamic riding experiments.



(a) (b) (c)

Figure 2.10: The prediction results under the perturbed bikebot riding experiments. (a) The generated perturbation torque and the steering angle. (b) Bikebot roll angles. (c) Rider trunk roll angles.

Table 2.5: RMS errors (in deg) of the estimated upper-limb angles over a 1-minute period (indoor experiments) by the GP model and the geometric and dynamic constraints only.

	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}
GP	2.8 ± 1.4	3.9 ± 1.7	5.3 ± 1.7	6.8 ± 3.0	4.4 ± 1.6	2.9 ± 0.7	2.7 ± 1.0	4.7 ± 1.4	6.2 ± 3.2	3.2 ± 0.8
Constraint	11.7 ± 2.5	6.3 ± 2.7	13.3 ± 1.6	10.0 ± 1.9	9.3 ± 1.4	12.9 ± 3.2	7.0 ± 1.6	12.9 ± 4.6	8.3 ± 3.3	6.9 ± 3.3

2.5.3 Discussions

From Figure 2.7, we notice that the upper-limb motion during bikebot riding are restrictive within a range of 30 degrees. These restrictions are coming from the fact that the human hands are fixed on the steering handlebar while the other ends of the upper-limb are connected to the human trunk. With this observation, it might be possible

Table 2.6: RMS errors (in deg) of the estimated upper-limb angles by various models for bicycle riding with elastic strings.

	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}
GPDM/ALE	11.1 ± 3.1	3.7 ± 0.3	14.3 ± 7.1	8.5 ± 2.3	4.7 ± 0.9	7.4 ± 0.7	4.3 ± 1.0	7.5 ± 1.4	9.5 ± 0.6	6.8 ± 1.4
GPDM/PCA	15.2 ± 6.3	5.9 ± 1.5	17.8 ± 7.0	8.2 ± 1.5	3.1 ± 0.6	13.8 ± 1.4	6.7 ± 3.4	14.6 ± 2.3	11.7 ± 1.8	6.2 ± 0.5
GP	17.1 ± 5.6	8.2 ± 2.7	44.3 ± 32.0	33.3 ± 8.6	10.9 ± 3.7	22.0 ± 10.9	10.4 ± 2.9	11.5 ± 3.0	31.0 ± 6.9	24.3 ± 15.2
Constraint	12.8 ± 3.3	38.7 ± 2.0	22.6 ± 5.4	32.0 ± 1.6	8.1 ± 2.5	15.2 ± 2.5	13.5 ± 0.9	34.2 ± 7.3	34.9 ± 3.0	11.6 ± 2.3

to obtain the upper-limb poses through a simple Gaussian process (GP) model to map from the trunk and steering angle data set, or even only using geometric constraints $\boldsymbol{z}_2$ and dynamic constraints z_1. We compare the prediction results of the upper-limb poses by using the GPDM and a simple GP model. Table 2.5 shows the comparison results for the upper-limb joint angles estimation under the GP model and the only constraint-based approach. Comparing with the similar indoor prediction results by the GPDM shown in Table 2.3, it is clear that the GPDM and the GP approaches result in similar performance (sums of RMS errors are equally 42.8.) Indeed, the t-test of the mean value of the RMS errors over the ten joint angles shows no significant difference between the GPDM and the GP model predictions ($p = 0.49$). From Table 2.5, we also observe that only with the geometric and dynamic constraints, it is difficult to accurately obtain the pose estimation (i.e., large estimation errors) due to limb motion redundancy.

We conduct additional experiments to further demonstrate the application of the GPDM with comparison of the GP model to human motion with greater variance and less constraints. Figures. 2.11(a) and 2.11(b) illustrate the new riding experiments that were conducted to extend the upper-limb motion with large flexibility. In the new experiments, instead firmly grasping the steering handlebar, the subject was asked to ride and steer the bicycle with hands holding a pair of elastic strings fixed on the handlebar; see Figure. 2.11(b). The subject moved his limb arbitrarily for multiple riding experiments runs inside a laboratory; see Figure. 2.11(a).

With this setup, rider's hands can flexibly move along the xx' and zz' directions during riding as shown in Figure. 2.11(b). This gives greater variances and less constraints than that in regular riding style. The less constrained hands positions can be observed by the dynamically changing handlebar steering angle γ and the actual hand steering angle γ' calculated by the marker positions on the human hands. Figure. 2.12(a) shows the differences between γ and γ' and these differences generate the large variance in upper-limb joint angles.

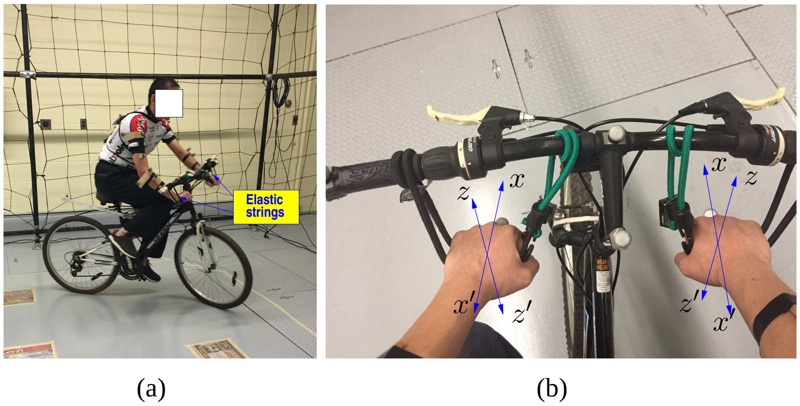


Figure 2.11: (a) Riding a bicycle with two hands on elastic strings tied on the steering handlebar. (b) A close view of the hands on the elastic strings with flexibility along the xx' and zz' directions.

We compare the prediction performance of the GPDM and the GP model trained with data sets from the new experiments. Same as in the previously studies, we constructed all the GPDM and the GP models by using the bicycle steering angle and human trunk angles as the inputs for the new experiments. The relative positions of human hands to the handlebar are not used to predict the joint angles. The geometry constraints are assumed to be the same as in the previous experiments but with larger constraint uncertainties.

Table 2.6 lists the comparison results of the joint angle predictions under these two models for multiple runs. For the GPDM, we list two reduction methods (i.e., ALE and PCA). For comparison purposes, we also compute and list the prediction results by only enforcing the geometric and dynamic constraints in Table 2.6. When we use the

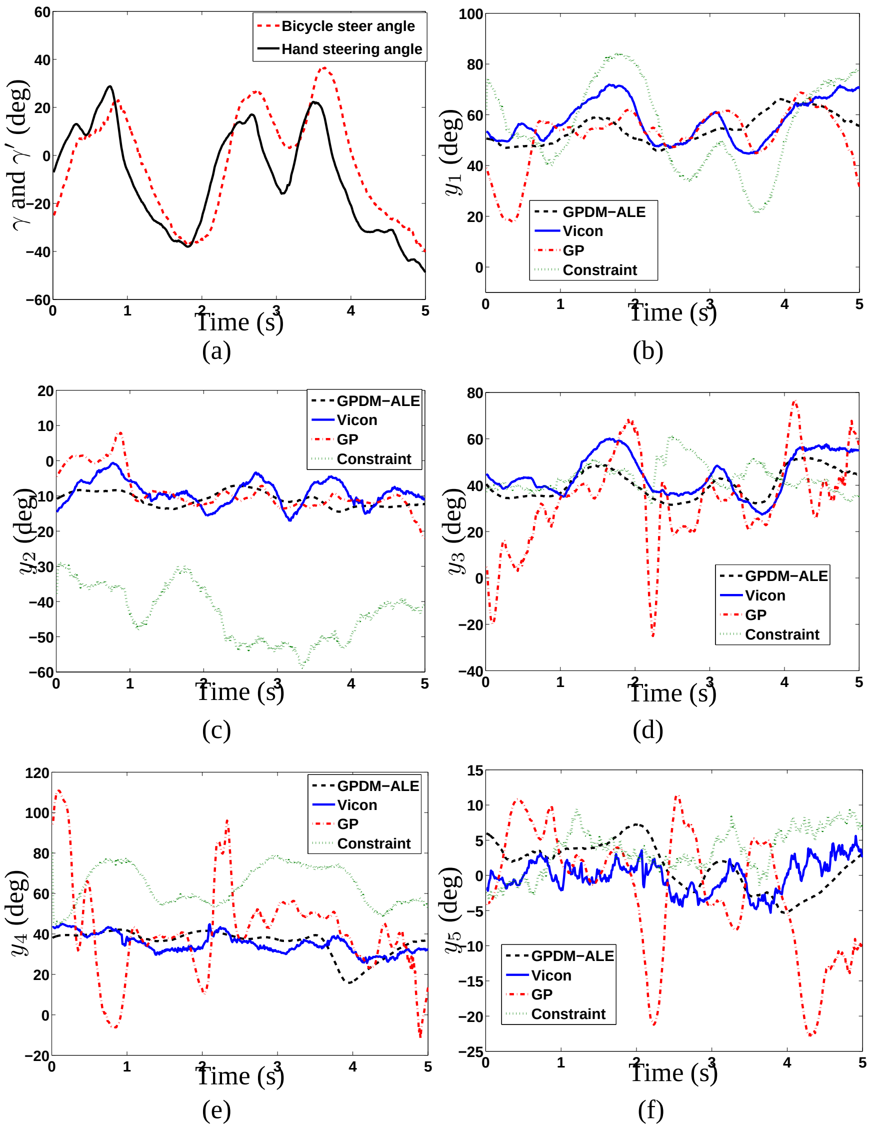


Figure 2.12: Upper-limb pose estimation comparison among various modeling approaches for new experiments. (a) Handlebar steering angle γ and human hands steering angle γ'. (b) y_1. (c) y_2. (d) y_3. (e) y_4. (f) y_5.

geometric and dynamic constraints in the EKF design, we treat the joint angle predictive models as constants with additive Gaussian noises. The use of the ALE reduction in GPDM generates more accurate results in almost all joint angles than those under the PCA reduction method (sums of RMS errors are 77.8 and 103.2, respectively). The t-test results of the mean value and standard deviation of the RMS errors over the ten joint angles show significantly smaller under the GPDM-ALE than those under the GPDM-PCA predictions ($p = 0.01$ and $p = 0.05$, respectively). We also see the significantly superior performance of the GPDM ALE approach than those with the GP model (sums of RMS errors are 77.8 and 213, respectively). The GP model and the only constraint-based approach cannot accurately predict most of the upper-limb joint angles. As one example, Figures. 2.12(b)-2.12(f) shows the joint angle estimations y_1-y_5 of one subject riding experiment under the GPDM-ALE, the GP model and the only constraint-based approach. We clearly see that the GP model and the only constraint-based scheme cannot capture the joint motion accurately in these experiments, particularly for joint angles y_3-y_5.

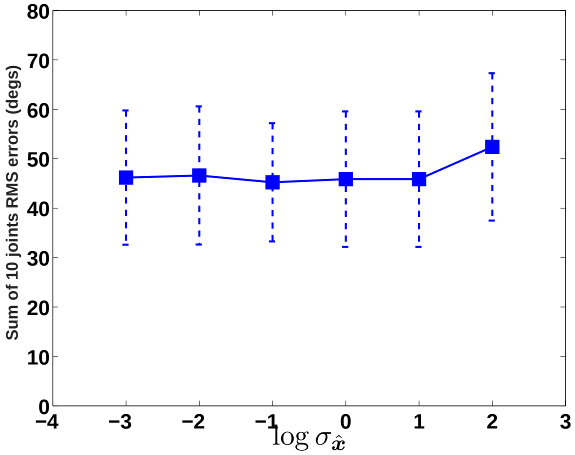


Figure 2.13: Prediction error of the model trained with different latent space observation noise uncertainty levels $\sigma_{\hat{\boldsymbol{x}}}$

During the latent variable initialization process, we chose the preset observation noise value $\sigma_{\hat{\boldsymbol{x}}} = 0.1$ of the variance of $P(\boldsymbol{X}|\hat{\boldsymbol{X}})$ during training. To illustrate the influence of the value $\sigma_{\hat{\boldsymbol{x}}}$ on the estimation results, Figure 2.13 shows the RMS errors of the ten predicted joint angles of the upper-limb poses (mean and one standard deviation) using different values of $\sigma_{\hat{\boldsymbol{x}}}$ during the training process. In the plot, the prediction is calculated using five different runs of the upper-limb motion during bikebot riding. From the results in Figure. 2.13, we clearly see that the prediction performance is almost kept at a constant level until the $\sigma_{\hat{\boldsymbol{x}}}$ values increase to 100. Figure 2.14 further illustrates the comparison of the trajectories of the latent space variables (x_1, x_2, x_3)

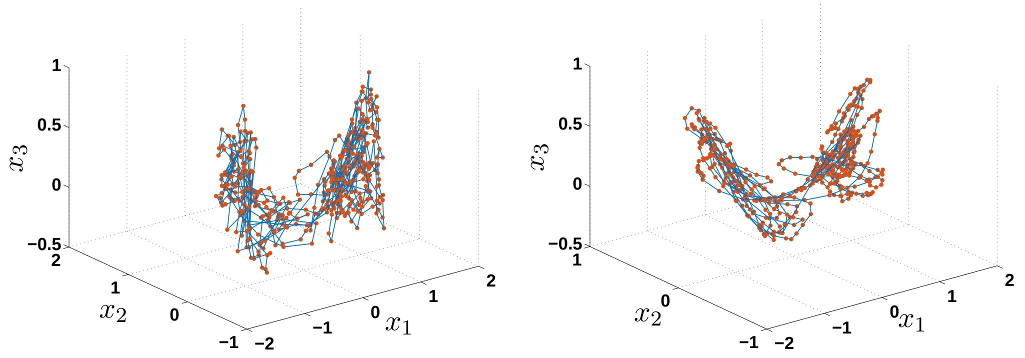


Figure 2.14: Trajectories of the latent space variables (x_1, x_2, x_3) for the upper-limb motion using different values of $\sigma_{\hat{\boldsymbol{x}}}$. (a) Directly from the ALE algorithm. (b) With a labeling uncertainty $\sigma_{\hat{\boldsymbol{x}}} = 100$.

for the upper-limb motion initially obtained ALE (Figure 2.14(a)) and with labeling uncertainty at $\sigma_{\hat{\boldsymbol{x}}} = 0.1$ (Figure 2.9(a)) and $\sigma_{\hat{\boldsymbol{x}}} = 100$ (Figure 2.14(b)). These plots clearly show that the larger values of $\sigma_{\hat{\boldsymbol{x}}}$ result smoother latent variable trajectories. The learning model with large values of $\sigma_{\hat{\boldsymbol{x}}}$ tends to possibly generate worse prediction results as shown in Figure 2.13 for $\sigma_{\hat{\boldsymbol{x}}} = 100$. Noting that the values of the latent space variables (x_1, x_2, x_3) generated by the ALE algorithm are all within $(-2,\ 2)$, we therefore chose $\sigma_{\hat{\boldsymbol{x}}} = 0.1$ in our implementation.

The EKF estimation convergence of φ_h and φ_b partially comes from constraint $z_1 = 0$ in (2.11) obtained from the systems dynamics. To demonstrate the impact of the dynamic and geometric constraints on the states estimation, we relax either of two constraints and compare the estimation results with these under the full constraints. Figure 2.15 shows the mean and standard deviation of the pose estimation with and without constraints (2.11) or (2.13). We observe that the dynamic constraint (2.11) mainly influences the accuracy of φ_b estimation and the geometric constraint (2.13) mainly affects the estimation accuracy of the trunk and the upper-limb joint angles. This observation can be explained by the sensitivity calculation of these constraints with respect to each state variable. Moreover, the observability matrix of the EKF state dynamics with the geometric constraint always has a full rank (we omit the calculation

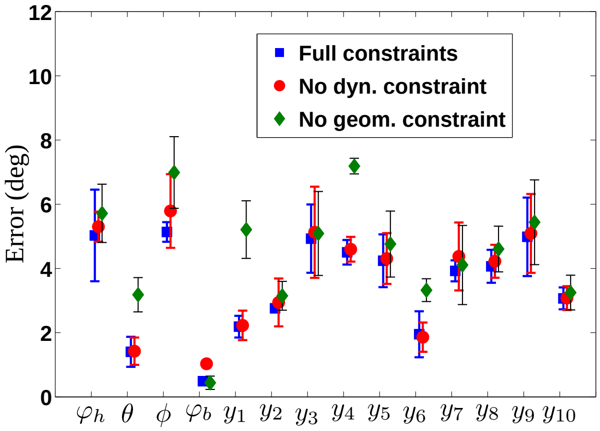
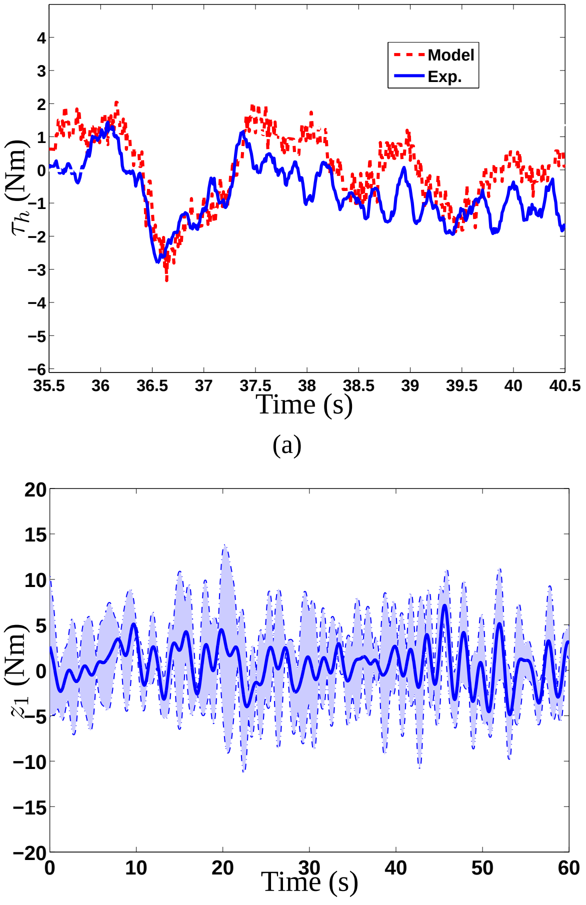


Figure 2.15: The pose estimation errors with and without constraints.

here) and therefore, the EKF design satisfies the necessary convergence condition.

To validate the bikebot-trunk dynamics, Figure 2.16(a) shows the comparison of the values of torque τ_h calculated by (2.9) with these measured by the seat force sensor in a straight-line riding experiment. The measured torque values match the model predictions. One attractive property to use constraint $z_1 = 0$ in (2.11) instead of dynamic equations (2.9) is the no need to use force or torque measurements. To validate constraint (2.11), Figure 2.16(b) plots the means and one standard deviations of the values of z_1 over one-minute riding experiments by five human subjects. Clearly, the values of z_1 are around zero and the variations can be captured through the noise term in the EKF design.

To understand the influence of the size of the training data set on the estimation performance, Figure 2.17 shows the sum of root mean square of the estimation errors of all ten upper-limb joints over the number of training data points. We notice that the estimation performance is consistent after the numbers of the used training data points are greater than a threshold (e.g., around 200 data points). Moreover, the proposed EKF design is not sensitive to the choice of the initial value of state variables. For example, we use two different means to initialize the state variables: one is to use average value for each variable to initialize each variable and the other is to use accurate



(a)

(b)

Figure 2.16: (a) Rider upper-body torque τ_h model comparison from a straight-line riding experiment. (b) Dynamic constraint $z_1 = 0$ variation (mean and one standard deviation) over five-subject experiments.

measurements. The resulting EKF estimation performance (RMS errors) are listed in Table 2.7 for a one-minute riding experiment. No significant performance differences are observed between these two cases.

Table 2.7: RMS errors (in deg) of each estimated angle over a 1-minute experiment run with different initial values.

Init. val.	φ_h	θ	ϕ	φ_b	y_1	y_2	y_3	y_4	y_5	y_6	y_7	y_8	y_9	y_{10}
Approx.	5.3	1.7	5.3	0.5	2.4	3.6	5.1	5.3	4.4	2.0	4.0	4.2	5.1	3.0
Measure	5.0	1.5	4.9	0.4	2.3	3.0	5.1	4.6	4.2	2.0	3.7	4.2	5.1	3.0

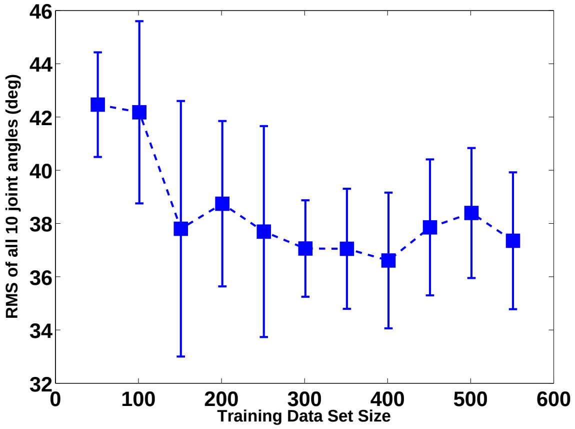


Figure 2.17: The RMS of all estimated ten upper-limb joint angles vs. the training data points.

2.6 Summary

We presented an integrated physical-machine learning modeling approach for physical human-robot interactions. In this modeling framework, physical models were used to capture the robot and human trunk dynamics while a machine-learning-based model was used to describe the highly redundant, high-DOF limb motion on low-dimensional latent space. Coupled interconnections and constraints were built between these two models. For dimensionality reduction, we presented a novel axial linear embedding (ALE) method. The main advantage of the proposed modeling approach was to represent the pHRI on the low-dimensional manifold and to enable the use of the existing estimation and control techniques effectively applied to complex pHRI. The new ALE reduction algorithm also preserved the physical meaning of the latent variables of the learning model. We illustrated and demonstrated the modeling framework through bikebot riding experiments with application to estimate the human and bikebot poses. We also compared the limb pose estimation performance of the proposed GPDM-ALE and the simple GP model. The comparison results demonstrated that the GPDM-ALE outperformed the GP model for physical human-bikebot interactions with large variance and less constraints. The attractive properties of using the proposed model is that even without attached any wearable sensors on the limb segments, the model-based estimation scheme predicted the limb motion accurately.

Chapter 3

Hybrid Zero Dynamics of Biped Walking with Foot Slip

3.1 Introduction

Foot slip is one of the major causes for human falls and injuries and these foot slip-induced falls cause enormous economic and societal costs [40]. To develop effective fall prevention technologies, it is critical to understand human locomotion and balance recovery under foot slip. Modeling and control of human walking locomotion are necessarily effective approaches to design new wearable assistive devices.

Several modeling approaches have been proposed in the past for human walking locomotion with foot slip. Simulation-based dynamic model is used in [41] to study motion stability of slip and fall. Robotic bipedal models [11, 42] are recently used to study human walking motion. For example, multi-link biped models are presented in [43, 44] to generate the human walking gaits with point foot-ground contact. In [44], curved foot is considered with a robotic bipedal model to demonstrate a controlled energy-efficient robotic walker. In these studies, the human bipedal model is given in a hybrid dynamics framework where foot impact generates the discrete event while single-stance or double-stance gait motions are in continuous time domain. The repetitive human walking gait is indeed captured by the hybrid zero dynamics (HZD) when the gaits follow the desired profiles. Most existing bipedal models are built on an assumption of no slip foot-ground contact and cannot be used for studying human gaits with foot slip.

In recent work [2], a multi-link robotic bipedal model is presented for human walking with foot slip. The model extends the bipedal framework in [11,43,44] by relaxing the foot sticking to ground condition and also equipping with circular rolling foot. The work in [2] however neither discusses in details about the resulting HZD under foot slip nor presents how to use this model for slip recovery control design. On the other hand, the work in [45] propose a slip recovery control design using a two-mass linear inverted pendulum (LIP) without considering whole-body dynamics.

For multi-link bipedal robotic walkers, the HZD-based control design has been implemented on many platforms [44,46–49]. However, all of these work is built on the assumption of no foot slip and the HZD includes only the dynamics of the progressing variable θ and its derivative. When foot slips, the resulting HZD contains additional state variable (i.e., slipping distance and velocity) and the dynamics are much more complicated than non-slip case [2]. One of main goals of this chapter is to present this HZD model and its properties. Moreover, we also use the HZD model to study the stability condition and design for slip recovery process.

Several control methods are used to design the gait recovery strategies under foot slip. In [2], a feedback linearization controller is applied to the multi-link whole-body robotic model to track the human gait profiles that are obtained from experiments. The balance recovery strategy in [45] uses the LIP model to design controllers of the center of pressure (COP) position for each gait phase to maintain the desired motion for human center of mass (COM).

In this chapter, we mainly focus on the HZD for human walking gaits under foot slip. The HZD is built on the multi-link robotic bipedal model for human walking under foot slip in [2]. We explicitly calcuate and present the HZD that consists of dynamics of the gait progression variable and the slipping distance. We then discuss the HZD stability conditions and properties under a set of slip recovery gait process that are obtained from human subject experiments. The main contribution of the work lies in the new HZD model and its application to human walking under foot slip. The results in this chapter complements the existing literature on human bipedal models and gait control. One interesting application of this chapter is to apply the HZD on highly skilled human locomotion such as figure skating in which a stable, agile locomotion is achieved by actively taking both the foot slipping and walking gaits.

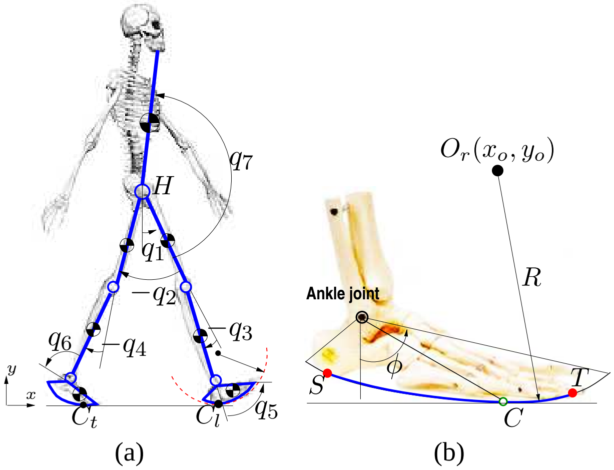
The remainder of this chapter is organized as follows. In Section 3.2, we review the robotic bipedal models for human walking without and with foot slip. Section 3.3 presents the HZD for human walking with foot slip and slip recovery process. We present experimental results in Section 3.4. Finally, the concluding remark and future work directions are discussed in Section 3.5.

3.2 Bipedal walking models and hybrid zero dynamics

3.2.1 System configuration

Figure. 3.1(a) shows the setup for the human bipedal model. The human gait is considered only in the sagittal plane. The human body is considered as a 7-link rigid body. The head, arm and torso (HAT) is considered as one link that is connected to the left- and right-thigh. Six active joints include two hip joints, two knee joints and two ankle joints. Similar to [44], let q_i, $i = 2, \dots, 7$, denote relative angles between links and q_1 denote absolute angle of stance thigh with respect to the vertical direction.

The joitn angle vector is defined as $\boldsymbol{q}_a = [q_1 \ \cdots \ q_7]^T$. A circular foot is consider with a rolling radius R on the solid ground; see Figure. 4.1(b). To capture the foot slip motion, we denote the position of the circle center O_r of the stance foot as $[x_o, y_o]^T$. The foot slip distance is $\boldsymbol{q}_s = [x_s, y_s]^T = [x_o + R(\phi - \phi_0), y_o - R]^T$, where ϕ is the absolute rolling angle of the stance foot and ϕ_0 is the value of ϕ at the initial time of that stance. $\dot{\boldsymbol{q}}_s = [\dot{x}_o + R\dot{\phi}, \dot{y}_o]^T$ is the slipping velocity. If the stance foot is purely rolling on the ground, $\boldsymbol{q}_s = \boldsymbol{0}$. We define generalized coordinate $\boldsymbol{q}_e = [\boldsymbol{q}_a^T, \boldsymbol{q}_s^T]^T$ and its velocity $\dot{\boldsymbol{q}}_e = [\dot{\boldsymbol{q}}_a^T, \dot{\boldsymbol{q}}_s^T]^T$. Denoting the velocity of the lowest point on the swing foot as $\dot{\boldsymbol{p}}_2$, then $\dot{\boldsymbol{p}}_2 = [\dot{x}_{o_2} + R\dot{\phi}_2, \dot{y}_{o_2}]^T$, where $[x_{o_2}, y_{o_2}]^T$ is the center of the swing foot circle and ϕ_2 is the absolute angle between the swing foot and the vertical direction.



(a)

(b)

Figure 3.1: (a) Schematic of the 7-link human walking model with curved foot contact. (b) Schematic of the foot-contact model.

One human walking cycle consists of a single-stance phase, a double-stance phase and a foot impact phase [11]. Since it contains majority of the stance duration, only single-stance phase and foot impact dynamics are considered in this study. This simplification also helps to highlight the hybrid zero dynamics (HZD) of slip recovery process in later discussion. Figure. 3.2 captures the basic characteristics of the discrete-continuous dynamics.

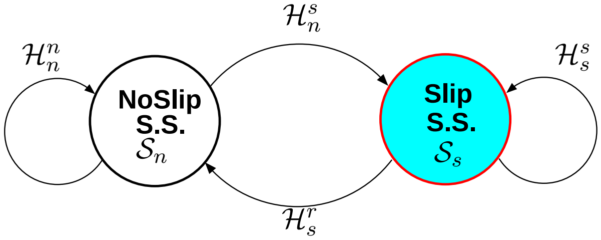


Figure 3.2: Finite state diagram of human walking gait with foot slip.

3.2.2 Biped dynamics without foot slip

Dynamics model

The non-slip single-stance dynamics is described as [11]

$$\boldsymbol{\Sigma}_n: \quad \boldsymbol{D}_n(\boldsymbol{q}_a)\ddot{q}_a + \boldsymbol{C}_n(\boldsymbol{q}_a, \dot{\boldsymbol{q}}_a)\dot{\boldsymbol{q}}_a + \boldsymbol{G}_n(\boldsymbol{q}_a) = \boldsymbol{B}_n\boldsymbol{u}, \tag{3.1}$$

where $\boldsymbol{D}_n(\boldsymbol{q}_a)$, $\boldsymbol{C}_n(\boldsymbol{q}_a, \dot{\boldsymbol{q}}_a)$, $\boldsymbol{G}_n(\boldsymbol{q}_a)$ and $\boldsymbol{B}_n$ are the inertia, Coriolis, gravity and input mapping matrices, respectively. The system is underactuation and absolute joint angle q_1 is not controlled by any joint torque. Therefore we have $\boldsymbol{B}_n = [\boldsymbol{0}_{N-1}\ \boldsymbol{I}_{N-1}]^T$, where $\boldsymbol{0}_n = [0 \cdots 0]^T \in \mathbb{R}^n$ is a zero column vector. A feedback linearization approach is adopted to control the joint angles $\boldsymbol{q}_a$ to follow a desired trajectory that is specified by a progression variable $\theta = \boldsymbol{c}\boldsymbol{q}_a$, where $\boldsymbol{c}$ is a constant progression vector. The feedback linearization controller enforces the virtual constraint

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{q}_a) = \boldsymbol{H}_0\boldsymbol{q}_a - \boldsymbol{h}_d(\theta) = \boldsymbol{0}, \tag{3.2}$$

where $\boldsymbol{H}_0 \in \mathbb{R}^{6\times 7}$ and function $\boldsymbol{h}_d : \mathbb{R} \mapsto \mathbb{R}^6$. At the end of single-stance phase, the swinging foot impacts the ground and becomes the new stance leg in the next single-stance phase.

The impact dynamics is obtained by integrating the double stance dynamics equation

$$\boldsymbol{\Sigma}_e: \quad \boldsymbol{D}_e(\boldsymbol{q}_e)\ddot{q}_e + \boldsymbol{C}_e(\boldsymbol{q}_e, \dot{\boldsymbol{q}}_e)\dot{\boldsymbol{q}}_e + \boldsymbol{G}_e(\boldsymbol{q}_e) = \boldsymbol{B}_e\boldsymbol{u} + \boldsymbol{E}_2^T(\boldsymbol{q}_e)\delta\boldsymbol{F}_2, \tag{3.3}$$

over the instantaneous impact time, where $\boldsymbol{E}_2(\boldsymbol{q}_e) = \frac{\partial \dot{\boldsymbol{p}}_2}{\partial \dot{\boldsymbol{q}}_e}(\boldsymbol{q}_e)$ is the Jacobian matrix of impacting foot contact point velocity $\dot{\boldsymbol{p}}_2$ with respect to $\dot{\boldsymbol{q}}_e$, $\delta\boldsymbol{F}_2 = [\delta F_{2x}, \delta F_{2y}]^T$ is the force applied on the impacting foot.

In [11], the integration of the Coriolis, gravity and input torques terms are neglected resulting in the impact dynamics as

$$\boldsymbol{D}_e(\boldsymbol{q}_e^-)\dot{\boldsymbol{q}}_e^+ - \boldsymbol{D}_e(\boldsymbol{q}_e^-)\dot{\boldsymbol{q}}_e^- = \boldsymbol{E}_2^T(\boldsymbol{q}_e^-)\boldsymbol{F}_2 \tag{3.4}$$

where $\boldsymbol{F}_2 = \int_{t^-}^{t^+} \delta \boldsymbol{F}_2(\tau) d\tau$.

Even though this model is accurate for periodic normal walking gait, it is not valid for the impact from slip gait to recover gait demonstrated in the experimental data. For a more general nonslip impact model, we do not neglect the integration of the input torques and obtain

$$\mathcal{H}_n^n : \quad \boldsymbol{D}_e(\boldsymbol{q}_e^-)\dot{\boldsymbol{q}}_e^+ - \boldsymbol{D}_e(\boldsymbol{q}_e^-)\dot{\boldsymbol{q}}_e^- = \boldsymbol{E}_2^T(\boldsymbol{q}_e^-)\boldsymbol{F}_2 + M \tag{3.5}$$

,where $M = \int_{t^-}^{t^+} \boldsymbol{B}_e \boldsymbol{u}(\tau) d\tau$ is a constant determined from experimental data. If the foot ground friction is big enough so that the impacting foot sticks to the ground after the impact, the velocity constraint $\boldsymbol{E}_2(\boldsymbol{q}_e^-)\dot{\boldsymbol{q}}_e^- = \boldsymbol{0}$ satisfies. Combining these two equations, the full nonslip impact dynamics $\mathcal{H}_n^n$ is

$$\mathcal{H}_n^n : \quad \begin{bmatrix} \boldsymbol{D}_e(\boldsymbol{q}_e^-) & -\boldsymbol{E}_2^T \\ \boldsymbol{E}_2 & \boldsymbol{0}_2 \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_e^+ \\ \boldsymbol{F}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{D}_e(\boldsymbol{q}_e^-) \\ \boldsymbol{0}_2 \end{bmatrix} \dot{\boldsymbol{q}}_e^- + \begin{bmatrix} M \\ \boldsymbol{0}_2 \end{bmatrix} . \tag{3.6}$$

where $E_2 = \frac{\partial \dot{\boldsymbol{p}}_2}{\partial \dot{\boldsymbol{q}}_a}(\boldsymbol{q}_a)$ is the Jacobian matrix of impacting foot contact point velocity $\dot{\boldsymbol{p}}_2$ with respect to $\dot{\boldsymbol{q}}_a$, $\boldsymbol{F}_2 = [F_{2x}\, F_{2y}]^T$ is the impulsive force applied on the impacting foot. The matrix on the left-hand side of (3.6) is invertible and after considering the relabeling of stance foot, we have

$$\begin{aligned} \dot{\boldsymbol{q}}_a^+ &= SI_{\boldsymbol{q}_a} \begin{bmatrix} \boldsymbol{D}_e(\boldsymbol{q}_e^-) & -\boldsymbol{E}_2^T \\ \boldsymbol{E}_2 & \boldsymbol{0}_2 \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{D}_e(\boldsymbol{q}_e^-) \\ \boldsymbol{0}_2 \end{bmatrix} \dot{\boldsymbol{q}}_e^- + SI_{\boldsymbol{q}_a} \begin{bmatrix} \boldsymbol{D}_e(\boldsymbol{q}_e^-) & -\boldsymbol{E}_2^T \\ \boldsymbol{E}_2 & \boldsymbol{0}_2 \end{bmatrix}^{-1} \begin{bmatrix} M \\ \boldsymbol{0}_2 \end{bmatrix} \\ &= \boldsymbol{\Delta}_n(\boldsymbol{q}_e^-)\dot{\boldsymbol{q}}_e^- + \boldsymbol{b}_n, \end{aligned} \tag{3.7}$$

where S is the relabel matrix, $I_{\boldsymbol{q}_a}$ picks $\dot{\boldsymbol{q}}_a^+$ from $[\dot{\boldsymbol{q}}_e^{+,T}, \boldsymbol{F}_2^T]^T$. Because the new stance foot sticks to the ground, $\dot{\boldsymbol{q}}_e^+ = [(\dot{\boldsymbol{q}}_a^+)^T\, \boldsymbol{0}_2]^T$ and if the previous step before impact is nonslip, we also have $\dot{\boldsymbol{q}}_e^- = [(\dot{\boldsymbol{q}}_a^-)^T\, \boldsymbol{0}_2]^T$. More detailed information about the dynamic model can be find in [11, 44].

Zero dynamics

The non-slip swing phase zero dynamics is obtained by enforcing the states of (3.1) onto the virtual constraint (3.2). Following a similar treatment in [44], we define $\boldsymbol{x}_a = [\boldsymbol{q}_a^T\ \dot{\boldsymbol{q}}_a^T]^T$ and re-write (3.1) into a first order equation

$$\dot{\boldsymbol{x}}_a = \boldsymbol{f}(\boldsymbol{x}_a) + \boldsymbol{g}(\boldsymbol{x}_a)\boldsymbol{u}, \tag{3.8}$$

where

$$\boldsymbol{f}(\boldsymbol{x}_a) = \begin{bmatrix} \dot{\boldsymbol{q}}_a \\ \\ -\boldsymbol{D}_n^{-1}(\boldsymbol{C}_n\dot{\boldsymbol{q}}_a + \boldsymbol{G}_n) \end{bmatrix},\ \boldsymbol{g}(\boldsymbol{x}_a) = \begin{bmatrix} \boldsymbol{0} \\ \\ \boldsymbol{D}_n^{-1}\boldsymbol{B}_n \end{bmatrix}.$$

By feedback linearization, we consider a coordinate transformation

$$\boldsymbol{\eta}_1 = \boldsymbol{h}(\boldsymbol{q}_a), \boldsymbol{\eta}_2 = L_{\boldsymbol{f}}\boldsymbol{h}(\boldsymbol{q}_a), \xi_1 = \theta(\boldsymbol{q}_a), \xi_2 = \boldsymbol{D}_a(\boldsymbol{q}_a)\dot{\boldsymbol{q}}_a, \tag{3.9}$$

where $L_{\boldsymbol{f}}\boldsymbol{h}(\boldsymbol{q}_a)$ is the Lie derivative of $\boldsymbol{h}(\boldsymbol{q}_a)$ along $\boldsymbol{f}$, and $\boldsymbol{D}_a(\boldsymbol{q}) \in \mathbb{R}^{1\times 7}$ is formed by the row of $\boldsymbol{D}_n(\boldsymbol{q})$ that corresponds to the unactuated joint angle q_1. If we choose $\boldsymbol{u} = (L_{\boldsymbol{g}}L_{\boldsymbol{f}}\boldsymbol{h})^{-1}(-L_{\boldsymbol{f}}^2\boldsymbol{h} + \boldsymbol{v})$ and $\boldsymbol{v}$ is chosen to regulate $\boldsymbol{\eta}_1 = \boldsymbol{\eta}_2 = \boldsymbol{0}$ exponentially, then output dynamics becomes

$$\dot{\boldsymbol{\eta}}_1 = \boldsymbol{\eta}_2,\ \dot{\boldsymbol{\eta}}_2 = \boldsymbol{v}. \tag{3.10}$$

The zero dynamics is given as

$$\boldsymbol{\Sigma}_{ZD}^n : \begin{cases} \dot{\xi}_1 = \frac{\partial\theta}{\partial\boldsymbol{q}_a}\dot{\boldsymbol{q}}_a =: k_1(\xi_1)\xi_2, \\ \\ \dot{\xi}_2 = \dot{\boldsymbol{q}}_a^T\frac{\partial D_a^T}{\partial\boldsymbol{q}_a}\dot{\boldsymbol{q}}_a - \boldsymbol{C}_a\dot{\boldsymbol{q}}_a - \boldsymbol{G}_a =: k_2(\xi_1, \xi_2). \end{cases} \tag{3.11}$$

The transformation of output states $\boldsymbol{\eta} := [\boldsymbol{\eta}_1^T\ \boldsymbol{\eta}_2^T]^T$ and internal states $\boldsymbol{\xi} := [\xi_1\ \xi_2]^T$ to joint angle states $\boldsymbol{x}_a$ is obtained as

$$\begin{bmatrix} \boldsymbol{\eta}_1 \\ \\ \xi_1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{h}(\boldsymbol{q}_a) \\ \\ \theta(\boldsymbol{q}_a) \end{bmatrix} =: \boldsymbol{\Phi}(\boldsymbol{q}_a),\ \begin{bmatrix} \boldsymbol{\eta}_2 \\ \\ \xi_2 \end{bmatrix} = \begin{bmatrix} \frac{\partial\boldsymbol{h}(\boldsymbol{q}_a)}{\partial\boldsymbol{q}_a} \\ \\ \boldsymbol{D}_a(\boldsymbol{q}_a) \end{bmatrix}\dot{\boldsymbol{q}}_a \tag{3.12}$$

and the inverse transformation is

$$\boldsymbol{q}_a = \boldsymbol{\Phi}^{-1}\left(\begin{bmatrix}\boldsymbol{\eta}_1\\ \xi_1\end{bmatrix}\right), \; \dot{\boldsymbol{q}}_a = \begin{bmatrix}\frac{\partial \boldsymbol{h}(\boldsymbol{q}_a)}{\partial \boldsymbol{q}_a}\\ \boldsymbol{D}_a(\boldsymbol{q}_a)\end{bmatrix}^{-1}\begin{bmatrix}\boldsymbol{\eta}_2\\ \xi_2\end{bmatrix}. \tag{3.13}$$

When $\boldsymbol{\eta} = \boldsymbol{0}$, $\boldsymbol{x}_a$ is a function of only $\boldsymbol{\xi}$ and the right-hand side of (3.11) can be written as function of $\boldsymbol{\xi}$.

Stable HZD of periodic walking

Getting onto the swing phase virtual constraint does not guarantee the stable periodic walking gait. To guarantee the stable periodic walking, The pre-impact swing phase states should be mapped to swing phase zero dynamics space again under the impact mapping (3.7). Mathematically, it is expressed as [11]

$$\boldsymbol{\Delta}_n(\mathcal{S} \cap \mathcal{Z}_{\alpha_n}) \subset \mathcal{Z}_{\alpha_n}, \tag{3.14}$$

where $\mathcal{S}$ is the configuration space where both feet are contacting the ground and $\mathcal{Z}_{\alpha_n}$ is the swing phase zero dynamics space. Assuming the pre-impact zero dynamics state is $\boldsymbol{\xi}^-$, the pre-impact full state is $\boldsymbol{x}_a^-$ by applying (3.13) with $\boldsymbol{\eta} = \boldsymbol{0}$. The post-impact state is obtained $\boldsymbol{x}_a^+$ by (3.7). The hybrid invariant set across impact reqires that the after-impact state is still on the zero dynamic space, namely,

$$\boldsymbol{\eta}_1 = \boldsymbol{h}(\boldsymbol{q}_a^+) = \boldsymbol{0}, \; \boldsymbol{\eta}_2 = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{q}_a}(\boldsymbol{q}_a^+)\dot{\boldsymbol{q}}_a^+ = \boldsymbol{0}.$$

3.2.3 Biped dynamics with foot slip

The equations of motion for biped with foot slip is obtained as [2]

$$\boldsymbol{D}_s\ddot{\boldsymbol{q}}_e + \boldsymbol{C}_s\dot{\boldsymbol{q}}_e + \boldsymbol{G}_s = \begin{bmatrix}\boldsymbol{B}_s\boldsymbol{u}\\ \boldsymbol{F}_s\end{bmatrix}, \tag{3.15}$$

where $\boldsymbol{D}_s$, $\boldsymbol{C}_s$, $\boldsymbol{G}_s$ and $\boldsymbol{B}_s$ are the inertia, Coriolis, gravity and input mapping matrices, respectively, and $\boldsymbol{F}_s = [F_x \; F_n]^T$ is the foot contact force. With several constraints [2], (3.15) is simplified by defining $\boldsymbol{q}_{es} = [\boldsymbol{q}_a^T \, x_s]^T$

$$\boldsymbol{\Sigma}_s : \; \boldsymbol{D}_s^s \ddot{\boldsymbol{q}}_{es} + \boldsymbol{C}_s^s \dot{\boldsymbol{q}}_{es} + \boldsymbol{G}_s^s = \boldsymbol{B}_s^s \boldsymbol{u}, \tag{3.16}$$

where $\boldsymbol{D}_s^s$, $\boldsymbol{C}_s^s$, $\boldsymbol{G}_s^s$ and $\boldsymbol{B}_s^s$ are detailed in [2].

System (3.16) is underactuated and indeed the absolute joint angle q_1 and the slipping distance x_s are two underactuated variables. We will discuss the HZD for (3.16) in the next section. The impact model under foot slip is obtained from the extension of non-slip case. Assuming that at the instant after the impact the touching point slips with $\dot{\boldsymbol{p}}_2^+ = [\dot{x}_s^+ \; 0]^T$, from $\dot{\boldsymbol{p}}_2 = \boldsymbol{E}_2 \dot{\boldsymbol{q}}_e$, we obtain

$$\mathcal{H}_n^s : \begin{bmatrix} \boldsymbol{D}_e(\boldsymbol{q}_e^-) & -\boldsymbol{E}_2^T \\ \boldsymbol{E}_2 & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_e^+ \\ \boldsymbol{F}_2^T \end{bmatrix} = \begin{bmatrix} \boldsymbol{D}_e(\boldsymbol{q}_e^-)\dot{\boldsymbol{q}}_e^- \\ \dot{x}_s^+ \\ 0 \end{bmatrix}. \tag{3.17}$$

Compared with (3.6), one more unknown $\dot{x}_s^+$ is introduced. We here use the friction coefficient to relate impulses $F_{2x} = -\mu F_{2y}$ because of the friction model and the integration over instantaneous impact time. Considering relabeling, we have

$$\dot{\boldsymbol{q}}_e^+ = \begin{bmatrix} \dot{\boldsymbol{q}}_a^+ \\ \dot{x}_s^+ \\ 0 \end{bmatrix} = \boldsymbol{\Delta}_s(\boldsymbol{q}_e^-)\dot{q}_e^-, \tag{3.18}$$

where $\boldsymbol{\Delta}_s$ denotes the foot-slip impact mapping matrix.

3.3 Hybrid zero dynamics of slip recovery

In this section, we first present the hybrid zero dynamics for bipedal walking with foot slip. Then, we discuss a set of slip recovery phases to regain the walking balance. The recovery phase sequence is observed in human experiment. Finally, we introduce the stability of slip recovery sequence.

3.3.1 Zero dynamics of slip swing phase

The slip swing phase dynamics (3.16) has two degrees of underactuation, i.e., the absolute joint angle q_1 and the slip distance x_s. The 6-dimension virtual constraint for slip swing phase is

$$\boldsymbol{y} = \boldsymbol{h}_s(\boldsymbol{q}_a) = \boldsymbol{0}. \tag{3.19}$$

Similar to the non-slip case, defining $\boldsymbol{x}_{es} = [\boldsymbol{q}_{es}^T, \dot{\boldsymbol{q}}_{es}^T]^T$, (3.16) can be written as

$$\dot{\boldsymbol{x}}_{es} = \boldsymbol{f}_s(\boldsymbol{x}_{es}) + \boldsymbol{g}_s(\boldsymbol{x}_{es})\boldsymbol{u} \tag{3.20}$$

where

$$\boldsymbol{f}_s(\boldsymbol{x}_{es}) = \begin{bmatrix} \dot{\boldsymbol{q}}_{es} \\ -(\boldsymbol{D}_s^s)^{-1}(\boldsymbol{C}_s^s \dot{\boldsymbol{q}}_s + \boldsymbol{G}_s^s) \end{bmatrix}, \ \boldsymbol{g}_s(\boldsymbol{x}_{es}) = \begin{bmatrix} \boldsymbol{0} \\ (\boldsymbol{D}_s^s)^{-1}\boldsymbol{B}_s^s \end{bmatrix}.$$

Defining the state transformation $\boldsymbol{\eta}_s = [\eta_{1s}^T, \eta_{2s}^T]^T = [h_s(q_a), L_{\boldsymbol{f}_s}\boldsymbol{h}_s]^T$, we use the feedback linearization to obtain the output dynamics $\dot{\boldsymbol{\eta}}_{1s} = \boldsymbol{\eta}_{2s}$, $\dot{\boldsymbol{\eta}}_{2s} = \boldsymbol{v}$, where $\boldsymbol{v}$ is the new control input to drive $\boldsymbol{\eta}_s$ to zero exponentially. We define the zero dynamics states as

$$\xi_{1s} = \theta_s(\boldsymbol{q}_a), \xi_{2s} = \boldsymbol{D}_{sq}^s(\boldsymbol{q}_a)\dot{\boldsymbol{q}}_{es}, x_{1s} = x_s, x_{2s} = \boldsymbol{D}_{sx}^s(\boldsymbol{q}_a)\dot{\boldsymbol{q}}_{es} \tag{3.21}$$

where $\boldsymbol{D}_{sq}^s(\boldsymbol{q}_a)$ and $\boldsymbol{D}_{sx}^s(\boldsymbol{q}_a)$ are the rows in $\boldsymbol{D}_s^s(\boldsymbol{q}_a)$ that correspond to the unactuated angle q_1 and slip distance x_s respectively. Note that $\boldsymbol{D}_s^s(\boldsymbol{q}_a)$ does not depend on x_s.

Similar to (3.12), we obtain the coordinate transformation between the new coordinates $\boldsymbol{\eta}_s$, $\boldsymbol{\xi}_s$, and $\boldsymbol{x}_s = [x_{1s}\, x_{2s}]^T$ with the original states $\boldsymbol{x}_{es}$ as

$$\begin{bmatrix} \boldsymbol{\eta}_{1s} \\ \xi_{1s} \\ x_{1s} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Phi}_s(\boldsymbol{q}_a) \\ x_s \end{bmatrix}, \quad \begin{bmatrix} \boldsymbol{\eta}_{2s} \\ \xi_{2s} \\ x_{2s} \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{\partial \boldsymbol{h}_s}{\partial \boldsymbol{q}_a} & \boldsymbol{0} \\ \multicolumn{2}{c}{\boldsymbol{D}^s_{eq}(\boldsymbol{q}_a)} \\ \multicolumn{2}{c}{\boldsymbol{D}^s_{ex}(\boldsymbol{q}_a)} \end{bmatrix}}_{\boldsymbol{\lambda}(\boldsymbol{q}_a)} \dot{\boldsymbol{q}}_{es} = \boldsymbol{\lambda}(\boldsymbol{q}_a)\dot{\boldsymbol{q}}_{es},$$

where $\boldsymbol{\Phi}_s(\boldsymbol{q}_a) = [\boldsymbol{h}_s\, \theta_s]^T$ and the inverse transformation is

$$\boldsymbol{q}_{es} = \begin{bmatrix} \boldsymbol{q}_a \\ x_s \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Phi}_s^{-1}\left(\begin{bmatrix} \boldsymbol{\eta}_{1s} \\ \xi_{1s} \end{bmatrix}\right) \\ x_{1s} \end{bmatrix}, \dot{\boldsymbol{q}}_{es} = \boldsymbol{\lambda}^{-1}(\boldsymbol{q}_a) \begin{bmatrix} \boldsymbol{\eta}_{2s} \\ \xi_{2s} \\ x_{2s} \end{bmatrix}. \tag{3.22}$$

The zero dynamics is given as

$$\dot{\xi}_{1s} = \frac{\partial \theta_s}{\partial \boldsymbol{q}_a}\dot{\boldsymbol{q}}_a,\ \dot{x}_{1s} = \dot{x}_s, \tag{3.23}$$

$$\dot{\xi}_{2s} = \dot{\boldsymbol{q}}_{es}^T \frac{\partial (\boldsymbol{D}^s_{eq})^T}{\partial \boldsymbol{q}_{es}} \dot{\boldsymbol{q}}_{es} - \boldsymbol{C}^s_{eq}\dot{\boldsymbol{q}}_s - \boldsymbol{G}^s_{eq} \tag{3.24}$$

$$\dot{x}_{2s} = \dot{\boldsymbol{q}}_{es}^T \frac{\partial (\boldsymbol{D}^s_{ex})^T}{\partial \boldsymbol{q}_s} \dot{\boldsymbol{q}}_s - \boldsymbol{C}^s_{ex}\dot{\boldsymbol{q}}_s - \boldsymbol{G}^s_{ex}. \tag{3.25}$$

From the property of robot motion equation (3.15), we obtain

$$C^s_{ex} = \dot{q}_s^T \frac{\partial (D^s_{es,us})^T}{\partial q_s}. \tag{3.26}$$

We simplify the zero dynamics by substituting (3.26) into (3.25) with (3.22) and $\boldsymbol{\eta}_s = \boldsymbol{0}$, and obtain

$$\begin{aligned} \begin{bmatrix} \dot{\xi}_{1s} \\ \dot{x}_{1s} \end{bmatrix} &= \begin{bmatrix} \frac{\partial \theta_s}{\partial \boldsymbol{q}_a} & 0 \\ \boldsymbol{0}_N^T & 1 \end{bmatrix} \boldsymbol{\lambda}^{-1}(\boldsymbol{q}_a) \begin{bmatrix} \boldsymbol{0}_{N-1} \\ \xi_{2s} \\ x_{2s} \end{bmatrix} =: k_{1s}(\xi_{1s}) \begin{bmatrix} \xi_{2s} \\ x_{2s} \end{bmatrix} \\ \dot{\xi}_{2s} &= \dot{\boldsymbol{q}}_{es}^T \frac{\partial (\boldsymbol{D}^s_{eq})^T}{\partial \boldsymbol{q}_{es}} \dot{\boldsymbol{q}}_{es} - \boldsymbol{C}^s_{eq}\dot{\boldsymbol{q}}_{es} - \boldsymbol{G}^s_{eq} =: k_{2s}(\xi_{1s}, \xi_{2s}, x_{2s}) \\ \dot{x}_{2s} &= -\boldsymbol{G}^s_{ex} =: k_{3s}(\xi_{1s}). \end{aligned} \tag{3.27}$$

3.3.2 Multi-step slip recovery process and stability

A slip recovery process can be considered a series of gaits. As we will describe in Section 3.4, we here use a recovery gait sequence as an example to illustrate the principles and stability of the recovery process. The results describing in the following can be extended to any other gait sequences.

As shown in Figure. 3.2, for normal walking, the dynamics is described by non-slip continuous dynamics $\mathcal{S}_n$ and transition through non-slip foot impact $\mathcal{H}_n^n$. The normal gait profile (virtual constraints) is denoted by α_n. When slip happens, normal walking swing phase $\mathcal{S}_n$ with gait α_n transits through the slip impact $\mathcal{H}_n^s$ to slip swing phase $\mathcal{S}_s$. In $\mathcal{S}_s$, human tries to touch down the swing foot as soon as possible to find the new support. This slip gait is α_s. Once the swing foot touches down, the slip phase $\mathcal{S}_s$ transits back to the non-slip walking phase $\mathcal{S}_n$ through stick impact $\mathcal{H}_s^n$. A recovery gait profile α^r is adopted during this phase. Finally, the gait is successfully recovered back to the normal walking phase $\mathcal{S}_n$ with gait profile α_n. In the above description, the slip recovery process can be summarized as

$$\mathcal{S}_n(\alpha_n) \xrightarrow{\mathcal{H}_n^s} \mathcal{S}_s(\alpha_s) \xrightarrow{\mathcal{H}_s^n} \mathcal{S}_n(\alpha_r) \xrightarrow{\mathcal{H}_n^n} \mathcal{S}_n(\alpha_n). \tag{3.28}$$

Note from Figure. 3.2 that a stable cyclic gait could exist for single-stance slip phase $\mathcal{S}_s$ with gait profile α_s through impact $\mathcal{H}_s^s$. This gait does not happen usually in human walking locomotion because it requires skills to regulate highly dynamic motion under foot slip. One example for such gaits is used in figure skating skills in which both stepping and foot slipping co-exist for stable gaits. Understanding of those motor skills is out of scope of this chapter.

The nonslip or slip swing phase zero dynamics controllers only drive the system states onto its own zero dynamics space. To build a zero dynamics space covering the whole slip recovery process, it is necessary to guarantee that the whole zero dynamics space is invariant under slip ($\mathcal{H}_n^s$) and recovery impacts ($\mathcal{H}_s^n$). Similar to (3.14), the following conditions should be satisfied for the recovery sequence (3.28)

$$\boldsymbol{\Delta}_s(\boldsymbol{S} \cap \boldsymbol{Z}_{\alpha_n}) \subset \boldsymbol{Z}_{\alpha_s}, \boldsymbol{\Delta}_n(\boldsymbol{S} \cap \boldsymbol{Z}_{\alpha_s}) \subset \boldsymbol{Z}_{\alpha_r}, \tag{3.29}$$

$$\boldsymbol{\Delta}_n(\boldsymbol{S} \cap \boldsymbol{Z}_{\alpha_r}) \subset \boldsymbol{Z}_{\alpha_n}, \tag{3.30}$$

where Z_{α_n}, Z_{α_s} and Z_{α_r} are the normal walking, slipping, recover step zero dynamics space respectively.

Assuming pre-impact zero dynamics state of a normal walking gait α_n is given as $\boldsymbol{\xi}^- \in \boldsymbol{S} \cap \boldsymbol{Z}_{\alpha_n}$, the pre-impact full state is then $\boldsymbol{x}_a(\boldsymbol{\xi}^-)$ by applying (3.13) with $\boldsymbol{\eta} = \boldsymbol{0}$. After the slip impact (3.18), the initial full state of slip swing phase is $\boldsymbol{x_{es}}^+(\boldsymbol{\xi}^-)$. Condition (3.29) requests that

$$\boldsymbol{\eta}_{1s}(\boldsymbol{x_{es}}^+) = \boldsymbol{h}_{\alpha_s}(\boldsymbol{q}_{es}^+) = \boldsymbol{0}, \boldsymbol{\eta}_{2s}(\boldsymbol{x_{es}}^+) = \frac{\partial \boldsymbol{h}_{\alpha_s}}{\partial \boldsymbol{q}_a}(\boldsymbol{q}_{es}^+)\dot{\boldsymbol{q}}_{es}^+ = \boldsymbol{0}. \tag{3.31}$$

Meanwhile, in the HZD space, after the impact the state $[\xi_{1,s}^+, \xi_{2,s}^+, x_{1,s}^+, x_{2,s}^+]^T \in Z_{\alpha_s}$ is related to the pre-impact state $\boldsymbol{\xi}^- \in \boldsymbol{S} \cap \boldsymbol{Z}_{\alpha_n}$ as

$$\begin{bmatrix} \xi_{1,s}^+ \\ \xi_{2,s}^+ \\ x_{1,s}^+ \\ x_{2,s}^+ \end{bmatrix} = \begin{bmatrix} \theta_s(\boldsymbol{q}_a^+) \\ \boldsymbol{D}_{sq}^s(\boldsymbol{q}_a^+)\dot{\boldsymbol{q}}_{es}^+ \\ x_s^+ \\ \boldsymbol{D}_{sx}^s(\boldsymbol{q}_a^+)\dot{\boldsymbol{q}}_{es}^+ \end{bmatrix} = \begin{bmatrix} \theta_s(\boldsymbol{S}\boldsymbol{q}_a^-) \\ \boldsymbol{D}_{sq}^s(\boldsymbol{S}\boldsymbol{q}_a^-)\Delta_s\dot{\boldsymbol{q}}_a^- \\ 0 \\ \boldsymbol{D}_{sx}^s(\boldsymbol{S}\boldsymbol{q}_a^-)\Delta_s\dot{\boldsymbol{q}}_a^- \end{bmatrix}$$

$$= \begin{bmatrix} \theta_s(\boldsymbol{S}\boldsymbol{q}_a(\xi_1^-)) \\ \boldsymbol{D}_{sq}^s(\boldsymbol{S}\boldsymbol{q}_a(\xi_1^-))\Delta_s\dot{\boldsymbol{q}}_a(\xi_2^-) \\ 0 \\ \boldsymbol{D}_{sx}^s(\boldsymbol{S}\boldsymbol{q}_a(\xi_1^-))\Delta_s\dot{\boldsymbol{q}}_a(\xi_2^-) \end{bmatrix} = \boldsymbol{\delta}_n^s(\boldsymbol{\xi}^-) \tag{3.32}$$

where the first second and third equality signs are obtained by applying (3.21),(3.18) and (3.13) respectively.

To compute the ending state in the slip swing phase, we integrate (3.27)with respect to time, until either $\xi_{1s} = \xi_{1s}^+$ (start of the step) or $\xi_{1s} = \xi_{1s}^-$ (ending of the step) which means either not being able to complete this step and return to the initial configuration of this phase or a complete step is achieved. We denote the ending state as

$$\begin{bmatrix} \xi_{2,s}^{-} \\ \boldsymbol{x}_s^{-} \end{bmatrix} = \boldsymbol{\Phi}_s(\boldsymbol{\xi}^{+}, \boldsymbol{x}_s^{+}). \tag{3.33}$$

At the end of slip swing phase, the swing foot impacts on the ground and the slipping foot lifts immediately. The pre-impact zero dynamic state is $[\boldsymbol{\xi}_s^{-}, \boldsymbol{x}_s^{-}]^T \in \boldsymbol{S} \cap \boldsymbol{Z}_{\alpha_s}$. The full state is $\boldsymbol{x}_{es}(\boldsymbol{\xi}_s^{-}, \boldsymbol{x}_s^{-})$ according to (3.22) under the condition that $\boldsymbol{\eta}_s = \boldsymbol{0}$. After the stick impact the initial state of recover step is $\boldsymbol{x}_a^{+}$ from impact model (3.7). Condition (3.29) requests that

$$\boldsymbol{\eta}_{1,\alpha_r} = \boldsymbol{h}_{\alpha_r}(\boldsymbol{q}_a^{+}) = \boldsymbol{0},\ \boldsymbol{\eta}_{2,\alpha_r} = \frac{\partial \boldsymbol{h}_{\alpha_r}}{\partial \boldsymbol{q}_a}(\boldsymbol{q}_a^{+})\dot{\boldsymbol{q}}_a^{+} = \boldsymbol{0} \tag{3.34}$$

In the HZD space, the initial zero dynamics state of the recover step swing phase $\boldsymbol{\xi}^{+} \in Z_{\alpha_r}$ is related to the pre-impact state $[\boldsymbol{\xi}_s^{-}, \boldsymbol{x}_s^{-}]^T \in \boldsymbol{S} \cap \boldsymbol{Z}_{\alpha_s}$ as

$$\begin{aligned} \begin{bmatrix} \xi_1^{+} \\ \xi_2^{+} \end{bmatrix} &= \begin{bmatrix} \theta(\boldsymbol{q}_a^{+}) \\ \boldsymbol{D}_a(\boldsymbol{q}_a^{+})\dot{\boldsymbol{q}}_a^{+} \end{bmatrix} = \begin{bmatrix} \theta(\boldsymbol{S}\boldsymbol{q}_a^{-}) \\ \boldsymbol{D}_a(\boldsymbol{S}\boldsymbol{q}_a^{-})(\Delta_n \dot{\boldsymbol{q}}_e^{-} + \boldsymbol{b}_n) \end{bmatrix} \\ &= \begin{bmatrix} \theta(\boldsymbol{S}\boldsymbol{q}_a(\xi_{1s}^{-})) \\ \boldsymbol{D}_a(\boldsymbol{S}\boldsymbol{q}_a(\xi_{1s}^{-}))(\Delta_n \dot{\boldsymbol{q}}_a(\xi_s^{-}, x_s^{-}) + \boldsymbol{b}_n) \end{bmatrix} = \boldsymbol{\delta}_s^{n}(\boldsymbol{\xi}_s^{-}, \boldsymbol{x}_s^{-}). \end{aligned} \tag{3.35}$$

where the first second and third equality sign are obtained by applying (3.9) (3.7) and (3.22) respectively.

The continuous recovery zero dynamics is described by (3.11). We solve (3.11) with respect to time until either $\xi_1 = \xi_1^{+}$ or $\xi_1 = \xi_1^{-}$ which indicates either not being able to complete this recover step and returning to the initial configuration of this phase or a complete recover step is achieved. We denote the ending state as

$$\xi_2^{-} = \Phi_{\alpha_r}(\boldsymbol{\xi}^{+}). \tag{3.36}$$

At the end of recover step swing phase, the zero dynamic state is $\boldsymbol{\xi}^- \in \boldsymbol{S} \cap \boldsymbol{Z}_{\alpha_r}$ and the full state is $\boldsymbol{x}_a(\boldsymbol{\xi}^-)$. After the impact the initial state is $\boldsymbol{x}_a^+$ from (3.7). Condition (3.30) requests that

$$\boldsymbol{\eta}_{1,\alpha_n} = \boldsymbol{h}_{\alpha_n}(\boldsymbol{q}_a^+) = \boldsymbol{0}, \boldsymbol{\eta}_{2,\alpha_n} = \frac{\partial \boldsymbol{h}_{\alpha_n}}{\partial \boldsymbol{q}_a}(\boldsymbol{q}_a^+)\dot{\boldsymbol{q}}_a^+ = \boldsymbol{0}.$$

Meanwhile, in the HZD space, the initial zero dynamics state of the normal walking swing phase $\boldsymbol{\xi}^+ \in Z_{\alpha_n}$ is related to pre-impact state $\boldsymbol{\xi}^- \in S \cap Z_{\alpha_r}$ as

$$\begin{bmatrix} \xi_1^+ \\ \\ \xi_2^+ \end{bmatrix} = \begin{bmatrix} \theta(\boldsymbol{q}_a^+) \\ \\ \boldsymbol{D}_a(\boldsymbol{q}_a^+)\dot{\boldsymbol{q}}_a^+ \end{bmatrix} = \begin{bmatrix} \theta(\boldsymbol{S}\boldsymbol{q}_a^-) \\ \\ \boldsymbol{D}_a(\boldsymbol{S}\boldsymbol{q}_a^-)(\Delta_n \dot{\boldsymbol{q}}_e^- + \boldsymbol{b}_n) \end{bmatrix}$$
$$= \begin{bmatrix} \theta(\boldsymbol{S}\boldsymbol{q}_a(\xi_1^-)) \\ \\ \boldsymbol{D}_a(\boldsymbol{S}\boldsymbol{q}_a(\xi_1^-))(\Delta_n \dot{\boldsymbol{q}}_a(\xi^-) + \boldsymbol{b}_n) \end{bmatrix} = \boldsymbol{\delta}_n^n(\boldsymbol{\xi}^-). \tag{3.37}$$

where the first second and third equality sign are obtained by applying (3.9) (3.7) and (3.22) respectively.

Finally, we integrate (3.11) with respect to time until either $\xi_1 = \xi_1^+$ to $\xi_1 = \xi_1^-$ which means either not being able to complete a step and returning to the initial configuration of this step or a complete step is achieved. We denote the final state as

$$\xi_2^- = \Phi_{\alpha_n}(\boldsymbol{\xi}^+). \tag{3.38}$$

The whole slip recovery process can be represented by state transiting in hybrid zero dynamics space. Starting from the moment right before the slip impact, the initial zero dynamics state $\boldsymbol{\xi}^- \in \boldsymbol{S} \cap \boldsymbol{Z}_{\alpha_n}$ is mapped by $\boldsymbol{\delta}_n^s$, $\boldsymbol{\Phi}_s$, $\boldsymbol{\delta}_s^n$, $\boldsymbol{\Phi}_{\alpha_r}$, $\boldsymbol{\delta}_n^n$ and $\boldsymbol{\Phi}_{\alpha_n}$ successively. The returned value of ξ_2 after the whole slip recovery process is expressed as

$$\xi_2 = \boldsymbol{\Phi}_{\alpha_n} \circ \boldsymbol{\delta}_n^n \circ \boldsymbol{\Phi}_{\alpha_r} \circ \boldsymbol{\delta}_s^n \circ \boldsymbol{\Phi}_s \circ \boldsymbol{\delta}_n^s(\xi_2^-) =: \rho_s(\xi_2^-) \tag{3.39}$$

For stable periodic normal walking gait, choosing Poincaré section as $\boldsymbol{\xi}^- \in \boldsymbol{S} \cap \boldsymbol{Z}_{\alpha_n}$, from [11], the Poincaré first return map has stable fixed point ξ_2^- i.e.

$$\xi_2^- = \Phi_{\alpha_n} \circ \delta_n^n(\xi_2^-) = \rho(\xi_2^-), \tag{3.40}$$

and

$$\frac{\partial \rho}{\partial \xi_2}(\xi_2^-) < 1. \tag{3.41}$$

These properties guarantee that there exists a region $\xi_2^- \in R \in S \cap Z_{\alpha_n}$ so that for every $\xi_2 \in R$

$$|\rho(\xi_2) - \xi_2^-| \leq |\xi_2 - \xi_2^-|. \tag{3.42}$$

In the HZD space, the success slip recover to normal gait is equivalent to

$$\xi_2 = \rho_s(\xi_2^-) \in R. \tag{3.43}$$

3.4 Experimental results

Figure. 3.3 shows the experimental configuration. We follow the same experiments description in [2, 50] and we briefly review the experimental description. We conduict the indoor walking experiments on a wooden platform. A portion of the platform is painted by soap film to create slip and recovery gaits when the subject steps on the slippery surface. The subject was asked to first practice walking on the platform before slipping tests. The low-friction portion is not noticeable to the subject such that the normal gait is used before slip happens. The human motion is obtained by the optical motion capture system (8 Bonita cameras from Vicon Inc.) Two 6-DOF force sensors (model SS-1 from INSENCO Co., Ltd) are used inside the shoe to measure the three dimensional ground reaction forces and torques (Figure. 3.3). Wireless data transmission is setup between the force sensors and the host computer. All sensors and the motion capture system are synchronized for data collection.

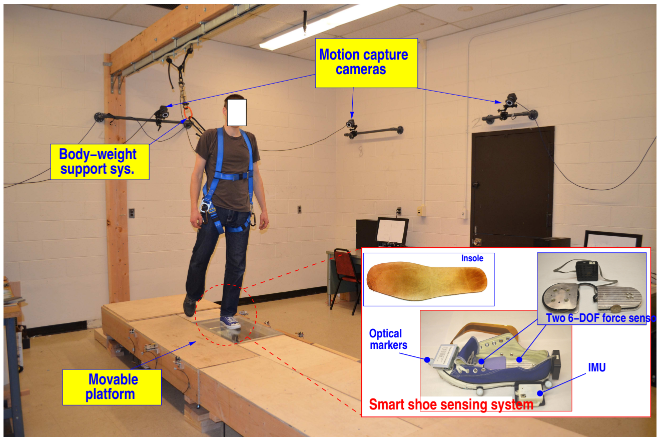
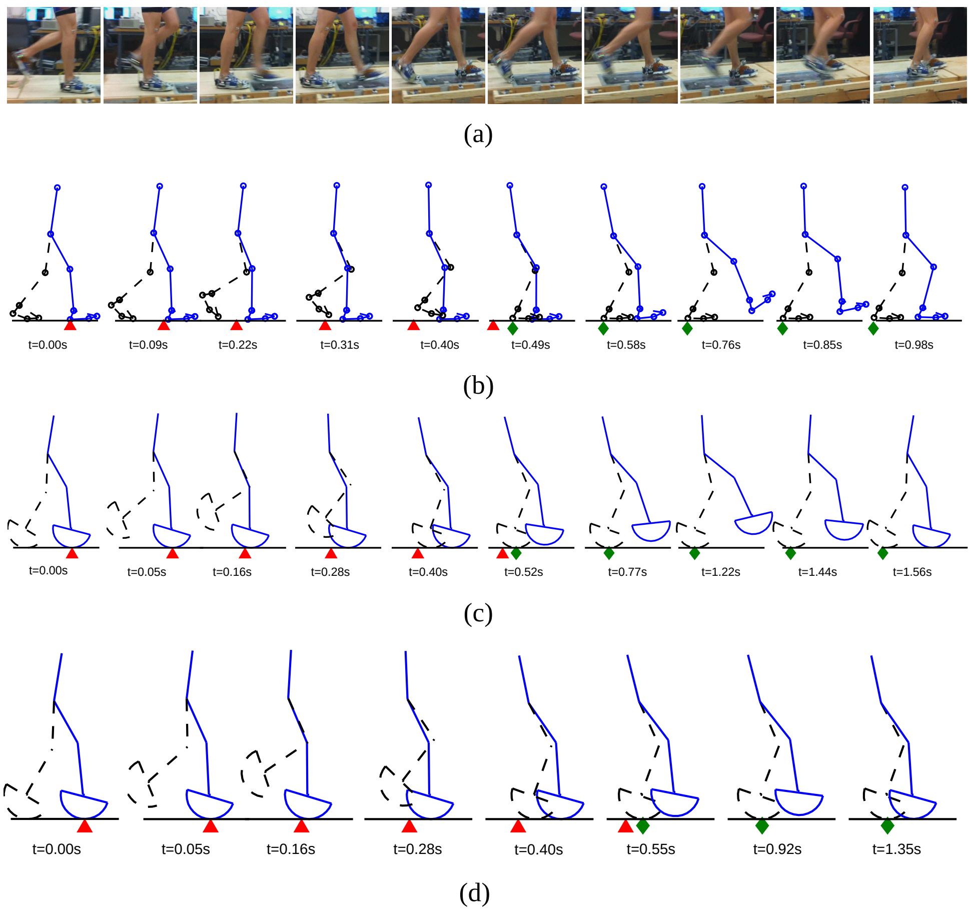


Figure 3.3: The slip and fall experimental setup with various sensor suites.

We first take the same experiment in [2] to demonstrate the HZD model prediction of a slip recovery process. Figure. 3.4(a) shows a video snapshot of a human slip recovery process. The subject starts the normal single-stance gait ($\mathcal{S}_s$ with gait profile α_n). At $t = 0.32$ s, the (left) swing leg touches down on the slippery ground and then starts slipping. Thus, the human gait enters the slip phase ($\mathcal{S}_s$ with gait profile α_s) [1]. Until $t = 0.96$ s when the (right) swing foot touches down, the (left) stance foot leaves the ground and the gait becomes a recovered single-stance phase without slipping ($\mathcal{S}_n$ with gait profile α_r). Finally, a few more recovery single-stance steps brings the normal gait (gait profile α_n). The entire recovery follows the process given in (3.28). Figure. 3.4(b) shows the skeleton snapshots in experiments and Figure. 3.4(c) demonstrates the skeleton snapshots predicted by the HZD model.

Figure. 3.5 shows the phase portraits of the zero dynamics for the recovery process. Figure. 3.5(a) illustrates the phase portrait in the ξ_1-ξ_2-x_{1s} coordinates and Figure. 3.5(b) shows the phase portrait in the ξ_1-ξ_2 plane. Comparing with the normal walking gait, it is clearly shown in these figures that the walking with foot slip generates much richer zero dynamics characteristics. The slip recovery process is a high-dimensional manifold and consists multiple portions of the phase portraits in 3D space

[1] We here neglect a short-duration double-stance phase



(a)

(b)

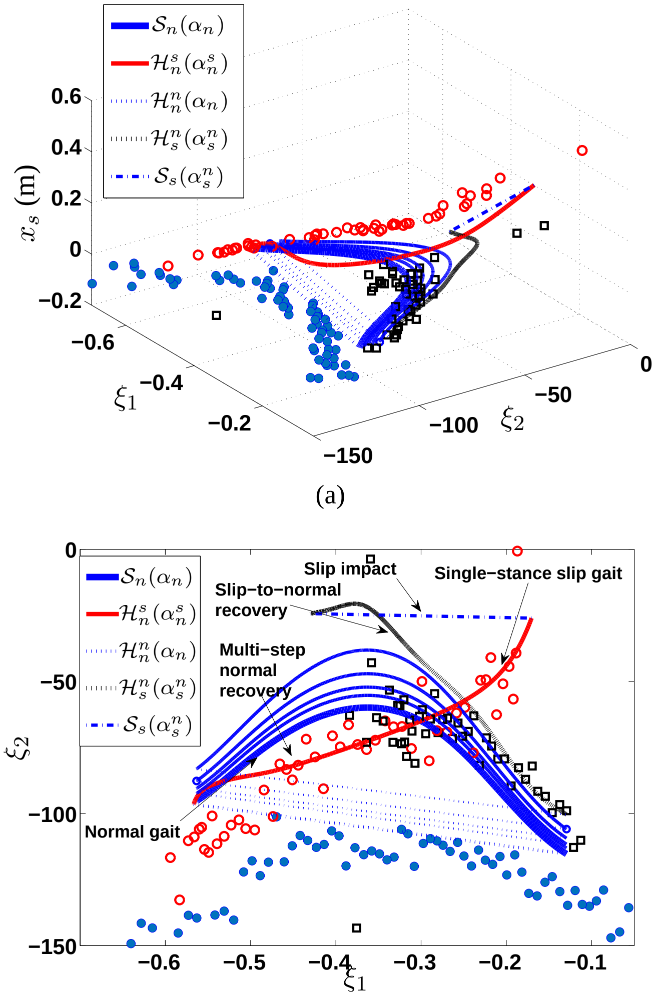
(c)

(d)

Figure 3.4: A snapshot of a human slip recovery process. (a) Video snapshot of the gaits. (b) Skeleton snapshops. (c) Skeleton prediction by the HZD model for a successful recovery. (d) Skeleton prediction by the HZD model for an unsuccessful recovery. In (b)-(d), the red-triangle markers indicate the slip initation locations and the green-diamond markers indicate the foot touchdown locations.

as shown in Figure. 3.5(a). As shown in the figures, the HZD model predictions for the normal walking $\mathcal{S}_n$ (gait profile α_n, i.e., solid blue curves) and foot-slip gait $\mathcal{S}_s$ (impact $\mathcal{H}_n^s$ and gait profile α_s, i.e., solid red curve) match with the experiments, that is, solid blue and empty red circular markers, respectively. Moreover, as shown in Figure. 3.5(b), after the single-stance slip phase $\mathcal{S}_s$, the subject tries to recover from the slip by taking slip impact (blue dash-dot line) and then slip-to-normal recovery $\mathcal{H}_s^n$ with gait profile α_r (black dot curve for model prediction and square dots for experiments.) The entire recovery process can be re-generated by the HZD model.

Figure. 3.6 further shows the HZD model prediction with experiments for the joint

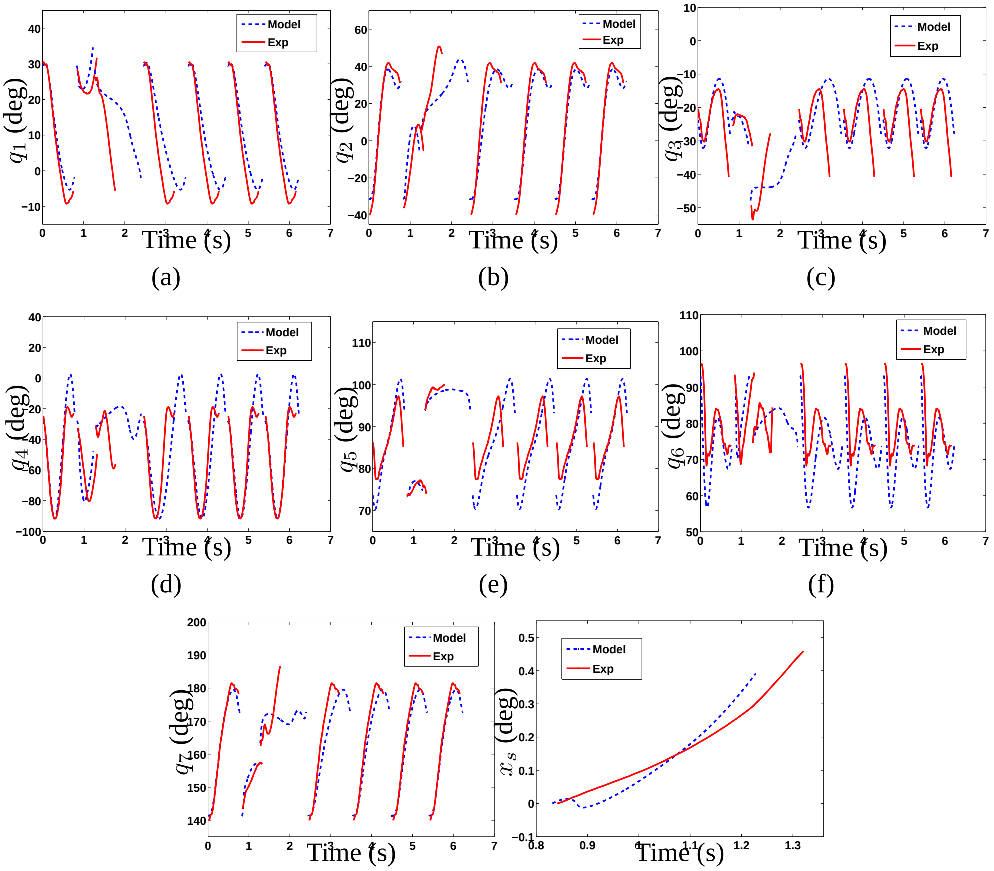


(a)

(b)

Figure 3.5: (a) 3D phase portrait (ξ_1-ξ_2-x_s) during the slip recovery process. (b) 2D phase portrait in ξ_1-ξ_2 plane. In both plots, the empty circles are experimental data during the phase $\mathcal{H}_n^s$ with gait profile α_n^s. The solid circles are experimental data during normal walking phase $\mathcal{S}_n$ with gait profile α_n.

angles and slipping distance. Figure. 3.6(a)-3.6(g) shows the joint angle comparison results of the model prediction and experiments and Figure. 3.6(h) shows the slipping distance comparison. The gait comparisons are shown in Figures. 3.4(b) and 3.4(c). These results clearly demonstrate that the HZD model predictions match the experiments during the slip recovery process. Note that the HZD model prediction results have slightly different phase timings due to the different parameters used in the simulation comparing with the experiments.



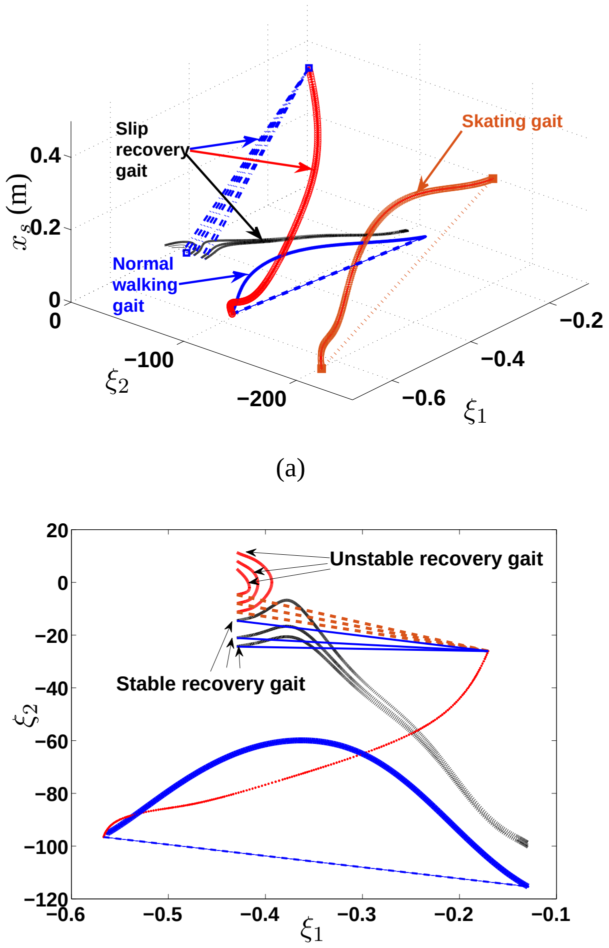
(a) (b) (c)

(d) (e) (f)

(g) (h)

Figure 3.6: (a)-(g): Joint angle (q_1 to q_7) comparisons between the HZD model prediction and the experiments during slip recovery gait. (h) Slipping distance x_s of the (left) stance leg during the slip recovery experiment.

Figure. 3.7(a) shows a collection of the phase portraits of the normal walking gait, slip recovery gait and skating gaits in the ξ_1-ξ_2-x_{1s} space. The steady skating gait is generated by considering both slipping and walking gaits as the skilled human motor locomotion. It is clear that the zero dynamics of the normal walking gait are located in the ξ_1-ξ_2 plane while the skating gait is in the 3D space with motion in the x_{1s} direction. The slip recovery gait consists of a series of transient motion that deviates from and then returns to the normal walking gaits. Figure. 3.7(b) further illustrates the slip recovery gaits in which both successful, stable recovery (blue and black curves) and unsuccessful, unstable recovery (dash and solid red curves) gaits are plotted. Both sets of recovery gaits are obtained by enforcing the same virtual constraints. The only



(a)

(b)

Figure 3.7: (a) Phase portraits of normal walking gait, slip recovery gait, and skating gaits in 3D space. (b) Phase portraint of the successful and unsuccessful slip recovery gaits in the ξ_1-ξ_2 plane.

difference for these difference trajectories is the value of $\boldsymbol{b}_n$ in (3.7) of the impact from slip gait to recover gait. The difference of $\boldsymbol{b}_n$ in (3.7) gives different initial ξ_2 values for the recover gait which imply the angular moments of falling. Once ξ_2 passes 0 and even be positive, the progression variable $\xi_1 = \theta$ changes direction and becomes decreasing which means cannot complete the recover step. Figures. 3.4(c) and 3.4(d) demonstrate the gait profiles for a successful and an unsuccessful (falling) slip recovery, respectively.

3.5 Summary

We presented hybrid zero dynamics (HZD) for human walking gaits with foot slip. The HZD is an extension of the existing dynamics for normal walking locomotion. We explicitly derived and presented the HZD for human walking with foot slip that contains two additional zero dynamics state. It is interesting to show that the HZD under foot slip present rich human motor skills, including the normal walking, slip recovery gaits and highly skilled skating motion. We presented stability condition for slip recovery gait and also successfully demonstrated the HZD-based recovery simulation. Human experiments were used to validate and compare the HZD model prediction and the results have demonstrated the effectiveness of the model and the analysis.

Chapter 4
Learning Latent Space Dynamics of Biped Locomotion

4.1 Introduction

Modeling human walking locomotion is important for studying human biomechanics and for designing assistive and rehabilitation devices and robotics. Physical principle-based and data-driven learning models are among the most commonly used methods to capture the bipedal dynamics. Human musculoskeletal systems can be modeled as a multi-link structure and then physical principals for rigid bodies provide dynamic models for human walking [11]. For example, two-, five-, and eight-DOF bipedal models are used and discussed to design robotic controllers in [46,47,51], respectively. Using the hybrid zero dynamics (HZD) concept [46], a low-dimensional cyclic model is obtained to achieve periodic gaits [11]. Extension of the bipedal robotic models to human locomotion are reported in [52] and [44]. In recent work [2], the bipedal models in [44,52] are extended to study human slip-and-fall walking gaits.

Data-driven modeling approaches, such as learning from demonstration, can capture human motion and motor skills in lower-dimensional space. For example, in [14], a Gaussian process dynamic model (GPDM) is presented to capture the low-dimensional latent dynamics for human motion such as walking etc. In [21], a manifold learning method is used for human motion tracking with visual observations.

Integration of the physical models with the learning models are also used in various applications. In [53,54], a perturbed dynamic approach is presented to synthesize the responsive motion by combining learned latent dynamic models with physics-based motion perturbations. In [37, 55], a hybrid dynamic model is used to study human-bicycle interactions and pose estimation applications.

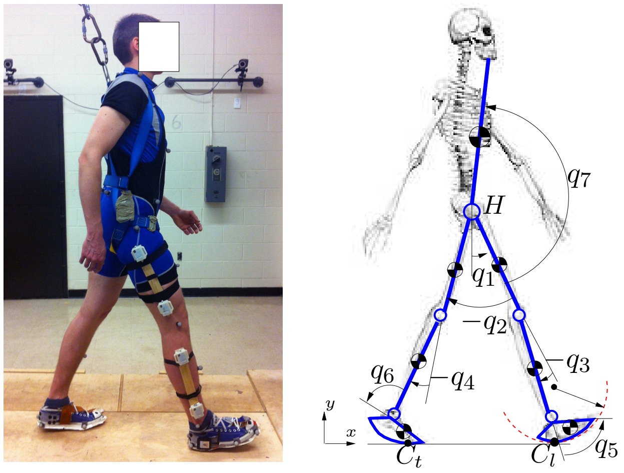
The above-mentioned physical principle-based and data-driven machine learning-based dynamic models are two different methods to capture human movement and locomotion. The goal of this work is to present the connection between these two modeling approaches for human bipedal walking. We reveal that the zero dynamics of physical model indeed maximizes the posterior probability of the learning problem in learned latent dynamic model.

The remaining of this chapter is organized as follows. We first introduce the two modeling methods and the problem statement in Section 4.2. Section 4.3 presents the main results about the relationships between these two models. We present the experimental results in Section 4.4. Finally, we conclude this chapter and discuss future work directions in Section 4.5.

4.2 Two bipedal models and problem statement

Figure. 4.1 illustrates the setup of the coordinates for human walking bipedal in the sagittal plane. The human body is considered as a seven-link rigid body with the HAT (head, arms and trunk) as one link that is connected to the left- and right-thigh. The human has two active hip joints, two active knee joints and two active ankle joints. We denote the relative angles at joints as q_i, $i = 2, \dots, 7$, and the absolute angle of the leading stance leg orientation with respect to the vertical direction as q_1 [2, 55].

We define $\boldsymbol{q} = [q_1 \;\; \cdots \;\; q_7]^T$ as the gait configuration. The foot-floor contact is considered as a circular disk rolling on the solid ground. A human walking cycle consists of a single-stance phase (i.e., only the stance foot rolls on the ground) and the double-stance phase (i.e., both feet roll on the ground). Due to the heel touch-down and impact, the bipedal walking has a hybrid dynamics property [11]. To simplify our



(a) (b)

Figure 4.1: Schematic of the 7-link human walking model with curved foot contact [2].

discussion and presentation, we mainly focus on the single-stance bipedal dynamics and the results can be readily extended to the double-stance phase.

4.2.1 Physical model

The seven-link biped in the single-stance phase is underactuated [11] and the equations of motion are given as

$$\boldsymbol{D}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{B}\boldsymbol{u}, \tag{4.1}$$

where $\boldsymbol{u} \in \mathbb{R}^6$ is the joint torque vector, matrices $\boldsymbol{D}(\boldsymbol{q})$, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ and $\boldsymbol{G}$ are obtained from Lagrange's equations [2]. An output (virtual constraint) gait $\boldsymbol{y} \in \mathbb{R}^6$ is given by

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{q}) = \boldsymbol{H}_0\boldsymbol{q} - \boldsymbol{h}_d(\theta), \tag{4.2}$$

where $\boldsymbol{H}_0 \in \mathbb{R}^{6\times 7}$, $\theta = \boldsymbol{c}\boldsymbol{q}$, $\boldsymbol{c} \in \mathbb{R}^{1\times 7}$ is the mapping vector, and function $\boldsymbol{h}_d : \mathbb{R} \mapsto \mathbb{R}^6$. The control task is to drive the configuration $\boldsymbol{q}$ onto the sub-manifold defined by $\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{q}) = \boldsymbol{0}$. To achieve this control task, we apply a coordinate transformation

$$\boldsymbol{\eta}_1 = \boldsymbol{h}(\boldsymbol{q}),\ \boldsymbol{\eta}_2 = \dot{\boldsymbol{\eta}}_1 = \boldsymbol{H}_\theta\dot{\boldsymbol{q}},\ \xi_1 = \theta(\boldsymbol{q}),\ \xi_2 = \boldsymbol{D}_a(\boldsymbol{q})\dot{\boldsymbol{q}},$$

where $\boldsymbol{H}_\theta = \boldsymbol{H}_0 - \frac{\partial \boldsymbol{h}_d}{\partial \theta}\boldsymbol{c} \in \mathbb{R}^{6\times 7}$, $\boldsymbol{D}_a(\boldsymbol{q}) \in \mathbb{R}^{1\times 7}$ is formed by the row of $\boldsymbol{D}(\boldsymbol{q})$ that corresponds to the unactuated joint angle q_1. The inverse transformation is

$$\boldsymbol{q} = \boldsymbol{H}_C^{-1}\begin{bmatrix}\boldsymbol{\eta}_1 + \boldsymbol{h}_d(\xi_1) \\ \xi_1\end{bmatrix}, \dot{\boldsymbol{q}} = \boldsymbol{H}_D^{-1}\begin{bmatrix}\boldsymbol{\eta}_2 \\ \xi_2\end{bmatrix}, \tag{4.3}$$

where

$$\boldsymbol{H}_C = \begin{bmatrix}\boldsymbol{H}_0 \\ \boldsymbol{c}\end{bmatrix}, \ \boldsymbol{H}_D = \begin{bmatrix}\boldsymbol{H}_\theta \\ \boldsymbol{D}_a(\boldsymbol{q})\end{bmatrix}.$$

If $\boldsymbol{u}$ is properly chosen to drive $\boldsymbol{\eta}_1 = \boldsymbol{\eta}_2 = \boldsymbol{0}$, the zero dynamics $\boldsymbol{\Sigma}_{ZD}$ is

$$\boldsymbol{\Sigma}_{ZD}: \begin{cases}\dot{\xi}_1 = \boldsymbol{c}\dot{\boldsymbol{q}} = \boldsymbol{c}\boldsymbol{H}_D^{-1}\begin{bmatrix}\boldsymbol{0} \\ \xi_2\end{bmatrix} =: g_1(\xi_1)\xi_2, \\ \dot{\xi}_2 = \dot{\boldsymbol{D}}_a(\boldsymbol{q})\dot{\boldsymbol{q}} - \boldsymbol{C}_a\dot{\boldsymbol{q}} - \boldsymbol{G}_a =: g_2(\xi_1, \xi_2),\end{cases} \tag{4.4}$$

where (4.3) is used to express $(\boldsymbol{q}, \dot{\boldsymbol{q}})$ into $\boldsymbol{\xi} = [\xi_1 \ \xi_2]^T$ (with $\boldsymbol{\eta}_1 = \boldsymbol{\eta}_2 = \boldsymbol{0}$). Note that $g_1(\xi_1) = \boldsymbol{c}\left(\boldsymbol{H}_D^{-1}\right)_7$ is only a function of $\boldsymbol{q}$ and thus of ξ_1, where $\left(\boldsymbol{H}_D^{-1}\right)_7$ is the 7th column vector of $\boldsymbol{H}_D^{-1}$.

4.2.2 Learning model

The learning model is built on the pose measurements. We denote the joint angle data set as $\boldsymbol{Y} = \{\boldsymbol{q}\}^M \in \mathbb{R}^{7\times M}$, where M is the number of the data points. We also denote the latent state variable as $\boldsymbol{x} \in \mathbb{R}^2$ since the system is one-dimensional underactuated. The latent dynamics $\boldsymbol{\Sigma}_l$ is formulated as [14, 55]

$$\boldsymbol{\Sigma}_l: \begin{cases}\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\alpha}) + \boldsymbol{w}_p, \\ \boldsymbol{q} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\beta}) + \boldsymbol{w}_o,\end{cases} \tag{4.5}$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are hyperparameters, $\boldsymbol{w}_p$ and $\boldsymbol{w}_o$ are Gaussian noises with zero mean. From $\boldsymbol{Y}$, we estimate maps $\boldsymbol{f}$ and $\boldsymbol{g}$ in (4.5) by identifying $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ through learning algorithms. Denoting $\boldsymbol{X} = \{\boldsymbol{x}\}^M \in \mathbb{R}^{2\times M}$, the estimation problem is formulated as maximizing a-posterior probability distribution $p(\boldsymbol{X}, \boldsymbol{\alpha}, \boldsymbol{\beta}|\boldsymbol{Y})$.

With the above defined dynamics $\boldsymbol{\Sigma}_{ZD}$ and $\boldsymbol{\Sigma}_l$, *our goal* of this work is to find the quantitative relationships and connections between $\boldsymbol{\Sigma}_{ZD}$ and $\boldsymbol{\Sigma}_l$.

4.3 Relationship between bipedal walking dynamics $\boldsymbol{\Sigma}_{ZD}$ and $\boldsymbol{\Sigma}_l$

We first present the results for the linear reduction latent dynamics and then for the general nonlinear learning model.

4.3.1 Local property of the two models

Walking gaits follow the virtual constraint (4.2) and therefore, data set $\boldsymbol{Y}$ satisfies this constraint. For any given data point $\boldsymbol{q}_D \in \boldsymbol{Y}$, $\boldsymbol{y}_D := \boldsymbol{h}(\boldsymbol{q}_D) = \boldsymbol{H}_0\boldsymbol{q}_D - \boldsymbol{h}_d(\boldsymbol{c}\boldsymbol{q}_D) = \boldsymbol{0}$. For $\boldsymbol{q}_0$ satisfies the virtual constraint in the neighborhood of $\boldsymbol{q}_D$, we have $\boldsymbol{y}_0 := \boldsymbol{h}(\boldsymbol{q}_0) = \boldsymbol{H}_0\boldsymbol{q}_0 - \boldsymbol{h}_d(\boldsymbol{c}\boldsymbol{q}_0) = \boldsymbol{0}$. Therefore, we obtain

$$\boldsymbol{H}_0(\boldsymbol{q}_D - \boldsymbol{q}_0) - (\boldsymbol{h}_d(\boldsymbol{c}\boldsymbol{q}_D) - \boldsymbol{h}_d(\boldsymbol{c}\boldsymbol{q}_0)) = \boldsymbol{0}.$$

Defining $\Delta\boldsymbol{q} = \boldsymbol{q}_D - \boldsymbol{q}_0$, from the above equation, we obtain the approximation

$$\boldsymbol{H}_\theta \Delta\boldsymbol{q} = \boldsymbol{0}. \tag{4.6}$$

Denoting the null space unit vector $\boldsymbol{b} \in \mathrm{null}(\boldsymbol{H}_\theta)$ and $\|\boldsymbol{b}\| = 1$, the solution of (4.6) is then written as

$$\boldsymbol{q}_D = \boldsymbol{q}_0 + \boldsymbol{b}x_d, \tag{4.7}$$

where $x_d \in \mathbb{R}$.

When a linear reduction method such as principal component analysis (PCA) is applied to $\boldsymbol{Y}$, it essentially estimates $\boldsymbol{b}$ in (4.7). Variable x_d is then the latent space variable. We formally present this result in the following lemma and the proof is given in Appendix B.1.

Lemma 4.1. *Given $\boldsymbol{Y}$ and locally approximation relationship in (4.7), the first principle axis of the PCA reduction of $\boldsymbol{Y}$ is $\boldsymbol{b}$.*

Because of the claim in Lemma 4.1, we abuse the notation by using $\boldsymbol{b}$ to denote the PCA reduction result from $\boldsymbol{Y}$ in the following discussions. To calculate x_d for any given $\boldsymbol{q}$, we use (4.7) and obtain

$$x_1 = x_d = \frac{\boldsymbol{a}^T(\boldsymbol{q} - \boldsymbol{q}_0)}{\boldsymbol{a}^T\boldsymbol{b}}, \tag{4.8}$$

where $\boldsymbol{a} \in \mathbb{R}^7$ is a constant column vector and $\boldsymbol{a}^T\boldsymbol{b} \neq \mathbf{0}$. Letting $x_2 = \dot{x}_1 = \frac{\boldsymbol{a}^T\dot{\boldsymbol{q}}}{\boldsymbol{a}^T\boldsymbol{b}}$, then

$$\dot{\boldsymbol{q}} = \boldsymbol{b}\dot{x}_1 = \boldsymbol{b}x_2. \tag{4.9}$$

To find the relationship between $\boldsymbol{x} = [x_1 \; x_2]^T$ and $\boldsymbol{\xi}$, plugging (4.3) into (4.8) and (4.9), we obtain diffeomorphic coordinate transformation

$$x_1 = \frac{1}{\boldsymbol{a}^T\boldsymbol{b}}\boldsymbol{a}^T\boldsymbol{H}_C^{-1}\begin{bmatrix}\boldsymbol{h}_d(\xi_1)\\ \xi_1\end{bmatrix}, \tag{4.10}$$

$$x_2 = \frac{1}{\boldsymbol{a}^T\boldsymbol{b}}\boldsymbol{a}^T\boldsymbol{H}_D^{-1}\begin{bmatrix}\mathbf{0}\\ \xi_2\end{bmatrix} \tag{4.11}$$

and the dynamics of $\boldsymbol{x}$ are

$$\dot{x}_1 = x_2, \; \dot{x}_2 = f_1(\boldsymbol{x}). \tag{4.12}$$

where $f_1(\boldsymbol{x})$ is obtained by transforming the zero dynamics in $\boldsymbol{\xi}$ coordinate into $\boldsymbol{x}$ coordinate.

4.3.2 Model relationship with nonlinear reduction

In this section, we focus on the relation between GPDM [14] and zero dynamics. Unlike parametric models, model $\boldsymbol{\Sigma}_l$ marginalizes the parameters and optimizes them with respect to the latent variable $\boldsymbol{x}$ and hyperparameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. The objective function of $\boldsymbol{\Sigma}_l$ is the posterior probability of latent states $\boldsymbol{X}$ and parameters given $\boldsymbol{Y}$ as

$$p(\boldsymbol{X}, \boldsymbol{\alpha}, \boldsymbol{\beta} | \boldsymbol{Y}) \propto p(\boldsymbol{Y} | \boldsymbol{X}, \boldsymbol{\beta}) p(\boldsymbol{X} | \boldsymbol{\alpha}) p(\boldsymbol{\alpha}) p(\boldsymbol{\beta}). \tag{4.13}$$

From (4.3), we re-write $\boldsymbol{q}$ by $\boldsymbol{\xi}$ of (4.4) as

$$\boldsymbol{q} = \boldsymbol{H}_C^{-1} \begin{bmatrix} h_d(\xi_1) \\ \\ \xi_1 \end{bmatrix} + \boldsymbol{n}_{ob}, \tag{4.14}$$

where $\boldsymbol{n}_{ob} \sim \mathcal{N}(\boldsymbol{0}, \sigma_y^2 \boldsymbol{I})$ is the model noise. Assuming that $\boldsymbol{Y}$ are generated from the zero dynamics (4.4) and an observation from (4.14), we apply the $\boldsymbol{\Sigma}_l$ model to $\boldsymbol{Y}$. Let us consider a case where the mapping $\xi_1 = \boldsymbol{c}\boldsymbol{q}$ is known. For each $\boldsymbol{q}_i \in \boldsymbol{Y}$ we compute $x_1^i = \boldsymbol{c}\boldsymbol{q}_i$ as the latent states and only optimize (4.13) with respect to $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. Since $\boldsymbol{X}$ is no longer an optimized variable, $\max p(\boldsymbol{X}, \boldsymbol{\alpha}, \boldsymbol{\beta} | \boldsymbol{Y})$ in (4.13) is achieved by separately maximizing $p(\boldsymbol{Y} | \boldsymbol{X}, \boldsymbol{\beta}) p(\boldsymbol{\beta})$ with respect to $\boldsymbol{\beta}$ and $p(\boldsymbol{X} | \boldsymbol{\alpha}) p(\boldsymbol{\alpha})$ with respect to $\boldsymbol{\alpha}$.

We define $\Xi = \{\boldsymbol{\xi}\}^M \in \mathbb{R}^{2 \times M}$ as the collection of M-point data $\boldsymbol{\xi}$ and $\boldsymbol{X}_1 \in \mathbb{R}^{1 \times M}$ and $\Xi_1 \in \mathbb{R}^{1 \times M}$ are the first rows of $\boldsymbol{X}$ and Ξ, respectively. Before we present the results to maximize the posterior probability discussed above, we first present the following results with proof given in Appendix B.2.

Lemma 4.2. *With $\boldsymbol{Y}$ and Ξ, the solution of the Gaussian process regression problem*

$$\min_{\boldsymbol{\beta}} -\log p(\boldsymbol{Y} | \Xi_1, \boldsymbol{\beta}) \tag{4.15}$$

with respect to $\boldsymbol{\beta}$ reconstructs the analytic mapping in (4.14).

With Lemma 4.2, we further obtain the following results.

Lemma 4.3. *With Ξ sampled from zero dynamics (4.4), the solution of the Gaussian process regression problem*

$$\min_{\boldsymbol{\alpha}} -\log p(\ddot{\xi}_1 | \xi_1, \dot{\xi}_1, \boldsymbol{\alpha}) \tag{4.16}$$

with respect to $\boldsymbol{\alpha}$ reconstructs the zero dynamics (4.4).

Proof. Applying a diffeomorphic coordinate transformation

$$\mathcal{T}_{\xi\zeta}: \begin{cases} \zeta_1 = \xi_1, \\ \zeta_2 = g_1(\xi_1)\xi_2 \end{cases} \tag{4.17}$$

to zero dynamics $\boldsymbol{\Sigma}_{ZD}$ in (4.4), we obtain

$$\boldsymbol{\Sigma}'_{ZD}: \begin{cases} \dot{\zeta}_1 = \zeta_2, \\ \dot{\zeta}_2 = g_1(\zeta_1)g_2(\zeta_1, \frac{\zeta_2}{g_1(\zeta_1)}) + \frac{\zeta_2^2}{g_1(\zeta_1)}\frac{\partial g_1(\zeta_1)}{\partial \zeta_1}. \end{cases} \tag{4.18}$$

Because only the trajectory of $\xi_1 = \zeta_1$ (i.e., $\boldsymbol{\Xi}_1$) is accessible, we introduce and define the vector field $\zeta_2(t) = \dot{\xi}_1$ to access the input and output data $\boldsymbol{x} = \boldsymbol{\zeta} = [\zeta_1 \ \zeta_2]^T$, $y = \dot{\zeta}_2 = \ddot{\xi}_1$ generated by (4.18). Following the proof of Lemma 4.2, if we minimize (4.16) to obtain the hyperparameter $\boldsymbol{\alpha}$, the mean value of the posterior predictive distribution should converge to the second equation of (4.18). □

Lemma 4.2 implies that the solution of problem (4.15) generates the relationship as (4.14), while the result in Lemma 4.3 implies that the solution of the regression problem (4.16) generates the relationship as (4.18). Using the results in Lemmas 4.2 and 4.3, we are now ready to present the results for the latent state dynamics in (4.5).

Theorem 4.1. *For the optimization problem*

$$\min_{\boldsymbol{X},\boldsymbol{\beta},\boldsymbol{\alpha}} -\log p(\boldsymbol{Y}|\boldsymbol{X}_1, \boldsymbol{\beta}) - \log p(\boldsymbol{X}|\alpha) + \frac{1}{2\sigma^2}\|\boldsymbol{X}_1 - \boldsymbol{c}\boldsymbol{Y}\|^2, \tag{4.19}$$

$p(\boldsymbol{X}|\boldsymbol{\alpha}) = p(\ddot{x}_1|x_1, \dot{x}_1, \boldsymbol{\alpha})$*, the solution is achieved when the parameters satisfy the following:*

1. $\boldsymbol{X}_1 = \boldsymbol{\Xi}_1$*;*

2. *The mean value of the posterior predictive distribution of the learned latent dynamics (determined by $\boldsymbol{\alpha}$) is given by $\dot{\zeta}_2$ in (4.18); and*

3. *The mean value of the posterior predictive distribution of the learned mapping function is given by (4.14).*

Proof. The third term of (4.19) is minimized by $\boldsymbol{X}_1 = \Xi_1$ and the problem then degenerates to $\min_{\boldsymbol{\beta}} - \log p(\boldsymbol{Y}|\Xi_1, \boldsymbol{\beta})$ and $\min_{\boldsymbol{\alpha}} - \log p(\Xi|\boldsymbol{\alpha})$ simultaneously. As shown in Lemma 4.2, the former is minimized by $\boldsymbol{\beta}$ that corresponds to a posterior predictive distribution with the mean function as expression (4.14). The latter is then minimized by $\boldsymbol{\alpha}$ that corresponds to a posterior predictive distribution with the mean function as (4.18). This completes the proof. □

The results in Theorem 4.1 implies that the latent dynamics is indeed the same as zero dynamics (4.18) in the new coordinate (ζ_1, ζ_2) with $x_1 = \zeta_1 = \xi_1$. The vector field $(x_1, \dot{x}_1)$ is given by (4.18). The relationships among the analytical model (4.1), zero dynamics $\boldsymbol{\Sigma}_{ZD}$ in (4.4) and latent dynamics $\boldsymbol{\Sigma}_l$ in (4.5) are illustrated in Figure. 4.2. It is clear that the relationship between $\boldsymbol{\Sigma}_{ZD}$ and $\boldsymbol{\Sigma}_l$ are the coordinate transformation $\mathcal{T}_{\xi\zeta}$.

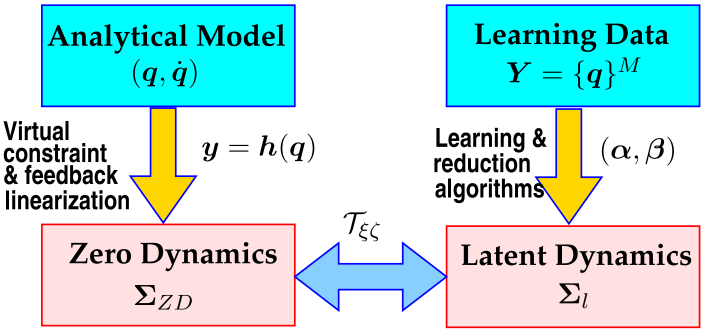

Analytical Model
$(\boldsymbol{q}, \dot{\boldsymbol{q}})$
Learning Data
$\boldsymbol{Y} = \{\boldsymbol{q}\}^M$
Virtual constraint & feedback linearization
$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{q})$
Learning & reduction algorithms
$(\boldsymbol{\alpha}, \boldsymbol{\beta})$
$\mathcal{T}_{\xi\zeta}$
Zero Dynamics
$\boldsymbol{\Sigma}_{ZD}$
Latent Dynamics
$\boldsymbol{\Sigma}_l$

Figure 4.2: Schematic diagram of the relationship between the analytical and learning models.

Remark 4.1. *In practice, only $\boldsymbol{Y}$ is measured and obtained and vector $\mathbf{c}$ is not readily available for the data-driven modeling approach. In the physical model design, the choice of $\mathbf{c}$ is manually determined and tuned to satisfy that $\theta = \mathbf{c}\boldsymbol{q}$ is monotonically increasing function during the swing phase and matrix $\boldsymbol{H}_C$ is invertible [11]. To use* *the learning algorithms (4.19), we modify the objective function as*

$$\min_{\boldsymbol{X},\boldsymbol{\beta},\boldsymbol{\alpha},\hat{\boldsymbol{c}}} -\log p(\boldsymbol{Y}|\boldsymbol{X}_1,\boldsymbol{\beta}) - \log p(\boldsymbol{X}|\alpha) + \frac{1}{2\sigma^2}\|\boldsymbol{X}_1 - \hat{\boldsymbol{c}}\boldsymbol{Y}\|^2. \tag{4.20}$$

In the above modification, vector $\hat{\boldsymbol{c}}$ is introduced as an optimized variable and similar to (4.19), the third term can be interpreted as $-\log p(\boldsymbol{X}|\hat{\boldsymbol{c}},\boldsymbol{Y})$, where $p(\boldsymbol{X}|\hat{\boldsymbol{c}},\boldsymbol{Y}) = \mathcal{N}(\hat{\boldsymbol{c}}\boldsymbol{Y},\sigma^2\boldsymbol{I})$. The mean value $\hat{\boldsymbol{c}}\boldsymbol{Y}$ can also be interpreted as a label of latent variable [23,56].

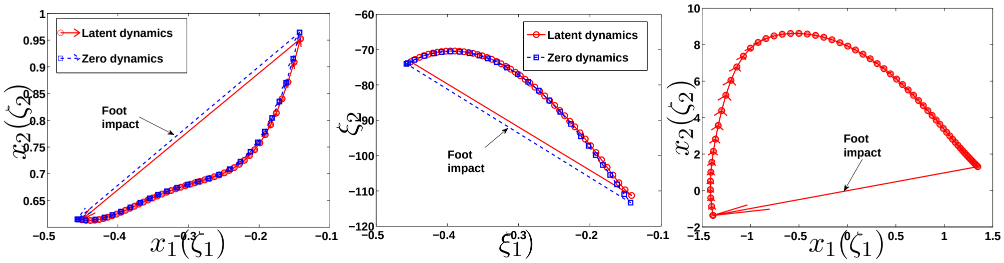
4.3.3 Application example:learning virtual constraints

We briefly describe an application of Theorem 4.1 for obtaining the virtual constraint (4.2) through the learning approach. No explicit virtual constraint is given for the learning model and instead the observation mapping in (4.5) implicitly determines a constraint for $\boldsymbol{q}$. Because the learning algorithm in Theorem 4.1 enforces $x_1 = \boldsymbol{c}\boldsymbol{q}$, the constraint being enforced is $\boldsymbol{q} = \boldsymbol{g}(\hat{\boldsymbol{c}}\boldsymbol{q},\boldsymbol{\beta}) + \boldsymbol{w}_o$. This constraint is parameterized by $\boldsymbol{\beta}$ and $\hat{\boldsymbol{c}}$. We therefore propose an approach to quantify different gaits in latent space and the metrics. Indeed, we express the virtual constraints for different gaits with the same shared parameter $\boldsymbol{\beta}$ and consider to use parameter $\hat{\boldsymbol{c}}$ to capture and characteristics of human gaits. We will demonstrate such application examples in the next section.

4.4 Experiments

With the collected human walking joint angles, we identify the analytical model parameters and optimize the gait profile parameters to fit the experiments [2]. We also build the PCA model (4.12) as well as the latent dynamics model (4.18). When building the latent dynamics by (4.19), we use the value of $\boldsymbol{c}$ for the analytical model (e.g., zero dynamics).

With the optimized model parameters for both zero dynamics $\boldsymbol{\Sigma}_{ZD}$ (4.4) and latent

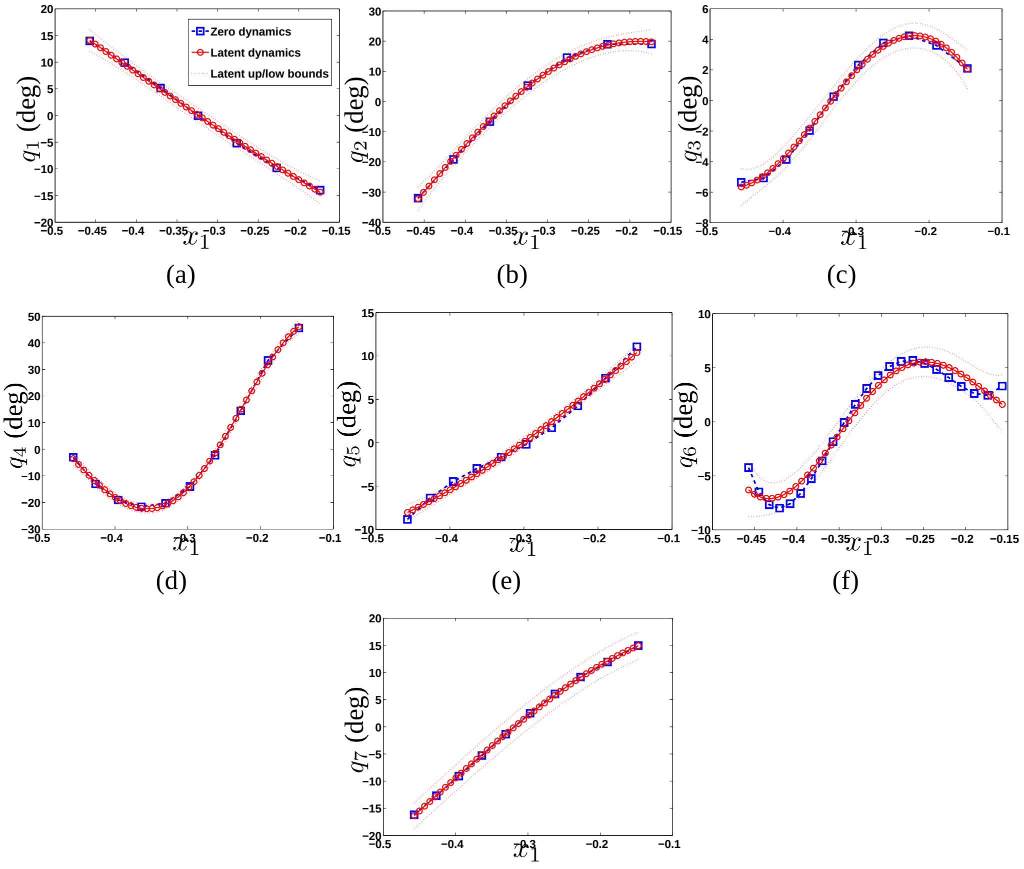


(a) (b) (c)

Figure 4.3: (a) The phase portraits of the zero dynamics and latent dynamics in coordinate (x_1, x_2). (b) The phase portraits of the zero dynamics and latent dynamics in coordinate (ξ_1, ξ_2). (c) The phase portrait of the PCA-based reduction model in coordinate (x_1, x_2). The arrows in the figures show the flow direction (positive time evolution.)

dynamics $\mathbf{\Sigma}_l$ (4.5), Figure. 4.3(a) shows the phase portraits of the two dynamic systems in coordinates (x_1, x_2) during one-single stance motion. For zero dynamics $\mathbf{\Sigma}_{ZD}$, coordinate transformation $\mathcal{T}_{\xi\zeta}$ (4.17) is used to convert into (ζ_1, ζ_2), namely, (x_1, x_2). The plots in Figure. 4.3(a) clearly show that both dynamics generate the same phase portraits. Since most existing work present the zero dynamics in (ξ_1, ξ_2) (e.g., [52]), we also transfer the dynamics from (x_1, x_2) and plot its phase portrait in (ξ_1, ξ_2) as shown in Figure. 4.3(b). For comparison purpose, Figure. 4.3(c) shows the phase portrait of the PCA-based dynamics (4.12) in (x_1, x_2).

While Figure. 4.3 demonstrates the comparison of the state dynamics, Figure. 4.4 presents the comparison results of the output mappings. In Figure. 4.4, the joint angle predictions $\boldsymbol{q}$ are plotted over the state variable x_1. For latent dynamics, the mean value of the posterior predictive distribution of the observation mapping is plotted with the red circle lines and the upper and lower bounds (one standard deviation) are plotted as the red dotted lines. For zero dynamics $\mathbf{\Sigma}_{ZD}$, we again transfer its states (ξ_1, ξ_2) into (ζ_1, ζ_2) (i.e., (x_1, x_2)). These plots confirm that the output mappings of two dynamics behave similar and generate close predictions of walking gaits. The results shown in Figures. 4.3 and 4.4 demonstrate that the latent dynamics indeed capture the same properties of the zero dynamics.



(a) (b) (c)

(d) (e) (f)

(g)

Figure 4.4: (a)-(g): Joint angles (q_1 to q_7) predictions (vs. state variable x_1) by the zero dynamics and the latent dynamics. The latent dynamic model prediction results are also marked with the upper and lower bounds of 2 standard deviations as the dotted lines in each figure.

The above results are obtained by using the known gait profile parameter $\boldsymbol{c}$ same as the zero dynamics. We here present the application example to instead use optimization (4.20) to estimate the value $\hat{\boldsymbol{c}}$. Moreover, the value of $\hat{\boldsymbol{c}}$ can be used to characterize and quantify the gait profile. To demonstrate this application, we collect joint angles of three different gaits: gait $\boldsymbol{Y}_1$ is obtained from data set in [44], gait $\boldsymbol{Y}_2$ is generated by a perturbed virtual constraint profile from $\boldsymbol{Y}_1$, and gait $\boldsymbol{Y}_3$ is collected by the human walking experiments. Figure. 4.5 shows the joint angles profiles for all seven joints. Obviously, the three gaits have different profiles but $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$ have similar trends.

We use (4.20) to obtain the latent dynamics and gait profile parameter $\hat{\boldsymbol{c}}$ while

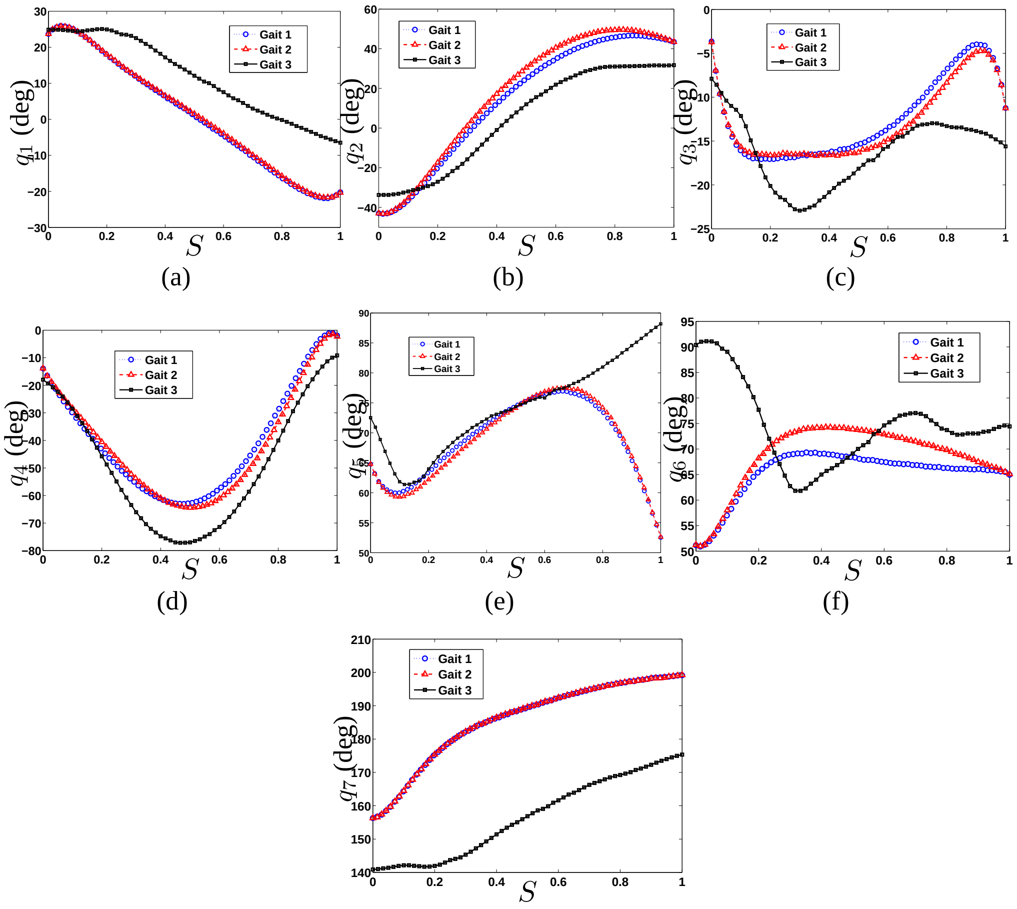


Figure 4.5: (a)-(g): Joint angles (q_1 to q_7) comparison among three gaits. For ease of comparison, the time is scaled to 1. Gait 1 takes 0.72 sec per step. Gait 2 takes 0.71 sec per step. Gait 3 takes 0.61 sec per step.

keeping parameter $\boldsymbol{\beta}$ as the same shared value (still as an optimized variable). We denote the optimized parameter $\hat{\boldsymbol{c}}_i$, $i = 1, 2, 3$, for gaits $\boldsymbol{Y}_i$, respectively. The values of parameters $\hat{\boldsymbol{c}}_i$ are listed in Table 4.1. To demonstrate the correlations of the latent dynamics for these three gaits, we introduce the distance metrics

$$d_{ij}^2 := \|\hat{\boldsymbol{c}}_i - \hat{\boldsymbol{c}}_j\|_2 \text{ and } d_{ij}^\infty := \|\hat{\boldsymbol{c}}_i - \hat{\boldsymbol{c}}_j\|_\infty, \; i, j = 1, 2, 3.$$

We calculate and list these metrics in Table 4.1. From these metrics, we clearly observe that $d_{12}^2 \ll d_{13}^2$ and $d_{12}^2 \ll d_{23}^2$ and this implies that gaits $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$ are similar and they are different from gait $\boldsymbol{Y}_3$. Similarly, the values of d_{12}^∞, d_{13}^∞ and d_{23}^∞ show the same trends. Figure. 4.6 further shows the the corresponding phase portraits of the latent dynamics. The difference among them is demonstrated consistently.

Table 4.1: Gait profile comparison using vector $\boldsymbol{c}$

Gaits	Gait profile parameter $\hat{\boldsymbol{c}}_i$	d^2_{ij}	d^∞_{ij}
Y_1	$[-0.4\ 1.4\ -0.8\ 0.4\ -1.0\ 0.3\ 0.8]$	$d^2_{12} = 0.26$	$d^\infty_{12} = 0.2$
Y_2	$[-0.6\ 1.4\ -0.7\ 0.5\ -1.1\ 0.3\ 0.7]$	$d^2_{23} = 1.37$	$d^\infty_{23} = 0.9$
Y_3	$[-0.6\ 1.4\ -0.9\ 0.2\ -0.2\ -0.6\ 0.5]$	$d^2_{13} = 1.31$	$d^\infty_{13} = 0.9$

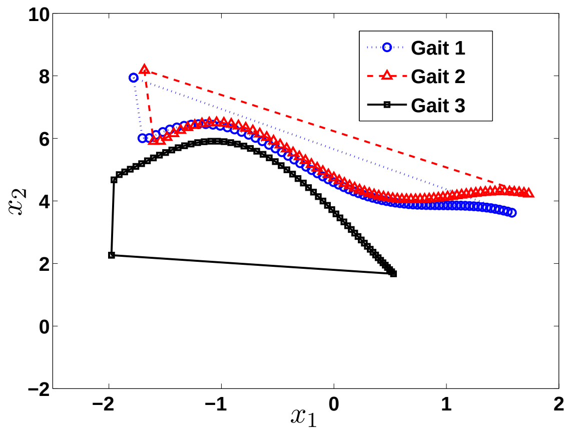


Figure 4.6: The phase portrait comparison of latent dynamics for three different gaits.

4.5 Summary

We presented relationship between the physical principle-based dynamic model and the data-driven latent dynamics for human walking locomotion. We analyzed and demonstrated that the equivalence of the zero dynamics (from the physical model) and the latent dynamics (from the manifold learning model) for the underactuated human walking gaits. The mapping functions between these two models were derived and their correspondence were explicitly presented and discussed. Human experiments were also conducted to validate and demonstrate the findings. Finally, we presented the application examples to use the new findings.

Chapter 5

Gaussian Processes based Balance Robot Control

5.1 Introduction

This chapter studies the learning-based modeling and control of underactuated balance robots [13]. The cart-pole system, Furuta pendulum and autonomous bicycles are examples of underactuated balance systems and the control goal of those systems is to balance the pendulum/bicycle while keeping the bases to follow the desired trajectories. Bipedal walkers are also a class of underactuated balance robots because the actuated joint angles are commanded to follow desired trajectories to form certain gaits, while the unactuated floating base should be kept stable across different steps. The control goal of underactuated balance robots is to achieve both external (actuated) subsystem trajectory tracking and internal (unactuated) subsystem balancing around unstable equilibrium point with limited control authority. The balancing and tracking tasks are often intertwined and sometimes competing each other for controlled performance. For example, control of an autonomous motorcycle requires the path-following task while balancing the platform [38]. It is common that the path-following and the platform balancing are two competing tasks as shown in [57]. Moreover, the intrinsic unstable internal or zero dynamics add additional challenges in control systems design.

For a nonlinear non-minimum phase system, no analytical causal controller can achieve exactly output tracking while maintaining the internal stability [12]. An innovative control design of underactuated balance robots is to take advantages of the interaction between the external and internal subsystems. In [13], an external/internal convertible (EIC) dynamic structure is explored to partition the underactuated balance systems into an external subsystem for path-following task and an internal subsystem for balance task. An elegant causal control design is proposed in [13] to guarantee the exponential convergence of the actual output to a tube containing the desired output and the internal state to a balance equilibrium manifold (BEM). The BEM concept plays a central role in the EIC-based control design and recently is used to capture and quantify the human rider riding skills [57].

Despite of the mathematical elegance of the BEM concept, its applicability needs the rigid EIC form of the dynamics model structure. Recently, machine learning-based controller design has shown potential to save effort of understanding individual system, handle un-modeled dynamic effects and achieve superior performance than first principle model-based controller design. Gaussian processes (GPs) machine learning [58] is widely applied to robot modeling and control. When they are applied to learn mechanical dynamical systems, GPs take the current robot states and control inputs as the learning inputs and the derivatives of robot states as the outputs. Gaussian processes predict differentiable and closed-form mean value and covariance of output distribution, and this property makes it easily integrated with optimization-based controller design such as model predictive control (MPC) and reinforcement learning [59–64]. Compared to other model learning approaches such as Artificial Neural Network (ANN) and Support Vector Machine (SVM), GPs provide predictive covariance which can be used as a quantitative evaluation of model uncertainty for robot controller design [59, 61, 63, 64].

MPC has been widely applied to preview-based online trajectory tracking [65]. At each control step, the controller online solves the optimal input sequence that minimizes the objective function and uses the first element of the optimal input sequence as the actuation. However, computational cost is the bottleneck preventing applying MPC to high-dimensional state space dynamics. In this chapter, we adopt singular perturbation technique to reduce the dimension of state space that the MPC has to consider about. By transforming the underactuated robot dynamics into the EIC form, the internal dynamics is feedback linearized and its convergent rate is assigned to be much faster than that of external dynamics. The internal states can be treated as control input to directly control the external state. Taking the cart-pole system as an intuitive example, this implies that through feedback linearization the pendulum angle is directly controlled and the cart position is controlled with the pendulum angle as input. We adopt MPC as an online feedback planner to design the internal state trajectory and an inverse dynamics controller to stabilize the designed trajectory. Since the robot model is learned from experiment data with GPs model, the proposed planning and control framework takes advantage of the predicted Gaussian distribution. The MPC trajectory planner takes the model uncertainty of the inverse dynamics model into consideration to avoid planning trajectories that the inverse dynamics is unable to stabilize.

The contribution of this work lies in three aspects. First, we extend the EIC form and the inversion based BEM solver proposed in [13] to an MPC based trajectory planner and an inverse dynamics trajectory stabilizer. The proposed planning and control framework takes advantage of MPC's numerical computation and save the effort of offline symbolic computation of function inversion. Therefore, this framework can be easily applied to models obtained from non-parametric machine learning approach such as GPs whose inverse function is difficult to obtain. Second, the proposed learning model-based controller is data efficient. Facing the difficult problem of stabilizing a non-minimum phase unstable system to achieve both internal subsystem balanced and external subsystem tracking desired trajectory, most existing approaches rely on either prior knowledge of the physical model or successful demonstration data from human or other baseline controllers. The proposed approach can take random excitation data for model learning and then achieve successful balancing while tracking task. The system only needs to be excited enough to reveal the open-loop system dynamics. The model is learned without any prior knowledge or successful demonstration. Finally, our planning and control framework explicitly incorporates the GPs predicted model uncertainty to enhance control robustness, that is, the MPC planner takes the inverse dynamics modeling uncertainty into consideration to generate the planned trajectory. We demonstrate the proposed planning and control framework by experiments on a Furuta pendulum.

5.2 Related works

Getz [13] proposed to partition the underactuated balance system into external and internal subsystems. The control goal of the external subsystem is to track the desired trajectory while that of the internal subsystem is to maintain stable around an unstable equilibrium point. The EIC form of the original dynamic model was used to describe the coupling effect of the external and internal subsystems. The fact that the external tracking trajectory determines the internal subsystem equilibrium point gives rise to the concept of BEM. Dynamic inversion technique is used in [13] to compute the BEM in order to design the integrated controllers. The proposed controller is proven to be asymptotically stable to a tube around the desired trajectory. The work in [66] formulates the EIC form in the multi-time-scale structure based on the singular perturbation theory. It also extends the result in [13] to output feedback with extended high-gain observers. The work in [67] extends the BEM approach to learning model-based control. GPs are adopted to identify the system dynamics. However, because the flexibility of GPs, the actual plant dynamics structure cannot be successfully captured even though the prediction error is small. To increase the data efficiency to learn the actual dynamic structure, the robot manipulator-based model form with symmetric inertia matrix is enforced in [67] as prior knowledge. The BEM approach with learned model demonstrates worse tracking performance than that with physical-principle model though the former results in smaller prediction error. We speculate that the BEM is not accurately captured by the learning model approach due to the flexible structure of GPs. The fact that dynamic inversion of the learning model does not accurately identify the BEM motivates the MPC approach proposed in this chapter.

Learning inverse dynamics has been demonstrated in robot control applications. The works in [68] and [69] adopt inverse dynamics controller using the global and local GPs regression models, respectively. The learned model predicts control inputs using the robot current states and the desired derivative of robot states. Even though the GPs provide predictive distribution of the control input, only the mean value of the Gaussian distribution is used and taken as the control input. The work in [70] proposes a GPs-based inverse dynamics control law with the feedback gain incorporating the variance of the predictive distribution, i.e., using low gain if the learned model is considered with high-confidence and otherwise high gain. Both [70] and [71] give theoretical guarantee of stability of GPs-based inverse dynamics controller. In [72], the authors applied deep neural network (DNN) to learn inverse dynamics to achieve impromptu trajectory tracking. The inverse dynamics controller however cannot be directly applied to underactuated non-minimum phase robot control due to the unstable internal dynamics. The work in [73] achieved non-minimum phase robot impromptu trajectory tracking for cart-pole system and quadrotor system by learning a stable, approximated inverse of a non-minimum phase baseline system from its input-state data. The algorithm first runs a baseline controller, usually a linear controller, to achieve the stabilization task and collect input-state data for DNN training. In the training phase, the inverse model of the stabilized baseline system is learned. In the testing phase, given the desired trajectory, the learned DNN model computes a reference trajectory for the baseline system. Under this learning-based inversion controller, the tracking performance is enhanced compared to the baseline system.

Optimization-based controllers such as MPC and reinforcement learning have been widely applied in controlling underactuated robot system such as cart-pole system, blimps and helicopters. In [60], the learning model captures the difference between the collected acceleration data of the blimp and the prediction from the physical-principle model so that the learning approach does not have to build the blimp model from scratch. In [74], the helicopter model is learned with maneuvers and trajectories that have been successfully demonstrated by human expert in order to avoid learning a model for the whole state space. By either adding prior knowledge of the robot model or learning from expert demonstration, collected data can be efficiently trained. The work in [61] however does not assume task-specific prior knowledge but takes advantage of the probabilistic nature of Gaussian process to achieve efficient learning. A lot of GP-based controller designs have explicitly taken advantage of the predicted Gaussian distribution to achieve robust control. In [59] [61] [64], the objective function is the expected tracking error over the prediction horizon with the consideration of the variance of the predictive distribution. In [63], the predictive variance is used to shrink the feasible region for the predictive mean value of the trajectory.

5.3 Notations

Vectors $\boldsymbol{\alpha}$ are denoted with bold lower-case characters. Matrices $\boldsymbol{A}$ are denoted with bold capital characters. Identity matrix is denoted as $\boldsymbol{I}_n$ with its dimension n as subscript. Approximately estimated values are denoted with tilde as $\tilde{\boldsymbol{\alpha}}$. Natural number is denoted as $\mathbb{N}$. Real number is denoted as $\mathbb{R}$. Positive real value is denoted as $\mathbb{R}^+$. n dimensional real valued vector space is denoted as $\mathbb{R}^n$. The sign "$:=$" represents the definition equality. $\min(\cdot,\cdot)$ and $\max(\cdot,\cdot)$ denotes the functions that take the minimum and maximum value of the arguments in the parenthesis, respectively. The smallest eigenvalue of matrix $\boldsymbol{A}$ is denoted by $\lambda_{\min}(\boldsymbol{A})$ and the largest by $\lambda_{\max}(\boldsymbol{A})$. The matrix norm is defined as $||\boldsymbol{A}|| = [\lambda_{\max}(\boldsymbol{A}^T\boldsymbol{A})]^{\frac{1}{2}}$. The vector norm is defined as $||\boldsymbol{\alpha}|| = \sqrt{\boldsymbol{\alpha}^T\boldsymbol{\alpha}}$. The matrix $\boldsymbol{P}$ induced vector norm is denoted as $||\boldsymbol{\alpha}||^2_{\boldsymbol{P}} = \boldsymbol{\alpha}^T\boldsymbol{P}\boldsymbol{\alpha}$. $\mathrm{tr}(\boldsymbol{A})$ is the trace of matrix $\boldsymbol{A}$. $\det(\boldsymbol{A})$ denotes the determinant of matrix $\boldsymbol{A}$. The expression $y = O(x^n)$ means that there exist constants $z_i > 0, i = 0, 1, ..., n$ such that $|y| \leq |\sum_{i=0}^{n} z_i x^i|$.

The expression $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ represents that $\boldsymbol{x}$ is a random variable satisfying Gaussian distribution with mean value $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. The expression $\dot{\boldsymbol{x}} \sim \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$ means that $\dot{\boldsymbol{x}}$ is a random variable satisfying a distribution either because $\boldsymbol{x}$ and $\boldsymbol{u}$ are random variables or because $\boldsymbol{f}$ is a Gaussian process modeled random function or both. The expectation operator is denoted as $\mathbb{E}$. $\boldsymbol{\Pi}$ denotes a probabilistic event and its probability is written as $\Pr\{\boldsymbol{\Pi}\}$.

In the discrete MPC presentation, $k \in \mathbb{N}$ is used to denote the current time step. $k + i$ with $i \in \mathbb{N}$ is used to denote the i-step forward time moment. A variable $\boldsymbol{\alpha}^0$ with a “0” superscript denotes the optimal value of the design parameter $\boldsymbol{\alpha}$.

5.4 Balance system control background

5.4.1 Underactuated system and BEM

We consider controlling an underactuated mechanical system modeled by Lagrangian dynamical model:

$$\boldsymbol{D}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{H}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{B}(\boldsymbol{q})\boldsymbol{u}. \tag{5.1}$$

In (5.1), $\boldsymbol{q} \in \mathbb{R}^{m+n}$ is the configuration state of the system. $\boldsymbol{u} \in \mathbb{R}^m$ is the control input. $\boldsymbol{D}(\boldsymbol{q})$ is the inertial matrix and $\boldsymbol{H}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ typically contains the centripetal term the Coriolis term and the gravitational term. $\boldsymbol{B}(\boldsymbol{q})$ is the input mapping matrix. The configuration space $\boldsymbol{q}$ can be decomposed into actuated subspace $\boldsymbol{\theta}_1 \in \mathbb{R}^m$ and unactuated subspace $\boldsymbol{\alpha}_1 \in \mathbb{R}^n$, i.e. $\boldsymbol{q} = [\boldsymbol{\theta}_1^T, \boldsymbol{\alpha}_1^T]^T$. Here it is assumed that $m \geq n$. We also define actuated generalized velocity $\boldsymbol{\theta}_2 := \dot{\boldsymbol{\theta}}_1$ and unactuated generalized velocity $\boldsymbol{\alpha}_2 := \dot{\boldsymbol{\alpha}}_1$ so that $\dot{\boldsymbol{q}} = [\boldsymbol{\theta}_2^T, \boldsymbol{\alpha}_2^T]^T$. Equation (5.1) is then partitioned into actuated and unactuated subsystems:

$$\begin{bmatrix} \boldsymbol{D}_{11}(\boldsymbol{q}) & \boldsymbol{D}_{12}(\boldsymbol{q}) \\ \boldsymbol{D}_{21}(\boldsymbol{q}) & \boldsymbol{D}_{22}(\boldsymbol{q}) \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{\theta}}_2 \\ \dot{\boldsymbol{\alpha}}_2 \end{bmatrix} + \begin{bmatrix} \boldsymbol{H}_1(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\ \boldsymbol{H}_2(\boldsymbol{q}, \dot{\boldsymbol{q}}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{B}_1(\boldsymbol{q}) \\ 0_{n\times m} \end{bmatrix} \boldsymbol{u}, \tag{5.2}$$

where $\boldsymbol{B}_1(\boldsymbol{q}) \in \mathbb{R}^{m\times m}$ is full rank. By inverting the mass matrix $\boldsymbol{D}(\boldsymbol{q})$ in (5.2), we obtain

$$\begin{bmatrix} \dot{\boldsymbol{\theta}}_2 \\ \dot{\boldsymbol{\alpha}}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{D}_{11}(\boldsymbol{q}) & \boldsymbol{D}_{12}(\boldsymbol{q}) \\ \boldsymbol{D}_{21}(\boldsymbol{q}) & \boldsymbol{D}_{22}(\boldsymbol{q}) \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{B}_1(\boldsymbol{q})\boldsymbol{u} - \boldsymbol{H}_1(\boldsymbol{q}, \dot{\boldsymbol{q}}) \\ -\boldsymbol{H}_2(\boldsymbol{q}, \dot{\boldsymbol{q}}) \end{bmatrix}. \tag{5.3}$$

In this work, we are interested in identifying the unknown physical model for control application. Therefore, a general version of (5.3) is formulated as:

$$\begin{cases} \dot{\boldsymbol{\theta}}_1 & = \boldsymbol{\theta}_2 \\ \dot{\boldsymbol{\theta}}_2 & = \boldsymbol{f}_\theta(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{u}) \\ \dot{\boldsymbol{\alpha}}_1 & = \boldsymbol{\alpha}_2 \\ \dot{\boldsymbol{\alpha}}_2 & = \boldsymbol{f}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{u}), \end{cases} \tag{5.4}$$

where $\boldsymbol{\theta} = [\boldsymbol{\theta}_1^T, \boldsymbol{\theta}_2^T]^T$ and $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1^T, \boldsymbol{\alpha}_2^T]^T$. The control goal of underactuated balance system is to force the subsystem $\boldsymbol{\theta}$ to track desired trajectory $\boldsymbol{\theta}_d = [\boldsymbol{\theta}_{d1}^T, \boldsymbol{\theta}_{d2}^T]^T$ with $\boldsymbol{\theta}_{d2} = \dot{\boldsymbol{\theta}}_{d1}$, while the subsystem $\boldsymbol{\alpha}$ to be balanced around unstable equilibrium point. For this reason, $\boldsymbol{\theta}$ subsystem is referred as the external subsystem and $\boldsymbol{\alpha}$ subsystem the internal subsystem.

In (5.4), the external and internal subsystems are coupled in both directions. Getz [13] proposes the EIC form in order to simplify this coupling effect in only one direction. Let

$$\boldsymbol{v} = \boldsymbol{f}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{u}) \tag{5.5}$$

and then the internal subsystem is feedback linearized as $\dot{\boldsymbol{\alpha}}_2 = \boldsymbol{v}$. Because $\boldsymbol{v} \in \mathbb{R}^n$ and $\boldsymbol{u} \in \mathbb{R}^m$ with $n \leq m$, only a subspace of $\boldsymbol{u}$ is obtained by inverting (5.5). Letting

$\boldsymbol{u} = [\boldsymbol{u}_d^T, \boldsymbol{u}_f^T]^T$ with $\boldsymbol{u}_d \in \mathbb{R}^n$ and $\boldsymbol{u}_f \in \mathbb{R}^{m-n}$, $\boldsymbol{u}_d$ is the subspace that depends on the internal subsystem according to the inverse dynamics model

$$\boldsymbol{u}_d = \boldsymbol{f}_\alpha^{-1}(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f), \tag{5.6}$$

while $\boldsymbol{u}_f$ can be freely designed. Therefore

$$\boldsymbol{u}(\boldsymbol{v}, \boldsymbol{u}_f) = [\boldsymbol{u}_d(\boldsymbol{v}, \boldsymbol{u}_f)^T, \boldsymbol{u}_f^T]^T. \tag{5.7}$$

The dynamics (5.4) under the inverse dynamics controller (5.7) is rewritten as

$$\begin{cases} \dot{\boldsymbol{\theta}}_1 &= \boldsymbol{\theta}_2 \\ \dot{\boldsymbol{\theta}}_2 &= \boldsymbol{f}_\theta[\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{u}(\boldsymbol{v}, \boldsymbol{u}_f)] \\ \dot{\boldsymbol{\alpha}}_1 &= \boldsymbol{\alpha}_2 \\ \dot{\boldsymbol{\alpha}}_2 &= \boldsymbol{v}. \end{cases} \tag{5.8}$$

In (5.8), internal subsystem has been feedback linearized and therefore is directly controlled by $\boldsymbol{v}$ and not affected by external subsystem. External subsystem is affected by both inputs $\boldsymbol{u}_f$ and $\boldsymbol{v}$.

Since $\boldsymbol{\alpha}$ dynamics is feedback linearizable, we design a proportional-differential (PD) controller to force $\boldsymbol{\alpha}$ to converge to any desired trajectory $\boldsymbol{\alpha}_d = [\boldsymbol{\alpha}_{d1}^T, \boldsymbol{\alpha}_{d2}^T]^T$, where $\boldsymbol{\alpha}_{d2} = \dot{\boldsymbol{\alpha}}_{d1}$, namely,

$$\boldsymbol{v} = \dot{\boldsymbol{\alpha}}_{d2} - \frac{k_d}{\epsilon}(\boldsymbol{\alpha}_2 - \boldsymbol{\alpha}_{d2}) - \frac{k_p}{\epsilon^2}(\boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_{d1}). \tag{5.9}$$

The controller (5.9) temporarily ignores the tracking task for the external subsystem $\boldsymbol{\theta}$. Here $\epsilon \in \mathbb{R}^+$ is the singular perturbation parameter. $k_p, k_d \in \mathbb{R}^+$ are constant control gains. To enforce the external subsystem trajectory tracking task, an intuitive idea is to design the desired internal trajectory $\boldsymbol{\alpha}_d(\boldsymbol{\theta}_d, \boldsymbol{\theta})$ to be dependent on $(\boldsymbol{\theta}_d, \boldsymbol{\theta})$ so that the external subsystem converges to desired output $\boldsymbol{\theta}_d$. BEM is introduced by [13] to identify such dependency. BEM is found by solving $\boldsymbol{\alpha}_{d0}(\boldsymbol{\theta}_d, \boldsymbol{\theta}) = [\boldsymbol{\alpha}_{d1}^T(\boldsymbol{\theta}_d, \boldsymbol{\theta}), \boldsymbol{\alpha}_{d2}^T =$

$\mathbf{0}^T]^T$ from

$$\boldsymbol{f}_\theta(\boldsymbol{\theta}, \boldsymbol{\alpha}_{d0}(\boldsymbol{\theta}_d, \boldsymbol{\theta}), \mathbf{0}^T) = \dot{\boldsymbol{\theta}}_{d2} - k_d(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_{d2}) - k_p(\boldsymbol{\theta}_1 - \boldsymbol{\theta}_{d1}) \tag{5.10}$$

with the function inversion technique. By using BEM design, the internal subsystem $\boldsymbol{\alpha}$ converges to $\boldsymbol{\alpha}_{d0}(\boldsymbol{\theta}_d, \boldsymbol{\theta})$, meanwhile the external subsystem $\boldsymbol{\theta}$ converges to a tube containing $\boldsymbol{\theta}_d$.

It is shown in [67] that the result of inverting (5.10) is not accurate when $\boldsymbol{f}_\theta$ is represented by learning model. In this chapter, we apply MPC to obtain the desired internal trajectory $\boldsymbol{\alpha}_d(\boldsymbol{\theta}_d, \boldsymbol{\theta})$. Note that we cannot directly apply MPC to system (5.8) to solve for input $\boldsymbol{v}$ because in that case the internal subsystem may not be stable.

5.4.2 Model reduction through singular perturbation

Rather than directly applying MPC to system (5.4) or (5.8), we first apply the PD controller (5.9) to (5.8) and the closed loop dynamics are

$$\begin{cases} \dot{\boldsymbol{\theta}}_1 &= \boldsymbol{\theta}_2 \\ \dot{\boldsymbol{\theta}}_2 &= \boldsymbol{f}_\theta[\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{u}(\dot{\boldsymbol{\alpha}}_{d2} - \frac{k_d}{\epsilon}(\boldsymbol{\alpha}_2 - \boldsymbol{\alpha}_{d2}) - \frac{k_p}{\epsilon^2}(\boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_{d1}), \boldsymbol{u}_f)] \\ \dot{\boldsymbol{\alpha}}_1 &= \boldsymbol{\alpha}_2 \\ \dot{\boldsymbol{\alpha}}_2 &= \dot{\boldsymbol{\alpha}}_{d2} - \frac{k_d}{\epsilon}(\boldsymbol{\alpha}_2 - \boldsymbol{\alpha}_{d2}) - \frac{k_p}{\epsilon^2}(\boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_{d1}). \end{cases} \tag{5.11}$$

Here, (5.7) has been used to replace $\boldsymbol{u}$ in (5.8). By performing coordinate transformation $\boldsymbol{e}_\alpha = \boldsymbol{\alpha} - \boldsymbol{\alpha}_d$ or equivalently $\boldsymbol{e}_\alpha = [\boldsymbol{e}_{\alpha 1}^T \ \boldsymbol{e}_{\alpha 2}^T]^T$ with

$$\begin{cases} \boldsymbol{e}_{\alpha 1} &= \boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_{d1} \\ \boldsymbol{e}_{\alpha 2} &= \boldsymbol{\alpha}_2 - \boldsymbol{\alpha}_{d2} \end{cases} \tag{5.12}$$

to the closed-loop system, we obtain

$$\begin{cases} \dot{\boldsymbol{\theta}}_1 & = \boldsymbol{\theta}_2 \\ \dot{\boldsymbol{\theta}}_2 & = \boldsymbol{f}_\theta[\boldsymbol{\theta}, \boldsymbol{\alpha}_d + \boldsymbol{e}_\alpha, \boldsymbol{u}(\dot{\boldsymbol{\alpha}}_{d2} - \frac{k_p}{\epsilon^2}\boldsymbol{e}_{\alpha 1} - \frac{k_d}{\epsilon}\boldsymbol{e}_{\alpha 2}, \boldsymbol{u}_f)] \\ \dot{\boldsymbol{e}}_{\alpha 1} & = \boldsymbol{e}_{\alpha 2} \\ \dot{\boldsymbol{e}}_{\alpha 2} & = -\frac{k_p}{\epsilon^2}\boldsymbol{e}_{\alpha 1} - \frac{k_d}{\epsilon}\boldsymbol{e}_{\alpha 2}. \end{cases} \tag{5.13}$$

In (5.13), it is possible to solve $\boldsymbol{e}_\alpha$ analytically and substitute it into $\boldsymbol{\theta}$ dynamics. As ϵ goes to 0, $\boldsymbol{e}_{\alpha 1}$ and $\boldsymbol{e}_{\alpha 2}$ converge to $\mathbf{0}$ with the rate of $\exp(-\frac{1}{\epsilon})$, which is faster than any polynomial order of ϵ. The $\boldsymbol{\theta}$ dynamics is referred as the slow dynamics, while the $\boldsymbol{e}_\alpha$ dynamics is referred as the fast dynamics. The singular perturbation theory [75] states that $||\boldsymbol{\theta}(t,\epsilon) - \hat{\boldsymbol{\theta}}(t)|| = O(\epsilon)$ or equivalently $||\boldsymbol{\theta}(t,\epsilon) - \hat{\boldsymbol{\theta}}(t)|| \leq k|\epsilon|$ for some $k \in \mathbb{R}^+$, where $\hat{\boldsymbol{\theta}}(t) = [\hat{\boldsymbol{\theta}}_1(t)^T, \hat{\boldsymbol{\theta}}_2(t)^T]^T$ is the solution of

$$\begin{cases} \dot{\hat{\boldsymbol{\theta}}}_1 & = \hat{\boldsymbol{\theta}}_2 \\ \dot{\hat{\boldsymbol{\theta}}}_2 & = \boldsymbol{f}_\theta[\hat{\boldsymbol{\theta}}, \boldsymbol{\alpha}_d, \boldsymbol{u}(\dot{\boldsymbol{\alpha}}_{d2}, \boldsymbol{u}_f)]. \end{cases} \tag{5.14}$$

Since predicting $\hat{\boldsymbol{\theta}}$ trajectory with reduced system (5.14) takes less computational effort than predicting $\boldsymbol{\theta}$ trajectory with system (5.13) and yet guarantees that $\hat{\boldsymbol{\theta}}$ is a close approximation of $\boldsymbol{\theta}$, we solve an MPC problem to force the reduced system (5.14) to follow desired trajectory $\boldsymbol{\theta}_d$. In the MPC formulation, the implicit constraint of $\boldsymbol{\alpha}_{d1}$, $\boldsymbol{\alpha}_{d2}$ and $\dot{\boldsymbol{\alpha}}_{d2}$ in (5.14) should be expanded explicitly as the dynamic constraint

$$\begin{cases} \dot{\hat{\boldsymbol{\theta}}}_1 & = \hat{\boldsymbol{\theta}}_2 \\ \dot{\hat{\boldsymbol{\theta}}}_2 & = \boldsymbol{f}_\theta[\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\alpha}}, \boldsymbol{u}(\hat{\boldsymbol{w}}, \boldsymbol{u}_f)] \\ \dot{\hat{\boldsymbol{\alpha}}}_1 & = \hat{\boldsymbol{\alpha}}_2 \\ \dot{\hat{\boldsymbol{\alpha}}}_2 & = \hat{\boldsymbol{w}} \end{cases} \tag{5.15}$$

with the definition $\hat{\boldsymbol{\alpha}}_1 := \boldsymbol{\alpha}_{d1}, \hat{\boldsymbol{\alpha}}_2 := \boldsymbol{\alpha}_{d2}, \hat{\boldsymbol{w}} := \dot{\boldsymbol{\alpha}}_{d2}$.

The design variable of the MPC optimization problem is the input trajectory $\hat{\boldsymbol{w}}$, $\boldsymbol{u}_f$ and the initial values $\hat{\boldsymbol{\alpha}}_1(0)$ and $\hat{\boldsymbol{\alpha}}_2(0)$. Note that although the form of (5.15) is the same as (5.8), $[\hat{\boldsymbol{\alpha}}_1^T(0), \hat{\boldsymbol{\alpha}}_2^T(0)]^T$ in (5.15) are design variables that need to be optimized in MPC formulation, while $[\boldsymbol{\alpha}_1^T(0), \boldsymbol{\alpha}_2^T(0)]^T$ in (5.8) are measured robot states. Suppose we obtain the optimization solution $\{\hat{\boldsymbol{w}}, \boldsymbol{u}_f, \hat{\boldsymbol{\alpha}}_1(0), \hat{\boldsymbol{\alpha}}_2(0)\}$ for the MPC problem, the controller is then calculated according to (5.9) and (5.7), that is,

$$\begin{cases} \boldsymbol{v}(0) = \hat{\boldsymbol{w}}(0) - \frac{k_d}{\epsilon}(\boldsymbol{\alpha}_2 - \hat{\boldsymbol{\alpha}}_2(0)) - \frac{k_p}{\epsilon^2}(\boldsymbol{\alpha}_1 - \hat{\boldsymbol{\alpha}}_1(0)) \\ \boldsymbol{u}(\boldsymbol{v}(0), \boldsymbol{u}_f(0)) = [\boldsymbol{u}_d(\boldsymbol{v}(0), \boldsymbol{u}_f(0))^T, \boldsymbol{u}_f^T(0)]^T. \end{cases} \tag{5.16}$$

The MPC problem is solved in every control sampling period and only the first step input in the MPC input trajectory is actually implemented on the system.

5.5 GP-based planning and control

Built on physical principles, underactuated robot dynamics (5.4) might not be precise due to unmodeled dynamics and obtaining the model requires a lot domain knowledge. In this chapter, Gaussian processes regression is adopted to learn the unknown dynamics based on the collected input-state data. According to the derivation in the previous section, the controller requires the information of model $\boldsymbol{f}_\theta$ to compute prediction and model $\boldsymbol{f}_\alpha^{-1}$ to conduct inverse dynamics control. However, it turns out that directly learning $\boldsymbol{f}_\alpha^{-1}$ cannot stabilize the $\boldsymbol{\alpha}$ dynamics when the learned function is used to predict the states that are far away from the training data set. Actually, if the testing input is far away from the training input, the GPs model produces the predictive distribution which is a zero-mean Gaussian distribution. This is certainly not desirable for the inverse dynamics model. We rewrite the model (5.4) as:

$$\begin{cases} \dot{\boldsymbol{\theta}}_1 = \boldsymbol{\theta}_2 \\ \dot{\boldsymbol{\theta}}_2 = \boldsymbol{f}_\theta(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{u}_d, \boldsymbol{u}_f) \\ \dot{\boldsymbol{\alpha}}_1 = \boldsymbol{\alpha}_2 \\ \dot{\boldsymbol{\alpha}}_2 + \boldsymbol{\kappa}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f) = \boldsymbol{u}_d, \end{cases} \tag{5.17}$$

where $\boldsymbol{f}_\theta$ and $\boldsymbol{\kappa}_\alpha$ are unknown functions that need to be identified later. One benefit of reshaping to (5.17) is that the inverse dynamics model now becomes $\boldsymbol{u}_d = \boldsymbol{v} + \boldsymbol{\kappa}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f)$ with $\boldsymbol{v}$ specified in (5.9). If the testing input is far away from the training input, this inverse dynamics model degenerates to $\boldsymbol{u}_d = \boldsymbol{v}$, which stabilizes $\boldsymbol{\alpha}$ dynamics by choosing high feedback gain in (5.9).

In the $\boldsymbol{\alpha}$ subdynamics of (5.17), $\boldsymbol{u}_d$ is expressed as function of $\boldsymbol{u}_f$ and $\dot{\boldsymbol{\alpha}}_2$, and therefore, $\boldsymbol{f}_\theta(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{u}_d, \boldsymbol{u}_f)$ can be written as $\boldsymbol{f}_\theta[\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{u}_d(\dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f), \boldsymbol{u}_f]$. The learning model is formulated according to (5.17), namely,

$$\begin{cases} \dot{\boldsymbol{\theta}}_1 = \boldsymbol{\theta}_2 \\ \dot{\boldsymbol{\theta}}_2 \sim \boldsymbol{gp}_\theta(\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f) \\ \dot{\boldsymbol{\alpha}}_1 = \boldsymbol{\alpha}_2 \\ \boldsymbol{u}_d - \dot{\boldsymbol{\alpha}}_2 \sim \boldsymbol{gp}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f), \end{cases} \tag{5.18}$$

where $\boldsymbol{gp}_\theta$ and $\boldsymbol{gp}_\alpha$ are the predictive distributions given by Gaussian processes models aiming to recover the unknown functions $\boldsymbol{f}_\theta$ and $\boldsymbol{\kappa}_\alpha$, respectively. To train the Gaussian process regression models $\boldsymbol{gp}_\theta$ and $\boldsymbol{gp}_\alpha$, the inputs are tuples $\{\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f\}$ and the outputs are $\dot{\boldsymbol{\theta}}_2$ and $(\boldsymbol{u}_d - \dot{\boldsymbol{\alpha}}_2)$, respectively. For each output dimension, an individual Gaussian process regression model is built. Gaussian process regression models for different outputs are assumed independent from each other. The input and output data are collected experimentally.

5.5.1 GP-based inverse dynamics control for trajectory stabilization

The inverse dynamics mapping from the planned internal trajectory to the control input is learned with the GPs models. In the training phase, the inputs are $\{\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f\}$ and the output is $\boldsymbol{u}_d - \dot{\boldsymbol{\alpha}}_2$. In the testing phase, the learned GPs are used to predict the control input $\boldsymbol{u}_d$ in real time, that is,

$$\boldsymbol{u}_d \sim \boldsymbol{v} + \boldsymbol{gp}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f), \tag{5.19}$$

where $\boldsymbol{v}$ is the designed $\dot{\boldsymbol{\alpha}}_2$ value

$$\boldsymbol{v} = \hat{\boldsymbol{w}} - \frac{k_d}{\epsilon}(\boldsymbol{\alpha}_2 - \hat{\boldsymbol{\alpha}}_2(0)) - \frac{k_p}{\epsilon^2}(\boldsymbol{\alpha}_1 - \hat{\boldsymbol{\alpha}}_1(0)) + \boldsymbol{r}(t), \tag{5.20}$$

$\{\hat{\boldsymbol{w}}, \hat{\boldsymbol{\alpha}}_1(0), \hat{\boldsymbol{\alpha}}_2(0)\}$ are solution from the MPC design and $\boldsymbol{r}(t)$ is the added auxiliary control that will be determined later.

$$\boldsymbol{gp}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f) \sim \mathcal{N}(\boldsymbol{\mu}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f), \boldsymbol{\Sigma}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f))$$

is the predictive Gaussian distribution, where $\boldsymbol{\mu}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f)$ and $\boldsymbol{\Sigma}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f)$ are input dependent and computed from (A.5) in Appendix A. By (5.19),

$$\boldsymbol{u}_d \sim \mathcal{N}(\boldsymbol{\mu}_d(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f), \boldsymbol{\Sigma}_d(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f))$$

is the Gaussian distribution with

$$\begin{cases} \boldsymbol{\mu}_d(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f) & = \boldsymbol{v} + \boldsymbol{\mu}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f) \\ \boldsymbol{\Sigma}_d(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f) & = \boldsymbol{\Sigma}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f). \end{cases} \tag{5.21}$$

The mean value $\boldsymbol{\mu}_d$ is used for control input.

Under the inverse dynamics controllers (5.19) and (5.20), we now show that the $\boldsymbol{\alpha}$ subdynamics is stabilized to $\hat{\boldsymbol{\alpha}}$. Plugging the controller $\boldsymbol{\mu}_d(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f) = \boldsymbol{v} + \boldsymbol{\mu}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f)$ into (5.17), the closed-loop dynamics of the $\boldsymbol{\alpha}$ subsystem is obtained as

$$\begin{cases} \dot{\boldsymbol{\alpha}}_1 & = \boldsymbol{\alpha}_2 \\ \dot{\boldsymbol{\alpha}}_2 & = \boldsymbol{v} + \boldsymbol{\mu}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f) - \boldsymbol{\kappa}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f). \end{cases} \tag{5.22}$$

As defined in (5.12), $\boldsymbol{e}_{\alpha 1} = \boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_{d1} = \boldsymbol{\alpha}_1 - \hat{\boldsymbol{\alpha}}_1$ and $\boldsymbol{e}_{\alpha 2} = \boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_{d2} = \boldsymbol{\alpha}_2 - \hat{\boldsymbol{\alpha}}_2$. The closed-loop error dynamics for $\boldsymbol{e}_\alpha$ is obtained as

$$\dot{\boldsymbol{e}}_\alpha = \boldsymbol{A}\boldsymbol{e}_\alpha + \boldsymbol{B}[\boldsymbol{r}(t) + \boldsymbol{\mu}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f) - \boldsymbol{\kappa}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f)], \tag{5.23}$$

where

$$\boldsymbol{A}=\begin{bmatrix}\mathbf{0} & \boldsymbol{I}_n\\ -\frac{k_p}{\epsilon^2}\boldsymbol{I}_n & -\frac{k_d}{\epsilon}\boldsymbol{I}_n\end{bmatrix},\boldsymbol{B}=\begin{bmatrix}\mathbf{0}\\ \boldsymbol{I}_n\end{bmatrix}. \tag{5.24}$$

Note that $\boldsymbol{A}$ is Hurwitz as long as $k_p > 0$ and $k_d > 0$. In order to show that $\boldsymbol{e}_\alpha$ is stable, it is required that the unknown n dimensional disturbance $\boldsymbol{\mu}_\alpha(\boldsymbol{\theta},\boldsymbol{\alpha},\boldsymbol{v},\boldsymbol{u}_f)-\boldsymbol{\kappa}_\alpha(\boldsymbol{\theta},\boldsymbol{\alpha},\dot{\boldsymbol{\alpha}}_2,\boldsymbol{u}_f)$ is bounded.

From the learning theory given by the Lemma A.3 in Appendix A, we have that for any $0<\delta<1$,

$$\Pr\{||\boldsymbol{\mu}_\alpha(\boldsymbol{\theta},\boldsymbol{\alpha},\boldsymbol{v},\boldsymbol{u}_f)-\boldsymbol{\kappa}_\alpha(\boldsymbol{\theta},\boldsymbol{\alpha},\boldsymbol{v},\boldsymbol{u}_f)||\leq||\boldsymbol{\beta}_\alpha^T\boldsymbol{\Sigma}_\alpha^{\frac{1}{2}}(\boldsymbol{\theta},\boldsymbol{\alpha},\boldsymbol{v},\boldsymbol{u}_f)||\}\geq(1-\delta)^n, \tag{5.25}$$

where $\boldsymbol{\beta}_\alpha$ is n-dimensional vector with its ith element

$$\beta_{\alpha,i}=\sqrt{2\|\kappa_{\alpha,i}\|_k^2+300\gamma_{\alpha,i}\ln^3(\frac{N+1}{\delta})}.$$

The details of the notations are included in Appendix A.

In (5.23), the disturbance term $\boldsymbol{\mu}_\alpha(\boldsymbol{\theta},\boldsymbol{\alpha},\boldsymbol{v},\boldsymbol{u}_f)-\boldsymbol{\kappa}_\alpha(\boldsymbol{\theta},\boldsymbol{\alpha},\dot{\boldsymbol{\alpha}}_2,\boldsymbol{u}_f)$ does not have the exactly same input to $\boldsymbol{\mu}_\alpha$ and $\boldsymbol{\kappa}_\alpha$. Its bound is given by the following lemma.

Lemma 5.1. *For any $\delta\in(0,1)$, the n-dimensional unknown disturbance term $\boldsymbol{\mu}_\alpha(\boldsymbol{\theta},\boldsymbol{\alpha},\boldsymbol{v},\boldsymbol{u}_f)-\boldsymbol{\kappa}_\alpha(\boldsymbol{\theta},\boldsymbol{\alpha},\dot{\boldsymbol{\alpha}}_2,\boldsymbol{u}_f)$ is bounded with probability greater than $(1-\delta)^n$, that is,*

$$\Pr\{||\boldsymbol{\mu}_\alpha(\boldsymbol{\theta},\boldsymbol{\alpha},\boldsymbol{v},\boldsymbol{u}_f)-\boldsymbol{\kappa}_\alpha(\boldsymbol{\theta},\boldsymbol{\alpha},\dot{\boldsymbol{\alpha}}_2,\boldsymbol{u}_f)||\quad\leq\rho(\boldsymbol{e}_\alpha,\boldsymbol{\theta},\boldsymbol{\alpha},\boldsymbol{v},\boldsymbol{u}_f)\}\geq(1-\delta)^n, \tag{5.26}$$

where

$$\rho(\boldsymbol{e}_\alpha,\boldsymbol{\theta},\boldsymbol{\alpha},\boldsymbol{v},\boldsymbol{u}_f):=\lambda_{\min}^{-1}(\boldsymbol{I}+\frac{\partial\boldsymbol{\kappa}_\alpha}{\partial\boldsymbol{v}})(\sum_{i=0}^{2}c_i||\boldsymbol{e}_\alpha||^i+||\boldsymbol{\beta}_\alpha^T\boldsymbol{\Sigma}_\alpha^{\frac{1}{2}}(\boldsymbol{\theta},\boldsymbol{\alpha},\boldsymbol{v},\boldsymbol{u}_f)||), \tag{5.27}$$

with c_0, c_1, and c_2 positive constants.

The proof of Lemma 5.1 is included in Appendix C.1. For representation brevity, we denote $\rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta}) := \rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f)$ in the following derivation.

Lemma 5.2. *Supposing $k_d^2 > 4k_p > 0$ so that the Hurwitz matrix $\boldsymbol{A}$ in (5.24) has real eigenvalues. $\boldsymbol{A}$ is diagonalizable with $\boldsymbol{A} = \boldsymbol{M}\boldsymbol{\Lambda}\boldsymbol{M}^{-1}$ where $\boldsymbol{\Lambda}$ is the diagonalized matrix and $\boldsymbol{M}$ is an invertible coordinate transformation. With the positive definite matrix $\boldsymbol{Q} := \boldsymbol{M}^{-T}\boldsymbol{M}^{-1}$, $\boldsymbol{P}$ is the solution of the Lyapunov equation $\boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} = -\boldsymbol{Q}$. The auxiliary control $\boldsymbol{r}(t)$ is designed as*

$$\boldsymbol{r}(t) = \begin{cases} -\rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta})\frac{\boldsymbol{B}^T\boldsymbol{P}\boldsymbol{e}_\alpha}{||\boldsymbol{B}^T\boldsymbol{P}\boldsymbol{e}_\alpha||}; & \textit{if } ||\boldsymbol{B}^T\boldsymbol{P}\boldsymbol{e}_\alpha|| > \xi \\ -\frac{\rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta})}{\xi}\boldsymbol{B}^T\boldsymbol{P}\boldsymbol{e}_\alpha; & \textit{if } ||\boldsymbol{B}^T\boldsymbol{P}\boldsymbol{e}_\alpha|| \le \xi \end{cases} \tag{5.28}$$

with the designed parameter $\xi > 0$. Under this controller, the error $\|\boldsymbol{e}_\alpha(t)\|$ satisfies

$$\Pr\left\{\|\boldsymbol{e}_\alpha(t)\| \le d_1\|\boldsymbol{e}_\alpha(0)\|e^{\frac{\lambda_1}{4\epsilon}t} + d_2\right\} \ge (1-\delta)^n, \tag{5.29}$$

where $\lambda_1 = \frac{-k_d + \sqrt{k_d^2 - 4k_p}}{2}$, $d_1 = \sqrt{\frac{\lambda_{\max}(\boldsymbol{P})}{\lambda_{\min}(\boldsymbol{P})}}$, $d_2 = \sqrt{-\frac{2\epsilon c_3}{\lambda_1\lambda_{\min}(\boldsymbol{P})}}$ and positive constant c_3 is defined in (C.12).

The proof of Lemma 5.2 is shown in Appendix C.2. Note that since $\lambda_1 < 0$, as the singular perturbation parameter ϵ approaches to 0, $e^{\frac{\lambda_1}{4\epsilon}t}$ converges to 0 with the rate faster than any polynomial function of ϵ.

The GP-based inverse dynamics controller derived above only uses the mean value $\boldsymbol{\mu}_d$ of the predictive distribution (5.19). However, from Lemma 5.1, the covariance of the predictive distribution $\boldsymbol{\Sigma}_d = \boldsymbol{\Sigma}_\alpha$ determines the modeling error bound $\rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta})$. From Lemma 5.2, $\rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta})$ determines the control performance of $\boldsymbol{e}_\alpha$. We will show how to incorporate $\boldsymbol{\Sigma}_d$ information into the trajectory planning phase in order to enhance the control performance of $\boldsymbol{e}_\alpha$ in later section.

5.5.2 MPC for internal subsystem trajectory planning

Applying the inverse dynamics controllers (5.19) and (5.20) on the unknown robot dynamics model (5.17), the closed-loop dynamics becomes

$$\begin{cases} \dot{\boldsymbol{\theta}}_1 = \boldsymbol{\theta}_2 \\ \dot{\boldsymbol{\theta}}_2 = \boldsymbol{f}_\theta(\boldsymbol{\theta}, \hat{\boldsymbol{\alpha}} + \boldsymbol{e}_\alpha, \boldsymbol{u}_d(\hat{\boldsymbol{w}} + \dot{\boldsymbol{e}}_{\alpha_2}, \boldsymbol{u}_f), \boldsymbol{u}_f) \\ \dot{\boldsymbol{e}}_\alpha = \boldsymbol{A}\boldsymbol{e}_\alpha + \boldsymbol{B}[\boldsymbol{r}(t) + \boldsymbol{\mu}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f) - \boldsymbol{\kappa}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f)]. \end{cases} \tag{5.30}$$

Here, we have used $\boldsymbol{\alpha} = \hat{\boldsymbol{\alpha}} + \boldsymbol{e}_\alpha$ to replace $\boldsymbol{\alpha}$ and $\dot{\boldsymbol{\alpha}}_2 = \hat{\boldsymbol{w}} + \dot{\boldsymbol{e}}_{\alpha 2}$ to replace $\dot{\boldsymbol{\alpha}}_2$ in $\boldsymbol{f}_\theta$. In the previous subsection, we have proven the convergence of $\boldsymbol{e}_\alpha$. This subsection inspects how to use MPC to find desired internal subsystem trajectories $\boldsymbol{u}_f$, $\hat{\boldsymbol{w}}$ and $\hat{\boldsymbol{\alpha}}(0)$ upon which (5.19) and (5.20) are built.

In order to formulate an MPC problem, a prediction model should be proposed. We substitute the unknown function $\boldsymbol{f}_\theta$ in (5.30) with the learned GP model $\boldsymbol{gp}_\theta$ and obtain

$$\begin{cases} \dot{\boldsymbol{\theta}}_1 = \boldsymbol{\theta}_2 \\ \dot{\boldsymbol{\theta}}_2 \sim \boldsymbol{gp}_\theta(\boldsymbol{\theta}, \hat{\boldsymbol{\alpha}} + \boldsymbol{e}_\alpha, \hat{\boldsymbol{w}} + \dot{\boldsymbol{e}}_{\alpha 2}, \boldsymbol{u}_f) \\ \dot{\boldsymbol{e}}_\alpha = \boldsymbol{A}\boldsymbol{e}_\alpha + \boldsymbol{B}[\boldsymbol{r}(t) + \boldsymbol{\mu}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f) - \boldsymbol{\kappa}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f)]. \end{cases} \tag{5.31}$$

Assuming $\boldsymbol{e}_\alpha$ converges to zero rapidly, we obtain the reduced system in discretized form,

$$\begin{cases} \hat{\boldsymbol{\theta}}_1(k+1) = \hat{\boldsymbol{\theta}}_1(k) + \hat{\boldsymbol{\theta}}_2(k)\Delta t \\ \hat{\boldsymbol{\theta}}_2(k+1) \sim \hat{\boldsymbol{\theta}}_2(k) + \boldsymbol{gp}_\theta[\hat{\boldsymbol{\theta}}(k), \hat{\boldsymbol{\alpha}}(k), \hat{\boldsymbol{w}}(k), \boldsymbol{u}_f(k)]\Delta t \\ \hat{\boldsymbol{\alpha}}_1(k+1) = \hat{\boldsymbol{\alpha}}_1(k) + \hat{\boldsymbol{\alpha}}_2(k)\Delta t \\ \hat{\boldsymbol{\alpha}}_2(k+1) = \hat{\boldsymbol{\alpha}}_2(k) + \hat{\boldsymbol{w}}(k)\Delta t, \end{cases} \tag{5.32}$$

where Δt is the sampling period.

At the kth step, state trajectory is predicted for the control decision. We use $\hat{\boldsymbol{\theta}}(k + i|k), i = 0, ..., H + 1$, to denote the predicted state at the $(k + i)$th-step given the observation $\boldsymbol{\theta}(k)$ with the initial condition $\hat{\boldsymbol{\theta}}(k|k) = \boldsymbol{\theta}(k)$, and H is the prediction horizon. The prediction model is from (5.32) and specifically,

$$\hat{\boldsymbol{\theta}}(k+i+1|k) \sim \boldsymbol{F}\hat{\boldsymbol{\theta}}(k+i|k) + \boldsymbol{G}\boldsymbol{gp}_{\boldsymbol{\theta}}(k+i), \tag{5.33}$$

where

$$\boldsymbol{F} = \begin{bmatrix} \boldsymbol{I}_m & \Delta t \boldsymbol{I}_m \\ \mathbf{0}_m & \boldsymbol{I}_m \end{bmatrix}, \boldsymbol{G} = \begin{bmatrix} \mathbf{0}_m \\ \Delta t \boldsymbol{I}_m \end{bmatrix}. \tag{5.34}$$

We here use simple notation $\boldsymbol{gp}_{\boldsymbol{\theta}}(k+i) = \boldsymbol{gp}_{\theta}[\hat{\boldsymbol{\theta}}(k+i|k), \hat{\boldsymbol{\alpha}}(k+i|k), \hat{\boldsymbol{w}}(k+i), \boldsymbol{u}_f(k+i)]$ for presentation brevity. $\hat{\boldsymbol{\theta}}(k+i+1|k)$ is predicted from Gaussian process $\boldsymbol{gp}_{\boldsymbol{\theta}}$ in (5.33) and generally it does not satisfy Gaussian distribution even when $\hat{\boldsymbol{\theta}}(k+i|k)$ is Gaussian. To make the prediction manageable, we adopt an approximation method in [61], i.e., linearization of the posterior GP mean function $\boldsymbol{\mu}_{gp_\theta}$. By this approximation, $\hat{\boldsymbol{\theta}}(k+i+1|k)$ is a Gaussian distribution with the following mean and covariance:

$$\boldsymbol{\mu}_{\hat{\theta}}(k+i+1|k) = \boldsymbol{F}\boldsymbol{\mu}_{\hat{\theta}}(k+i|k) + \boldsymbol{G}\boldsymbol{\mu}_{gp_\theta}(k+i) \tag{5.35a}$$

$$\boldsymbol{\Sigma}_{\hat{\theta}}(k+i+1|k) = \boldsymbol{F}\boldsymbol{\Sigma}_{\hat{\theta}}(k+i|k)\boldsymbol{F}^T + \boldsymbol{G}(\frac{\partial \boldsymbol{\mu}_{gp_\theta}}{\partial \boldsymbol{\theta}}\boldsymbol{\Sigma}_{\hat{\theta}}(k+i)\frac{\partial \boldsymbol{\mu}_{gp_\theta}}{\partial \boldsymbol{\theta}}^T + \boldsymbol{\Sigma}_{gp_\theta})\boldsymbol{G}^T, \tag{5.35b}$$

where $\boldsymbol{\mu}_{gp_\theta}$ and $\boldsymbol{\Sigma}_{gp_\theta}$ are the mean value function and covariance function of the Gaussian process $\boldsymbol{gp}_{\theta}$, respectively. Note that $\boldsymbol{\Sigma}_{gp_\theta} = \boldsymbol{\Sigma}_{gp_\theta}(\boldsymbol{\mu}_{\hat{\theta}}(k+i|k), \hat{\boldsymbol{\alpha}}(k+i|k), \hat{\boldsymbol{w}}(k+i), \boldsymbol{u}_f(k+i))$ is input dependent. By Lemma A.1, $||\boldsymbol{\Sigma}_{gp_\theta}|| \leq \sigma^2_{\boldsymbol{f}\max} := \max_{j=1}^m(\sigma^2_{f_{\theta_j}} + \sigma^2_j)$, where j is the index of the dimension of $\boldsymbol{f}_\theta$, $\sigma_{f_{\theta_j}}$ and σ_j are hyper-parameters for the kernel function corresponding to $\boldsymbol{f}_{\theta_j}$. The following lemma is given to bound the state covariance $\boldsymbol{\Sigma}_{\hat{\theta}}(k+i|k)$.

Lemma 5.3. *Assume the learned mean value function $\boldsymbol{\mu}_{gp_\theta}$ is with bounded gradient with respect to $\boldsymbol{\theta}$, i.e. $||\frac{\partial \boldsymbol{\mu}_{gp_\theta}}{\partial \boldsymbol{\theta}}|| \leq L_1$. With small sampling period Δt, we have*

$$\|\boldsymbol{\Sigma}_{\hat{\theta}}(k+i|k)\| \leq i(\Delta t)^2\|\boldsymbol{\Sigma}_{gp_\theta}\| \leq i(\Delta t)^2\sigma^2_{\boldsymbol{f}\max}. \tag{5.36}$$

The proof of Lemma 5.3 is shown in Appendix C.3.

For the reduced system (5.32) to follow a desired trajectory $\boldsymbol{\theta}_d$, an MPC problem should be solved at each control sampling period with the objective function:

$$\begin{aligned}\bar{J}^k_{\hat{\theta},\hat{W}_H} &= \sum_{i=0}^{H} l_s(k+i) + \|\hat{\boldsymbol{\alpha}}(k)\|^2_{\boldsymbol{Q}_2} + l_f(k+H+1)\\ &= \sum_{i=0}^{H}\Big[\mathbb{E}\|\boldsymbol{e}_{\hat{\theta}}(k+i)\|^2_{\boldsymbol{Q}_1} + \|\hat{\boldsymbol{w}}(k+i)\|^2_{\boldsymbol{R}} + \|\boldsymbol{u}_f(k+i)\|^2_{\boldsymbol{R}}\Big] + \|\hat{\boldsymbol{\alpha}}(k)\|^2_{\boldsymbol{Q}_2}\\ &\qquad + \mathbb{E}\|\boldsymbol{e}_{\hat{\theta}}(k+H+1)\|^2_{\boldsymbol{Q}_3},\end{aligned} \tag{5.37}$$

where $\boldsymbol{e}_{\hat{\theta}}(k+i) = \hat{\boldsymbol{\theta}}(k+i|k) - \boldsymbol{\theta}_d(k+i)$. Matrices $\boldsymbol{Q}_i$, $i = 1, 2, 3$, and $\boldsymbol{R}$ are positive definite. The kth-step MPC input variable is

$$\hat{\boldsymbol{W}}_H(k) = \{\hat{\boldsymbol{\alpha}}(k), \hat{\boldsymbol{w}}(k+i), \boldsymbol{u}_f(k+i), i = 0, \ldots, H\}. \tag{5.38}$$

The expectation operator in (5.37) is used to approximate $\boldsymbol{\theta}(k+i)$ with the probabilistic variable $\hat{\boldsymbol{\theta}}(k+i|k)$ from (5.35). Noticing that for Gaussian distribution $\hat{\boldsymbol{\theta}}(k+i|k) \sim \mathcal{N}(\boldsymbol{\mu}_{\hat{\theta}}(k+i|k), \boldsymbol{\Sigma}_{\hat{\theta}}(k+i|k))$, we have $\mathbb{E}\|\boldsymbol{e}_{\hat{\theta}}(k+i)\|^2_{\boldsymbol{Q}_1} = \|\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+i)\|^2_{\boldsymbol{Q}_1} + \mathrm{tr}(\boldsymbol{Q}_1\boldsymbol{\Sigma}_{\hat{\theta}}(k+i|k))$, where $\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+i) = \boldsymbol{\mu}_{\hat{\theta}}(k+i|k) - \boldsymbol{\theta}_d(k+i)$.

In (5.37), the stage cost $l_s(k+i)$ is defined as

$$\begin{aligned}l_s(k+i) &= \mathbb{E}[\|\boldsymbol{e}_{\hat{\theta}}(k+i)\|^2_{\boldsymbol{Q}_1}] + \|\hat{\boldsymbol{w}}(k+i)\|^2_{\boldsymbol{R}} + \|\boldsymbol{u}_f(k+i)\|^2_{\boldsymbol{R}}\\ &= \|\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+i)\|^2_{\boldsymbol{Q}_1} + \mathrm{tr}(\boldsymbol{Q}_1\boldsymbol{\Sigma}_{\hat{\theta}}(k+i|k)) + \|\hat{\boldsymbol{w}}(k+i)\|^2_{\boldsymbol{R}} + \|\boldsymbol{u}_f(k+i)\|^2_{\boldsymbol{R}}.\end{aligned} \tag{5.39}$$

Similarly, the terminal cost $l_f(k+H+1)$ is defined as

$$l_f(k+H+1) = \mathbb{E}\,\|\boldsymbol{e}_{\hat{\theta}}(k+H+1)\|^2_{\boldsymbol{Q}_3} = \|\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+H+1)\|^2_{\boldsymbol{Q}_3} + \mathrm{tr}(\boldsymbol{Q}_3\boldsymbol{\Sigma}_{\hat{\theta}}(k+H+1|k)). \tag{5.40}$$

The reduced dynamics (5.35) is used to predict the future trajectory and this fact brings computational benefit. However, the objective function (5.37) neglects the $\boldsymbol{\alpha}$ subsystem dynamics. In fact, the convergent rate and the domain of attraction of $\boldsymbol{e}_\alpha$ affect

$\boldsymbol{\theta}$ tracking performance as shown in (5.30). To incorporate the internal subsystem trajectory tracking performance into the MPC problem, we modify the objective function as

$$J^k_{\hat{\theta},\hat{W}_H} = \bar{J}^k_{\hat{\theta},\hat{W}_H} + \nu\|\boldsymbol{\Sigma}_d(\hat{\boldsymbol{W}}_H(k))\|, \tag{5.41}$$

where $\boldsymbol{\Sigma}_d(\hat{\boldsymbol{W}}_H(k))$ is the covariance of the predictive distribution (5.19) at the kth step and $\nu > 0$ is a weighting factor. The rationale to include $\boldsymbol{\Sigma}_d$ in the cost function is to incorporate the inverse dynamics model uncertainty in the MPC planning phase. As shown in Lemmas 5.1 and 5.2, the converging property of $\boldsymbol{e}_\alpha$ depends on $\boldsymbol{\Sigma}_d$. By encouraging small $\boldsymbol{\Sigma}_d$, MPC picks the planned trajectory that can be stabilized by the inverse dynamics controller with high confidence. The significance of adding this term into the objective function is demonstrated in the control performance analysis and the result sections.

The optimal control input by the MPC design is denoted as

$$\hat{\boldsymbol{W}}^0_H(k) = \operatorname{argmin}_{\hat{\boldsymbol{W}}_H(k)} J^k_{\hat{\theta},\hat{W}_H}. \tag{5.42}$$

The optimization is formulated as an unconstrained MPC and solved with gradient decent approach. The optimal control input $\hat{\boldsymbol{W}}^0_H(k)$ is used in the inverse dynamics controller (5.20).

Two items need to be clarified before the MPC solution (5.42) can guarantee the convergence of actual state variable $\boldsymbol{\theta}$ to the desired $\boldsymbol{\theta}_d$. First, the approximated model (5.35) is used instead of the unaccessible actual model (5.30) for computing the state prediction. The impact of using this model approximation on tracking stability is discussed in Section 5.6. Second, even though the prediction $\boldsymbol{\mu}_{\hat{\theta}}$ from (5.35) is an accurate approximation of $\boldsymbol{\theta}$ from (5.30), the convergence of $\boldsymbol{\mu}_{\hat{\theta}}$ to the desired $\boldsymbol{\theta}_d$ under the controller (5.42) needs to be proven. The rest of this subsection is devoted to address the second item above. It should be noted that since the prediction model (5.35a) for $\boldsymbol{\mu}_{\hat{\theta}}$ is exact, no difference exists between $\boldsymbol{\mu}_{\hat{\theta}}(k+i|k)$ and $\boldsymbol{\mu}_{\hat{\theta}}(k+i|k+j)$, $j \leq i$, in the discussion of the convergence of $\boldsymbol{\mu}_{\hat{\theta}}$ to $\boldsymbol{\theta}_d$. The input (5.42) does not automatically guarantee the convergence of $\boldsymbol{\mu}_{\hat{\theta}}$ to $\boldsymbol{\theta}_d$ because of the finite prediction horizon. The stability can be instead ensured with the appropriate choice of the terminal cost $l_f(k+H+1)$ and the terminal constraint as shown in [76]. We here briefly describe the terminal cost design process to ensure this convergence.

Suppose that for the desired trajectory $\boldsymbol{\theta}_d$, there exists the corresponding desired inputs $\{\boldsymbol{\alpha}_d, \boldsymbol{w}_d, \boldsymbol{u}_{f,d}\}$ satisfying the mean propagation dynamics (5.35a), that is,

$$\boldsymbol{\theta}_d(k+i+1) = \boldsymbol{F}\boldsymbol{\theta}_d(k+i) + \boldsymbol{G}\boldsymbol{\mu}_{gp_\theta}(\boldsymbol{\theta}_d, \boldsymbol{\alpha}_d, \boldsymbol{w}_d, \boldsymbol{u}_{f,d}). \tag{5.43}$$

To show the stability of tracking error $\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}} = \boldsymbol{\mu}_{\hat{\theta}} - \boldsymbol{\theta}_d$ under the controller (5.42), we assess the $\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}$ dynamics by taking the difference between (5.43) and (5.35a), namely,

$$\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+i+1) = \boldsymbol{F}\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+i) + \boldsymbol{G}[\boldsymbol{\mu}_{gp_\theta}(\boldsymbol{\mu}_{\hat{\boldsymbol{\theta}}}, \hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{w}}, \boldsymbol{u}_f) - \boldsymbol{\mu}_{gp_\theta}(\boldsymbol{\theta}_d, \boldsymbol{\alpha}_d, \boldsymbol{w}_d, \boldsymbol{u}_{f,d})]. \tag{5.44}$$

Defining input $\boldsymbol{u}_e = [\hat{\boldsymbol{\alpha}}^T - \boldsymbol{\alpha}_d^T \; \hat{\boldsymbol{w}}^T - \boldsymbol{w}_d^T \; \boldsymbol{u}_f^T - \boldsymbol{u}_{f,d}^T]^T$, (5.44) is then linearized around its equilibrium point at the origin and we obtain

$$\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+i+1) = \boldsymbol{A}_e\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+i) + \boldsymbol{B}_e\boldsymbol{u}_e(k+i) \tag{5.45}$$

with $\boldsymbol{A}_e = \boldsymbol{F} + \boldsymbol{G}\frac{\partial \boldsymbol{\mu}_{gp_\theta}}{\partial \boldsymbol{\theta}_d}$ and $\boldsymbol{B}_e = \boldsymbol{G}\left[(\frac{\partial \boldsymbol{\mu}_{gp_\theta}}{\partial \boldsymbol{\alpha}_d})^T \; (\frac{\partial \boldsymbol{\mu}_{gp_\theta}}{\partial \boldsymbol{w}_d})^T \; (\frac{\partial \boldsymbol{\mu}_{gp_\theta}}{\partial \boldsymbol{u}_{f,d}})^T\right]^T$.

By [76], stability of the error dynamics (5.44) is guaranteed by the solution $\hat{\boldsymbol{W}}_H^{*0}(k)$ of the following MPC problem

$$\hat{\boldsymbol{W}}_H^{*0}(k) = \operatorname{argmin}_{\hat{\boldsymbol{W}}_H(k)} J_{\hat{\theta},\hat{W}_H}^{k*}, \tag{5.46}$$

where $J_{\hat{\theta},\hat{W}_H}^{k*} = \sum_{i=0}^{H} l_s^*(k+i) + l_f^*(k+H+1)$, with

$$l_s^*(k+i) = \|\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+i)\|_{\boldsymbol{Q}_1^*}^2 + \|\boldsymbol{e}_{\hat{\alpha}}(k+i)\|_{\boldsymbol{Q}_2^*}^2 + \|\Delta\hat{\boldsymbol{w}}(k+i)\|_{\boldsymbol{R}^*}^2 + \|\Delta\boldsymbol{u}_d\|_{\boldsymbol{R}^*}^2, \tag{5.47a}$$

$$l_f^*(k+H+1) = \|\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+H+1)\|_{\boldsymbol{Q}_3^*}^2, \tag{5.47b}$$

$\boldsymbol{e}_{\hat{\alpha}}(k+i) = \hat{\boldsymbol{\alpha}}(k+i) - \boldsymbol{\alpha}_d(k+i)$, $\Delta\hat{\boldsymbol{w}}(k+i) = \hat{\boldsymbol{w}}(k+i) - \boldsymbol{w}_d(k+i)$, and $\Delta\boldsymbol{u}_d = \boldsymbol{u}_f(k+i) - \boldsymbol{u}_{f,d}(k+i)$. Positive definite matrices $\boldsymbol{Q}_i^*$, $i = 1, 2, 3$, and $\boldsymbol{R}^*$ are chosen for design specification. [76] proposed a systematical approach to design the terminal cost matrix $\boldsymbol{Q}_3^*$ and the corresponding terminal region $\boldsymbol{\Omega}_e$. Within $\boldsymbol{\Omega}_e$, a linear state feedback controller is designed for the linearized system (5.45) to ensure the stability of the original dynamics (5.44) with the decreasing terminal cost, that is, if $\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+H+1) \in \boldsymbol{\Omega}_e$, controller $\boldsymbol{u}_e = -\boldsymbol{K}_e\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}$ guarantees $\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+H+2) \in \boldsymbol{\Omega}_e$ with $l_f^*(k+H+2) \leq l_f^*(k+H+1) - l_s^*(k+H+1)$ for (5.44).

Taking the MPC objective function (5.46) under the optimal input as the Lyapunov function, we have

$$\begin{aligned} J^{(k+1)*}_{\hat{\theta}^0,\hat{W}_H^{*0}} - J^{k*}_{\hat{\theta}^0,\hat{W}_H^{*0}} &= -l_s^*(k) + l_f^*(k+H+2) - l_f^*(k+H+1) + l_s^*(k+H+1) \\ &\leq -l_s^*(k). \end{aligned}$$

From the monotonicity of $J^{k*}_{\hat{\theta}^0,\hat{W}_H^{*0}}$, namely, $\|\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k)\|^2_{\boldsymbol{Q}_1^*} \leq J^{k*}_{\hat{\theta}^0,\hat{W}_H^{*0}} \leq \|\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k)\|^2_{\boldsymbol{Q}_3^*}$, we have

$$J^{(k+1)*}_{\hat{\theta}^0,\hat{W}_H^{*0}} \leq (1 - \frac{\lambda_{\min}(\boldsymbol{Q}_1^*)}{\lambda_{\max}(\boldsymbol{Q}_3^*)})J^{k*}_{\hat{\theta}^0,\hat{W}_H^{*0}}.$$

Comparing our proposed MPC problems (5.42) with (5.46), we notice two main differences. The first difference is that the former one includes the model uncertainty through the covariance terms. The second difference is that the former does not need the desired input trajectories $\boldsymbol{\alpha}_d$, $\boldsymbol{w}_d$ and $\boldsymbol{u}_{f,d}$, which are difficult to obtain. The MPC problem (5.41) only assumes that the desired trajectories exist but unknown. This is one of the attractive properties of the proposed control design.

To apply the stability result of (5.46) to show the stability of the MPC design in (5.41), the following lemmas are needed.

Lemma 5.4. *For the terminal cost $l_f^*(k+H+1)$ and the stage cost $l_s^*(k+i)$ defined in (5.47), let matrices $\boldsymbol{Q}_1$ and $\boldsymbol{R}$ in (5.39) and $\boldsymbol{Q}_3$ in (5.40) satisfy $\boldsymbol{Q}_1 = \boldsymbol{Q}_1^*$,*

$\lambda_{\max}(\boldsymbol{R}) < \lambda_{\min}(\boldsymbol{R}^*)$ *and* $\boldsymbol{Q}_3 = \boldsymbol{Q}_3^*$*, then*

$$\begin{aligned} l_f(k+H+2) \ &\leq l_f(k+H+1) + \operatorname{tr}(\boldsymbol{Q}_3 \boldsymbol{\Sigma}_{\hat{\theta}}(k+H+2)) \\ &\quad - l_s(k+H+1) + \operatorname{tr}(\boldsymbol{Q}_1 \boldsymbol{\Sigma}_{\hat{\theta}}(k+H+1)) \end{aligned}$$

if the following conditions are satisfied

$$\|\hat{\boldsymbol{w}}(k+H+1)\| \geq \lambda_R \|\boldsymbol{w}_d(k+H+1)\|, \tag{5.48a}$$

$$\|\boldsymbol{u}_f(k+H+1)\| \geq \lambda_R \|\boldsymbol{u}_{f,d}(k+H+1)\|, \tag{5.48b}$$

where $\lambda_R = \frac{2\lambda_{\min}(\boldsymbol{R}^*)}{\lambda_{\min}(\boldsymbol{R}^*) - \lambda_{\max}(\boldsymbol{R})}$.

The proof of this lemma is included in Appendix C.4. With the result in Lemma 5.4 on the terminal cost, we obtain the bound of tracking error $\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+i) := \boldsymbol{\mu}_{\hat{\theta}}(k+i|k) - \boldsymbol{\theta}_d(k+i)$ for $i = 0, \ldots, H+1$ by the following lemma.

Lemma 5.5. *Using the objective function* $J^k_{\hat{\theta}^0, \hat{W}^0_H}$ *under the optimal input (5.42) as the Lyapunov function, the tracking error satisfies* $\|\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+i)\| \leq a_4(i)\|\boldsymbol{e}_\theta(k)\| + a_5(i)$, *where* $a_4(i) = d_3^{\frac{i}{2}} \sqrt{\frac{\lambda_{\max}(\boldsymbol{Q}_3)}{\lambda_{\min}(\boldsymbol{Q}_1)}}$ *and* $a_5(i) = \sqrt{\frac{d_3^i(\alpha^2_{\max} + \nu\sigma^2_{\kappa\max}) + d_4\frac{1-d_3^i}{1-d_3}}{\lambda_{\min}(\boldsymbol{Q}_1)}}$*. Here* $0 < d_3 = 1 - \frac{\lambda_{\min}(\boldsymbol{Q}_1)}{\lambda_{\max}(\boldsymbol{Q}_3)} < 1$ *and* $d_4 = (1 + \frac{\lambda_{\min}(\boldsymbol{Q}_1)}{\lambda_{\max}(\boldsymbol{Q}_3)})(\nu\sigma^2_{\kappa\max} + \alpha^2_{\max}) + m\lambda_m(H+2)(\Delta t)^2\sigma^2_{\boldsymbol{f}\max}$ *where* $\lambda_m = \lambda_{\max}(\boldsymbol{Q}_1) + \lambda_{\max}(\boldsymbol{Q}_3)$.

The proof of Lemma 5.5 is included in Appendix C.5. Because $0 < d_3 < 1$, $a_4(i)$ converges to 0 and $a_5(i)$ converges to $\sqrt{\frac{d_4}{(1-d_3)\lambda_{\min}(\boldsymbol{Q}_1)}}$ exponentially as i goes to infinity. Since $J^k_{\hat{\theta}^0, \hat{W}^0_H}$ is positive definite, this lemma shows that if we choose $J^k_{\hat{\theta}^0, \hat{W}^0_H}$ as Lyapunov function, then values of $J^k_{\hat{\theta}^0, \hat{W}^0_H}$ decrease along the trajectory predicted from model (5.35) as long as (5.48) holds. The result in Lemma 5.5 implies that by solving the MPC problem (5.42), the mean value variable $\boldsymbol{\mu}_{\hat{\theta}}$ predicted by (5.35) is stabilized to $\boldsymbol{\theta}_d$ exponentially.

5.5.3 Summary of GP-based planning and control

The framework of GPs based planning and control is shown in Figure 5.1. In each control sampling period, the trajectory planner solves the MPC problem (5.42) with the model (5.35) and gives the planned internal trajectory $\hat{\boldsymbol{W}}_H^0$ as output. The MPC also takes the predicted variance $\boldsymbol{\Sigma}_d$ from the inverse dynamics model into consideration. The inverse dynamics controller then takes the designed internal trajectory $\hat{\boldsymbol{W}}_H^0$ and uses (5.19) and (5.20) to predict the control input $\boldsymbol{\mu}_d$ and implement on the actuator.

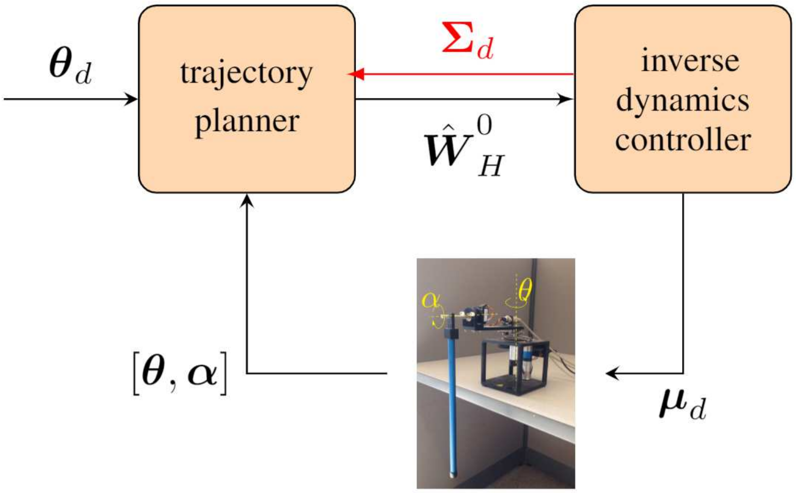


Figure 5.1: GP-based planning and control framework

The fact that GPs provide prediction uncertainty is attractive for robust control. In our proposed framework, the prediction uncertainty is used in both the MPC based trajectory planning and the inverse dynamics stabilization. We adopt the approach used in [64] and [61] that incorporates the GPs uncertainty into prediction horizon to plan internal trajectory. The uncertainty $\boldsymbol{\Sigma}_d$ of the inverse dynamics model provided by GPs in (5.19) also affects the trajectory planning phase.

5.6 Control performance analysis

In this section, we show stability analysis of the proposed control framework and present the impact of modeling error to controller performance.

For the unknown robot dynamics model (5.17), under the learning based inverse dynamics controller (5.19) and (5.20), we have the closed loop dynamics (5.30). We assume that both models (5.17) and (5.30) are deterministic with unknown functions $\boldsymbol{f}_\theta$ and $\boldsymbol{\kappa}_\alpha$. For (5.30), we propose Lyapunov candidate function

$$V(k) = V_\theta(k) + \zeta V_\alpha(k), \tag{5.49}$$

where constant $\zeta > 0$, $V_\alpha(k) = \boldsymbol{e}_\alpha^T(k)\boldsymbol{P}\boldsymbol{e}_\alpha(k)$, $\boldsymbol{P}$ is defined in Lemma 5.2, and $V_\theta(k)$ is similar to the MPC cost function in (5.37) without expectation operator used in $\bar{J}^k_{\hat{\theta},\hat{W}_H}$ and under the optimal control $\hat{\boldsymbol{W}}^0_H(k)$, namely, $V_\theta(k) = \bar{J}^k_{\theta,\hat{W}^0_H}$,

$$\begin{aligned} \bar{J}^k_{\theta,\hat{W}^0_H} &= \sum_{i=0}^{H}\big[\|\boldsymbol{e}_\theta(k+i)\|^2_{Q_1} + \|\hat{\boldsymbol{w}}^0(k+i)\|^2_R + \|\boldsymbol{u}^0_f(k+i)\|^2_R\big] \\ &\quad + \|\hat{\boldsymbol{\alpha}}^0(k)\|^2_{Q_2} + \|\boldsymbol{e}_\theta(k+H+1)\|^2_{Q_3}. \end{aligned} \tag{5.50}$$

Here $\boldsymbol{e}_\theta(k+i) = \boldsymbol{\theta}(k+i) - \boldsymbol{\theta}_d(k+i)$. Note that $\bar{J}^k_{\theta,\hat{W}^0_H}$ is a quadratic function of the actual state $\boldsymbol{\theta}(k+i)$ following the unknown deterministic model (5.30) under the optimal control $\hat{\boldsymbol{W}}^0_H$ given by (5.42). At the kth step, it is impossible to directly evaluate $\bar{J}^k_{\theta,\hat{W}^0_H}$ because unaccessible future states and the unknown model (5.30), and instead, its value is approximated by $\bar{J}^k_{\hat{\theta}^0,\hat{W}^0_H}$ in (5.37).

We assess the decreasing property of the proposed Lyapunov candidate function as

$$\begin{aligned} \Delta V(k) &= [V(k+1) - J^{k+1}_{\hat{\theta}^0,\hat{W}^0_H}] - [V(k) - J^k_{\hat{\theta}^0,\hat{W}^0_H}] + J^{k+1}_{\hat{\theta}^0,\hat{W}^0_H} - J^k_{\hat{\theta}^0,\hat{W}^0_H} \\ &= [\bar{J}^{k+1}_{\theta,\hat{W}^0_H} + \zeta V_\alpha(k+1) - \bar{J}^{k+1}_{\hat{\theta}^0,\hat{W}^0_H} - \nu\|\boldsymbol{\Sigma}_d(\hat{W}^0_H(k+1))\|] \\ &\quad -[\bar{J}^k_{\theta,\hat{W}^0_H} + \zeta V_\alpha(k) - \bar{J}^k_{\hat{\theta}^0,\hat{W}^0_H} - \nu\|\boldsymbol{\Sigma}_d(\hat{W}^0_H(k))\|] + J^{k+1}_{\hat{\theta}^0,\hat{W}^0_H} - J^k_{\hat{\theta}^0,\hat{W}^0_H} \\ &= (\bar{J}^{k+1}_{\theta,\hat{W}^0_H} - \bar{J}^{k+1}_{\hat{\theta}^0,\hat{W}^0_H}) - (\bar{J}^k_{\theta,\hat{W}^0_H} - \bar{J}^k_{\hat{\theta}^0,\hat{W}^0_H}) + \zeta[V_\alpha(k+1) - V_\alpha(k)] \\ &\quad +(J^{k+1}_{\hat{\theta}^0,\hat{W}^0_H} - J^k_{\hat{\theta}^0,\hat{W}^0_H}) - \nu[\|\boldsymbol{\Sigma}_d(\hat{W}^0_H(k+1))\| - \|\boldsymbol{\Sigma}_d(\hat{W}^0_H(k))\|]. \end{aligned} \tag{5.51}$$

In above equation, term $\bar{J}^k_{\theta,\hat{W}^0_H} - \bar{J}^k_{\hat{\theta}^0,\hat{W}^0_H}$ quantifies the difference between the approximated reduced system cost-to-go and the actual cost-to-go at the kth step. At the kth step, we use $\boldsymbol{\theta}(k+i|k)$ for $i \geq 0$ to denote the model predicted value of $\boldsymbol{\theta}(k+i)$ given the measured state $\boldsymbol{\theta}(k)$ with the initial condition $\boldsymbol{\theta}(k|k) = \boldsymbol{\theta}(k)$. Similar to (5.33), the evolution of $\boldsymbol{\theta}(k+i|k)$ follows discretized form of (5.18), namely,

$$\boldsymbol{\theta}(k+i+1|k) \sim \boldsymbol{F}\boldsymbol{\theta}(k+i|k) + \boldsymbol{G}\boldsymbol{g}\boldsymbol{p}_\theta(k+i) \tag{5.52}$$

with the mean value $\boldsymbol{\mu}_\theta(k+i+1|k)$ and variance $\boldsymbol{\Sigma}_\theta(k+i+1|k)$ calculations similar to (5.35). The difference between model (5.52) and model (5.33) is that the former depends on the actual internal state $\boldsymbol{\alpha}(k+i)$, while the latter uses the designed internal state $\hat{\boldsymbol{\alpha}}(k+i|k)$ by the singular purturbed model. Model (5.33) is actually used for $\boldsymbol{\theta}$ trajectory prediction through the MPC formulation. Figure 5.2 further illustrates the relationship among the three different $\boldsymbol{\theta}$-prediction models (5.17), (5.52) and (5.33).

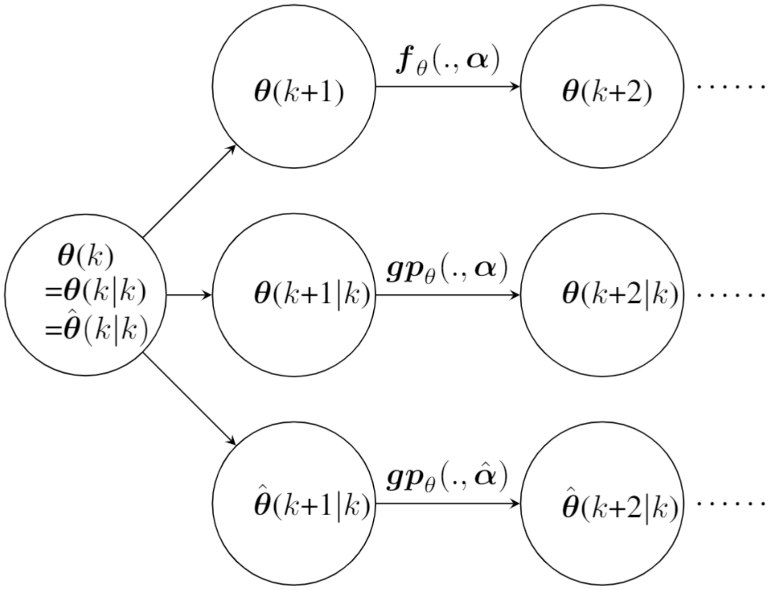

$\boldsymbol{\theta}(k)$
$=\boldsymbol{\theta}(k|k)$
$=\hat{\boldsymbol{\theta}}(k|k)$
$\boldsymbol{\theta}(k+1)$
$\boldsymbol{f}_\theta(.,\boldsymbol{\alpha})$
$\boldsymbol{\theta}(k+2)$
$\boldsymbol{\theta}(k+1|k)$
$\boldsymbol{g}\boldsymbol{p}_\theta(.,\boldsymbol{\alpha})$
$\boldsymbol{\theta}(k+2|k)$
$\hat{\boldsymbol{\theta}}(k+1|k)$
$\boldsymbol{g}\boldsymbol{p}_\theta(.,\hat{\boldsymbol{\alpha}})$
$\hat{\boldsymbol{\theta}}(k+2|k)$
$\cdots\cdots$

Figure 5.2: Flow chart of the state estimation by three predictive models.

We now quantify the difference between $\hat{\boldsymbol{\theta}}(k+i|k)$ and $\boldsymbol{\theta}(k+i|k)$. At $i=0$, $\hat{\boldsymbol{\theta}}(k|k) = \boldsymbol{\theta}(k|k) = \boldsymbol{\theta}(k)$. The difference between $\hat{\boldsymbol{\theta}}(k+i|k)$ and $\boldsymbol{\theta}(k+i|k)$ comes from the difference between the reduced model (5.33) and full model (5.52).

Lemma 5.6. *Assuming the mean value of the predictive distribution* [1] $\boldsymbol{\mu}_{gp_\theta}(\boldsymbol{\mu}_\theta, \boldsymbol{\alpha})$ *is Lipshitz in* $\boldsymbol{\mu}_\theta$ *and* $\boldsymbol{\alpha}$*, namely,*

$$\|\boldsymbol{\mu}_{gp_\theta}(\cdot, \boldsymbol{\alpha}) - \boldsymbol{\mu}_{gp_\theta}(\cdot, \hat{\boldsymbol{\alpha}})\| \leq L_2 \|\boldsymbol{e}_\alpha\|,$$
$$\|\boldsymbol{\mu}_{gp_\theta}(\boldsymbol{\mu}_\theta, \cdot) - \boldsymbol{\mu}_{gp_\theta}(\boldsymbol{\mu}_{\hat{\theta}}, \cdot)\| \leq L_3 \|\boldsymbol{\mu}_\theta - \boldsymbol{\mu}_{\hat{\theta}}\|$$

with constants $L_2, L_3 > 0$*,* $\tilde{\boldsymbol{\mu}}_\theta(k+i) := \boldsymbol{\mu}_\theta(k+i|k) - \boldsymbol{\mu}_{\hat{\theta}}(k+i|k)$ *satisfies* $\|\tilde{\boldsymbol{\mu}}_\theta(k+i)\| \leq \varrho_{\hat{\theta}}(i)\|\boldsymbol{e}_\alpha(k)\| + \varrho_2(i)$*, where*

$$\varrho_{\hat{\theta}}(i) \;=\; d_1 L_2 \Delta t \left[\left(\frac{1-a_1^i}{1-a_1} - i\right)\left(1 - \frac{L_3 \Delta t}{1-a_1}\right) + i\right],$$

$a_1 = e^{\frac{\lambda_1}{4\epsilon}\Delta t}$ *and* $\varrho_2(i) = d_2 L_2 \Delta t[i + \frac{1}{2} L_3 \Delta t (i-1) i]$. d_1*,* d_2*,* λ_1 *are defined in Lemma 5.2 and* Δt *is the sampling period.*

The proof of this lemma is included in Appendix C.6. We then inspect the difference between $\boldsymbol{\theta}(k+i|k)$ and $\boldsymbol{\theta}(k+i)$. The difference between $\boldsymbol{\theta}(k+i|k)$ and $\boldsymbol{\theta}(k+i)$ comes from the difference between the learning model (5.52) and the unknown actual model (5.17). From Lemma A.3, we obtain the GP learned prediction guaranteed to be closed to the m-dimensional model $\boldsymbol{f}_\theta$ with probability [2], i.e.,

$$\Pr\{\|\boldsymbol{\mu}_{gp_\theta}(\boldsymbol{\theta}, \boldsymbol{\alpha}) - \boldsymbol{f}_\theta\| \leq \|\boldsymbol{\beta}_\theta^T \boldsymbol{\Sigma}_{gp_\theta}^{\frac{1}{2}}\|\} \geq (1-\delta)^m, \tag{5.53}$$

where $\boldsymbol{\beta}_\theta$ is a m-dimensional vector with its jth element

$$\beta_{\theta,j} = \sqrt{2||f_{\theta,j}||_k^2 + 300\gamma_{\theta,j} \ln^3(\frac{N+1}{\delta})}.$$

$\gamma_{\theta,j}$ is the maximum information gain for $f_{\theta,j}$. For $i > 0$, the following lemma gives the difference upper-bound of $\boldsymbol{\theta}_\mu(k+i) := \boldsymbol{\mu}_\theta(k+i|k) - \boldsymbol{\theta}(k+i)$.

Lemma 5.7. *Defining the probabilistic events* $\boldsymbol{\Pi}_{\mu_\theta}^\theta(k+i) = \big\{\|\boldsymbol{\theta}_\mu(k+i)\| \leq \varrho_{\mu_\theta}\big\}$ *for* $i = 1, \ldots, H+1$*, where* $\varrho_{\mu_\theta} = \Delta t \sum_{j=0}^{i-1} \|\boldsymbol{\beta}_\theta^T \boldsymbol{\Sigma}_{gp_\theta}^{\frac{1}{2}}(k+j|k)\|$*. Then,* $\Pr\Big\{\bigcap_{j=0}^{i} \boldsymbol{\Pi}_{\mu_\theta}^\theta(k+j)\Big\} \geq (1-\delta)^{mi}$.

[1] *For presentation convenience, we drop the third and four arguments and use notation* $\boldsymbol{\mu}_{gp_\theta}(\boldsymbol{\mu}_\theta, \boldsymbol{\alpha})$ *to represent* $\boldsymbol{\mu}_{gp_\theta}(\boldsymbol{\mu}_\theta, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f)$*.*

[2] We here drop all arguments $(\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f)$ of functions $\boldsymbol{f}_\theta$ and $\boldsymbol{\Sigma}_{gp_\theta}$ for presentation brevity.

The proof of this lemma is included in Appendix C.7. Lemmas 5.6 and 5.7 give the error bounds on $\tilde{\boldsymbol{\mu}}_\theta(k+i) = \boldsymbol{\mu}_\theta(k+i|k) - \boldsymbol{\mu}_{\hat{\theta}}(k+i|k)$ and $\boldsymbol{\theta}_\mu(k+1) = \boldsymbol{\mu}_\theta(k+i|k) - \boldsymbol{\theta}(k+i)$, respectively. Combining these results, we have the following lemma on the error bound of $\tilde{\boldsymbol{\theta}}_\mu(k+i) := \boldsymbol{\mu}_{\hat{\theta}}(k+i|k) - \boldsymbol{\theta}(k+i)$.

Lemma 5.8. *Defining the probabilistic events*

$$\boldsymbol{\Pi}^{\theta}_{\mu_{\hat{\theta}}}(k+i) = \Big\{\|\tilde{\boldsymbol{\theta}}_\mu(k+i)\| \le \varrho_{\hat{\theta}}(i)\|\boldsymbol{e}_\alpha(k)\| + a_2(i)\Big\}$$

for $i = 1, \ldots, H+1$, *where* $a_2(i) = \varrho_2(i) + \varrho_{\mu_\theta}$, *then*

$$\Pr\Big\{\bigcap_{j=0}^{i} \boldsymbol{\Pi}^{\theta}_{\mu_{\hat{\theta}}}(k+j)\Big\} \ge (1-\delta)^{mi+n(i-1)}.$$

With the result of Lemma 5.8, we estimate the difference of $\bar{J}^k_{\theta,\hat{W}^0_H} - \bar{J}^k_{\hat{\theta}^0,\hat{W}^0_H}$ in the following lemma.

Lemma 5.9. *Defining the probabilistic event*

$$\boldsymbol{\Pi}^{J}_{\bar{J}}(k) = \Big\{|\bar{J}^k_{\hat{\theta}^0,\hat{W}^0_H} - \bar{J}^k_{\theta,\hat{W}^0_H}| \le \rho_J(\boldsymbol{e}_\alpha, \boldsymbol{e}_\theta)\Big\},$$

where

$$\begin{aligned} \rho_J(\boldsymbol{e}_\alpha, \boldsymbol{e}_\theta) \;&=\; \lambda_{\max}(\boldsymbol{Q}_3)\sum_{i=0}^{H+1}\Big\{\bar{\xi}_1(i)\|\boldsymbol{e}_\alpha(k)\|^2 + \bar{\xi}_2(i)\|\boldsymbol{e}_\alpha(k)\|\|\boldsymbol{e}_\theta(k)\| \\ &\quad + \bar{\xi}_3(i)\|\boldsymbol{e}_\alpha(k)\| + \bar{\xi}_4(i)\|\boldsymbol{e}_\theta(k)\| + \bar{\xi}_5(i)\Big\}, \end{aligned} \tag{5.54}$$

$\bar{\xi}_1(i) = \varrho^2_{\hat{\theta}}(i)$, $\bar{\xi}_2(i) = 2\varrho_{\hat{\theta}}(i)a_4(i)$, $\bar{\xi}_3(i) = 2\varrho_{\hat{\theta}}(i)[a_2(i) + a_5(i)]$, $\bar{\xi}_4(i) = 2a_2(i)a_4(i)$, *and* $\bar{\xi}_5(i) = a_2(i)(a_2(i) + 2a_5(i)) + mi(\Delta t)^2\sigma^2_{\boldsymbol{f}\,\max}$. $\varrho_{\hat{\theta}}(i)$ *and* $a_2(i)$ *are defined in Lemmas 5.6 and 5.8, respectively.* $a_4(i)$ *and* $a_5(i)$ *are defined in Lemma 5.5, then*

$$\Pr\{\boldsymbol{\Pi}^{J}_{\bar{J}}(k)\} \ge (1-\delta)^{m(H+1)+nH}. \tag{5.55}$$

The proof of this lemma is included in Appendix C.8.

The results in Lemma 5.9 are used to apply for the first two pairs of terms of $\Delta V(k)$ in (5.51). Letting $\boldsymbol{e}(k) = [\boldsymbol{e}^T_\theta(k)\ \boldsymbol{e}^T_\alpha(k)]^T$ denote the error vector, it is straightforward to obtain that the Lyapunov function candidate $V(k)$ in (5.49) satisfies $\underline{\lambda}\|\boldsymbol{e}(k)\|^2 \leq V(k) \leq \overline{\lambda}\|\boldsymbol{e}(k)\|^2$, where $\underline{\lambda} = \min(\lambda_{\min}(\boldsymbol{Q}_1), \zeta\lambda_{\min}(\boldsymbol{Q}))$ and $\overline{\lambda} = \max(\lambda_{\max}(\boldsymbol{Q}_3),$ $\zeta\lambda_{\max}(\boldsymbol{Q}))$, where matrices $\boldsymbol{Q}$ and $\boldsymbol{Q}_1$ are defined in Lemma 5.2 and (5.37), respectively. We are now ready to give the following result for the overall controller performance.

Theorem 5.1. *For the parameters $\bar{\xi}_j(i)$, $i = 0, 1, \dots, H+2$, $j = 1, \dots, 5$, given in Lemma 5.9, define $\xi_j = \overline{\lambda}\left[\bar{\xi}_j(0) + 2\sum_{i=1}^{H+1}\bar{\xi}_j(i) + \bar{\xi}_j(H+2)\right]$, $\gamma_1 = \sqrt{\eta}$, $\gamma_2 = \frac{\xi_3}{2\gamma_1}$, $\gamma_3 = \sqrt{\lambda_{\min}(\boldsymbol{Q}_1)}$, $\gamma_4 = \frac{\xi_4}{\gamma_3}$, and $\gamma_5 = \frac{\xi_4^2}{\gamma_3^2} + \frac{\xi_3^2}{4\gamma_1^2} + \xi_5 + \hat{\alpha}^2_{\max} + \nu\sigma^2_{\boldsymbol{\kappa}\max} + \zeta c_3\Delta t + m\lambda_m(H+2)(\Delta t)^2\sigma^2_{\boldsymbol{f}\max}$, where constant $\lambda_m = \lambda_{\max}(\boldsymbol{Q}_1) + \lambda_{\max}(\boldsymbol{Q}_3)$. The following fact is then held*

$$\begin{aligned}&\Pr\{V(k+1) \leq (1 - \frac{\gamma_3^2}{4\overline{\lambda}})V(k) + \gamma_5\} \geq \\ &\Pr\left\{\boldsymbol{\Pi}_{\bar{J}}^{J}(k+1) \cap \boldsymbol{\Pi}_{\bar{J}}^{J}(k)\right\} \geq (1-\delta)^{(m+n)H+2m+n}\end{aligned}$$

if

$$\eta = \frac{1}{4}\zeta\lambda_{\min}(\boldsymbol{Q})\Delta t - \xi_1 - \frac{\xi_2^2}{2\lambda_{\min}(\boldsymbol{Q}_1)} - \frac{\lambda_{\min}(\boldsymbol{Q}_1)}{4} > 0. \tag{5.56}$$

The proof of Theorem 5.1 is shown in Appendix C.9. If $V(k+1) \leq (1-\frac{\gamma_3^2}{4\overline{\lambda}})V(k) + \gamma_5$ holds for i consecutive steps, we have

$$V(k+i) \leq (1 - \frac{\gamma_3^2}{4\overline{\lambda}})^i V(k) + \gamma_5 \frac{4\overline{\lambda}(1 - (1 - \frac{\gamma_3^2}{4\overline{\lambda}})^i)}{\gamma_3^2}. \tag{5.57}$$

Defining $V_{ss} = \lim_{i\to\infty} V(k+i)$ and $\|\boldsymbol{e}\|_{ss} = \lim_{i\to\infty}\|\boldsymbol{e}(k+i)\|$ for any fixed k, then $V_{ss} \leq \frac{4\overline{\lambda}}{\gamma_3^2}\gamma_5$ and $\|\boldsymbol{e}\|_{ss} \leq \sqrt{\frac{4\overline{\lambda}}{\gamma_3^2\underline{\lambda}}\gamma_5}$. Modeling errors are important factors for the steady-state error bound. The term $a_2(i)$ defined in Lemma 5.8 quantifies the modeling error for $\boldsymbol{e}_\theta$ dynamics. As the value of $a_2(i)$ increases, terms ξ_3, ξ_4 and ξ_5 increase and consequently γ_5 increases. The term $\rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta})$ defined in Lemma 5.1 quantifies the modeling error for $\boldsymbol{e}_\alpha$ dynamics. As the value of $\rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta})$ increases, γ_5 also increases.

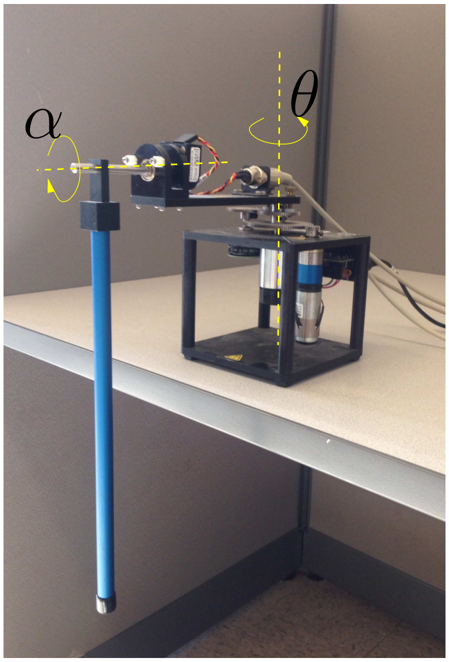


Figure 5.3: Rotary Inverted Pendulum: θ is the actuated DOF; α is the unactuated DOF.

5.7 Experimental results

We demonstrate the algorithm on the rotary inverted pendulum platform as shown in Figure 5.3. The hardware system is provided by Quanser Consulting Inc. In this system, the acutauted joint is the rotary base angle θ and the unactuated joint is the pendulum angle α. The pendulum angle α is defined to be 0 when it is vertically upright. The rotary base is controlled by a motor. The voltage V_m applied to the motor is the control input to the system. The control goal is to balance the pendulum around upright while the rotary base angle should track a desired trajectory θ_d.

To obtain the learned model of the system, we need to perturb the system and collect the motion data. We implement an open loop actuator input

$$V_m = \begin{cases} v_1 \sin(f_1 t) + v_2 \sin(f_2 t) \ , |\alpha| \leq \frac{\pi}{3} \\ 0 \ , |\alpha| > \frac{\pi}{3}, \end{cases} \tag{5.58}$$

where the unit is voltage. Here v_1 and v_2 are designed to satisfy the voltage limit $|V_m| \leq$

5. f_1 and f_2 are designed to excite the system in both low and high frequency. In our experiment, we choose $v_1 = 3$, $v_2 = 1.5$, $f_1 = 8$ rad/sec and $f_2 = 40$ rad/sec. Under this input, we swing up the pendulum manually by giving the pendulum a velocity when the angle $|\alpha| \geq \frac{\pi}{2}$. This open-loop controller cannot stabilize the pendulum to stay around the upright position. For each swing, the pendulum angle α might stay in the range of $|\alpha| \leq \frac{\pi}{3}$ for less than one second and then fall. We swing up the pendulum repeatedly to get enough data for training the model. Control input V_m and motion data $[\theta, \alpha, \dot{\theta}, \dot{\alpha}, \ddot{\theta}, \ddot{\alpha}]$ are recorded when $|\alpha| \leq \frac{\pi}{3}$. The joint angles $[\theta, \alpha]$ are measured with encoders. Their velocities $[\dot{\theta}, \dot{\alpha}]$ and accelerations $[\ddot{\theta}, \ddot{\alpha}]$ are obtained by once and twice numerical differentiation of the joint angles with low pass filters. The open-loop controller is executed at the rate of 100Hz. The data is also collected at the sampling frequency of 100 Hz. In Figure 5.4, an example trial of collected θ and α angles under the open-loop input is shown. It can be seen from the α angle in Figure 5.4 that the pendulum cannot achieve balance and multiple trials of manual swing is applied on the pendulum in order to collect enough data satisfying $|\alpha| \leq \frac{\pi}{3}$.

We adopt the Gaussian processes regression method to train machine learning models. The chosen inputs and outputs are described in Section 5.5. The only assumed knowledge about the physical system is that the learned model is not a function of the rotary base angle θ. Therefore, θ is not an input to the model. The prior Gaussian processes are assumed to have zero-mean function and squared exponential kernel functions. The hyper-parameters of the kernel function are determined through maximizing the log likelihood [58].

The controller implementation of the rotary inverted pendulum system is through Matlab Simulink. Within each control sampling period, the MPC trajectory planner and inverse dynamics controller give updated output. The MPC previews the desired trajectory from current time to 0.56 sec later. The previewed trajectory is sampled with a period of 0.02 sec, that is, $\Delta t = 0.02$ in equation (5.32) and $H = 27$ in (5.37).

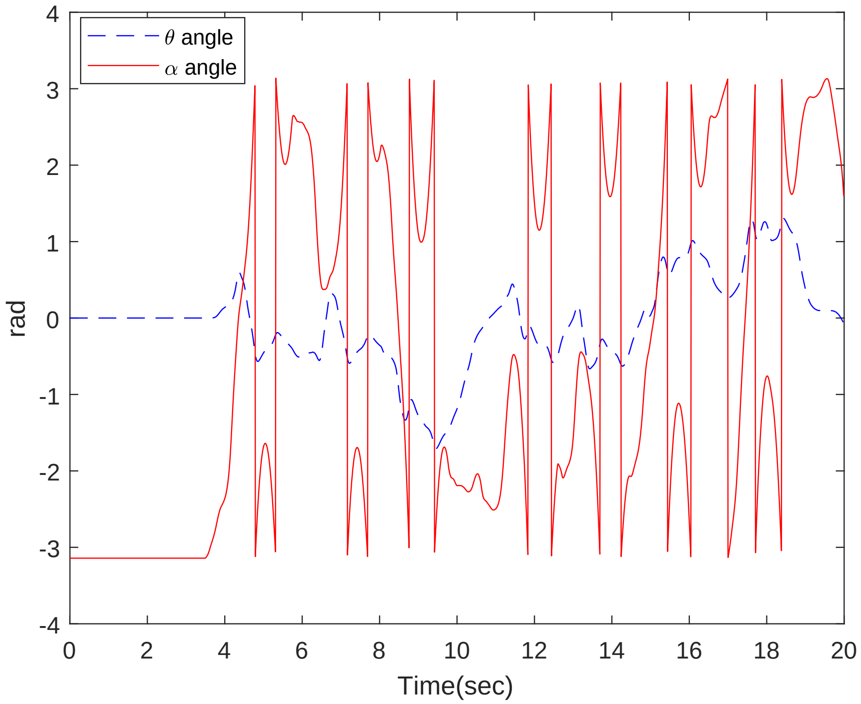
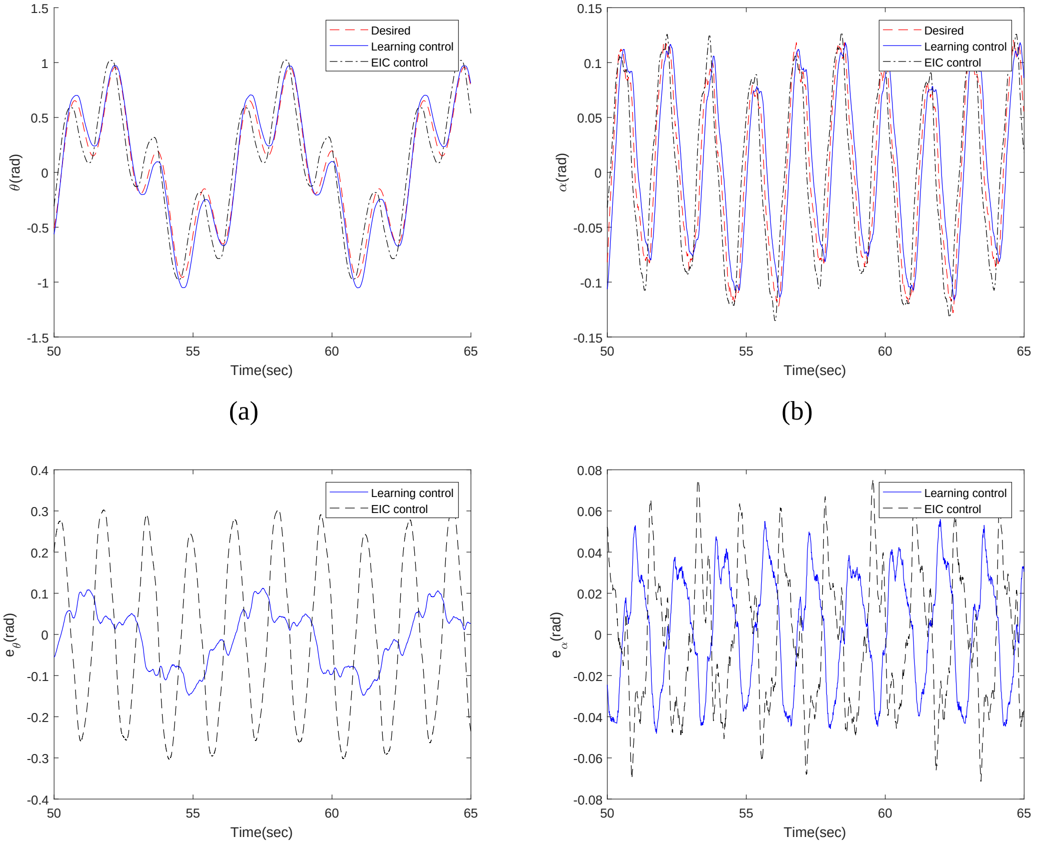


Figure 5.4: Collected training data under the open loop input excitation.

The weight matrices in objective (5.37) in implementation are $\boldsymbol{Q}_1 = \mathrm{diag}([100, 100])$, $\boldsymbol{Q}_2 = \mathrm{diag}([100, 100])$, $\boldsymbol{Q}_3 = \mathrm{diag}([1000, 1000])$, $\boldsymbol{R} = 10$ and $\nu = 1$ in (5.41).

Figure 5.5 shows the control performance under the proposed design. A set of 800 training data points are collected as the training data set. In this experiment, the desired external trajectory was designed $\theta_d = 0.6\sin(t) + 0.4\sin(4t)$ rad. Figure 5.5(a) shows the tracking performance of external subsystem angle θ and Figure 5.5(b) for internal subsystem angle α. For comparison purpose, we also include the physical model-based EIC control performance given in [13]. Figures 5.5(c) and 5.5(d) further compare the tracking errors $\boldsymbol{e}_\theta$ and $\boldsymbol{e}_\alpha$ under these two controllers. It is clear from these results that the learning-based control design effectively captures the underactuated balance robotic dynamics and both the external tracking and internal balancing tasks are satisfactory. The performance under the learning-based design outperforms that with the physical model-based controller.



(a)

(b)

(c)

(d)

Figure 5.5: Control performance comparison between the proposed learned model-based design and the EIC-based controller. (a) External angle θ tracking profiles. (b) Internal angle α tracking profiles. (c) External angle tracking errors e_θ. (d) Internal angle tracking errors e_α.

To understand the influence of training data sizes on the control performance, we vary the number of the training data set from 200 to 800 points to obtain different learned models. These models are used to track the same trajectory $\theta_d(t)$ rad as mentioned above. Figure 5.6 shows the error distribution contours under different numbers of training data set for learning control and the EIC-based control design. For each learned model, the plot includes the tracking errors of a 90-sec motion duration. The results clearly imply that with only 200 training data, the controller barely achieves the balancing and tracking tasks with large errors. With the increasing training data sets, the magnitudes of both the balancing and tracking errors decrease. With a set of 800

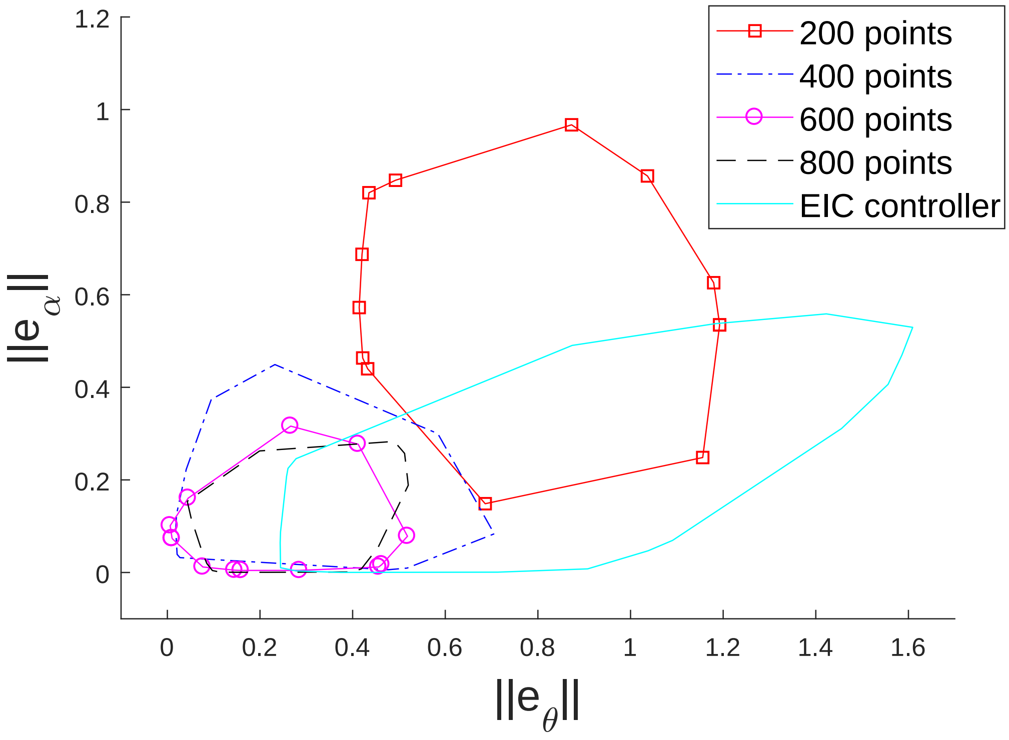


Figure 5.6: Impact of training data number to the tracking and balancing performance

training data points, the learned model-based controller achieves superior performance than that under the analytical model-based controller.

The trade-off between the tracking and balancing performance is tuned by the choice of ν value in the MPC objective function (5.41). Experiments are conducted to show the performance with the same learned model (obtained by using 200 training data points) under different values of ν. Figure 5.7 shows the performance of the tracking and balancing errors with different ν values. We intentionally chose an inaccurate learned model and therefore, the value of $\|\Sigma_d\|$ in (5.41) is relatively large. When $\nu = 0$, the balancing performance is not robust due to the poor inverse dynamics model. With $\nu = 10$, the system achieves a good trade-off between balancing and tracking tasks. With a further increased ν value (i.e., $\nu = 40, 60$), the tracking performance becomes worse, and when $\nu > 80$ the controller fails to balance the pendulum.

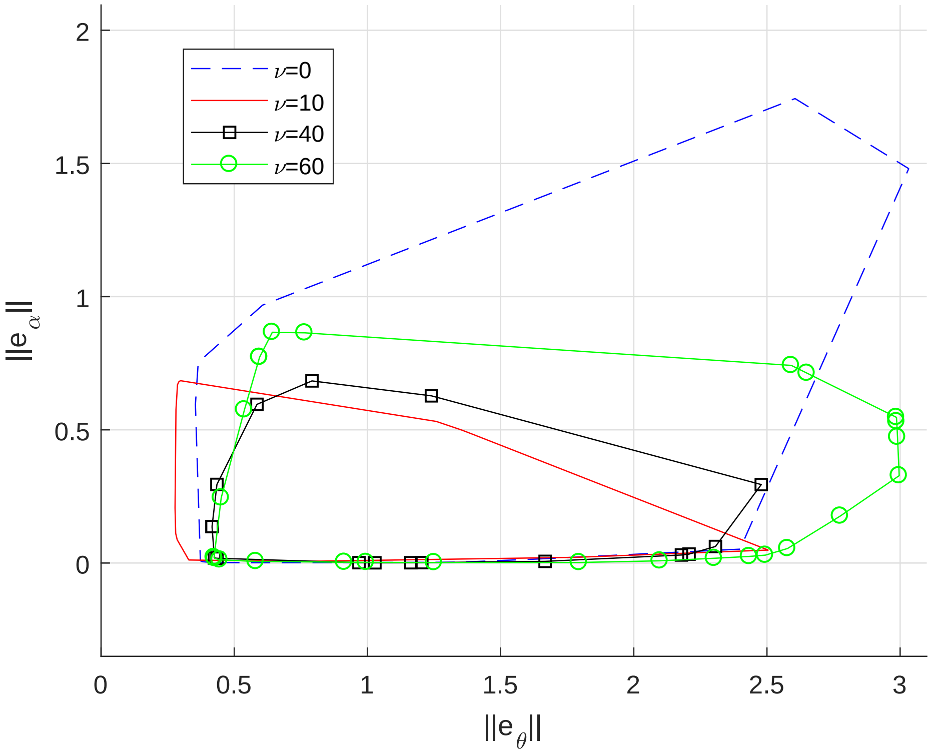


Figure 5.7: Control performance with different ν values

The averages of the variance of the inverse dynamics model for 60-second trials are 0.255, 0.174, 0.108 and 0.108 for $\nu = 0, 10, 40, 60$, respectively. The results clearly show that with increased ν values, the magnitude of $\boldsymbol{\Sigma}_d$ decreases. This confirms that the use of the integrated $\|\boldsymbol{\Sigma}_d\|$ in the objective function helps the control performance improvements.

5.8 Summary

This chapter proposed a learning model based planning and control framework for underactuated balance robot. The characteristic of the underactuated balance robot is that the equilibrium point varies according to the desired output trajectory rather than being a constant point. We proposed a trajectory planning and stabilization framework. The trajectory planner used an MPC design to find the desired internal trajectory. The inverse dynamics controller stabilized the system state onto the planned internal trajectory.

The GPs learning model adopted in this work provided predictive distribution with the mean value and the variance. The proposed planning and control framework explicitly preferred trajectories with small prediction variance through online optimization. This property resulted in improved performance robustness as shown in the experiments. Furthermore, the learned model can be obtained by exciting the system dynamics without prior knowledge about the system or successful balancing demonstration.

Chapter 6

Conclusions and Future Works

6.1 Conclusions

This dissertation studied the learning-based modeling and control approaches of underactuated balance robots. The most significant property of underactuated balance robots is that the unactuated subsystem is not directly controlled by inputs but indirectly affected by the actuated subsystem. This property determines that the robot state can only travel a latent manifold, not any arbitrary trajectory of the entire state space. Many control approaches of underactuated balance robots are motivated by identifying the latent manifold and designing feedback control laws to stabilize the robot state onto the latent manifold [11] [13]. However, the physical-principle-based latent manifold identification and stabilization approaches rely on in-depth knowledge of the robot system, which limits their applications. This dissertation takes a machine learning viewpoint and approach to identify the latent manifold and design the stabilization controllers for underactuated balance robot.

In Chapter 2, the learned latent manifold was used in modeling high dimensional human-robot interaction. In many human-robot interaction examples, the high dimensional human limbs motion can be captured by latent manifold model. A new dimensional reduction approach was developed to construct learning-based latent manifold with physical interpretation. The learning-based model for human limbs subsystem was coupled with the physical model for human trunk and robot subsystem through inputs and constraints. The integrated model was applied on human-bikebot interaction pose estimation in the extended Kalman filter framework. The proposed learning-based approach gave accurate pose estimation for human-bikebot interaction compared to the direct measurements from IMUs.

In Chapter 3, the physical-principled latent manifold design was extended to study biped slip recovery control. Slipping dynamics was obtained by removing the constraint of stance foot sticking to ground from the normal walking dynamics. A sequence of recovery gaits, i.e., the virtual constraints, were designed so that it would bring the robot from slipping back to periodic walking. In the hybrid zero dynamics space, periodical walking corresponded to a periodic orbit; while slip recovery could be viewed as a trajectory that initially was perturbed away from the periodical orbit and finally converged back to the walking periodic orbit. The extended biped model is validated by experiments.

Chapters 2 and 3 showed that both the learning-based and the physical-principled latent manifold approaches can be applied to model high dimensional robot dynamics. In Chapter 4, the relation between these two approaches were discussed with the application to biped walking. It was shown that the learning-based latent manifold model can reconstruct the physical-principled virtual constraints and the zero dynamics. This implied that the balancing skill could be identified by learning-based latent manifold approach from demonstration.

In Chapter 5, a learning-model based control framework was proposed to achieve underactuated balance robot tracking while balancing task. The latent manifold was online identified by the MPC approach to design feasible trajectories for tracking while balancing task. The robot state was stabilized onto the latent manifold by an inverse dynamics controller. The learned models provided confidence level measurement that was considered in the optimization based trajectory planning in order to improve the control robustness. The controller outperformed the physical-principled controller without any prior knowledge of the model or successful balancing demonstration. The proposed control framework was demonstrated on a rotary inverted pendulum and was readily applicable to a broad class of underactuated balance robots such as autonomous bicycles.

6.2 Future works

There are a few research directions that follow this dissertation's work. Chapter 5 has proposed a learning-model based controller design that does not require prior knowledge of the system or successful demonstration data, and yet achieves competitive performance as analytical model based control design. It is expected that such controller design could be extended to high degrees of freedom systems such as autonomous bicycle and biped robot. Even though the autonomous bicycle modeling and control problems have been studied in [5] [77], several open problems still need to be solved to achieve fully autonomous bicycle goal. It is more challenging to balance bicycle in low speed than in high speed. At low speed riding, the contribution of steering and acceleration to the bicycle balancing is different from that during high speed riding. Learning-based modeling approach could be applied to identify the bicycle dynamics in both low and high speeds. It is also a challenging task to achieve autonomous bicycle driving on rough terrain such as uneven or slippery ground. Under these ground conditions, the nonholonomic wheel contact constraint is no longer satisfied. Machine learning approach could be applied to build a dynamic model for the bicycle platform that takes the tires-ground friction and dragging force into consideration. With this precise tires-ground interaction model, new control framework could be proposed to achieve robust and agile maneuvers on rough terrains.

In Chapter 3, the design criteria of the slip recovery gait sequence was proposed. Human demonstrated slip recovery gaits were modified to satisfy the criteria. However, human might not demonstrate the optimal slip recovery policy. A formal approach would be formulated to construct an optimization problem with the criteria in Chapter 3 and other physical inspired objective function, such as energy cost.

Besides the optimization approach to obtain the latent manifold for balancing skill, Chapter 4 gave an example of learning the latent manifold from demonstration. This idea can be extended to learning from demonstration [74] by learning the demonstrated latent manifold and applying to a new robot. Due to the possible structure difference between the teaching and the apprentice robots, the latent manifold cannot be directly copied but be customized on the apprentice robot. All of these research topics can be the future works of this dissertation.

Appendices

Appendix A
Gaussian Process

Gaussian process (GP) is a flexible non-parametric machine learning model. The word "non-parametric" means that the number of model parameters is changing with the number of training data. Besides its flexibility, a significant advantage of GP is that it predicts both the mean value and covariance of the output which makes it compatible with probabilistic state estimation and control. For the input space with more training data, GP predicts output with lower uncertainty. In this section, we briefly introduce the derivation of GP. Detailed introduction of GP is referred to [58].

A.1 Regression

A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution. A real value process $f(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}^1$ is determined by its mean value function $\mu(\boldsymbol{x})$ and covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$ as

$$\begin{aligned}
&\mu(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})], \\
&k(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}[(f(\boldsymbol{x}) - \mu(\boldsymbol{x}))(f(\boldsymbol{x}') - \mu(\boldsymbol{x}'))].
\end{aligned} \tag{A.1}$$

The mean value function and covariance functions are determined through model selection and hyperparameters learning which will be described in the next section.

Suppose the training data set contains N input output data pairs $\mathcal{D} = \{\boldsymbol{x}^i, y^i\}_{i=1}^N$. The observed output y^i is a noisy observation of the underlying function value with zero mean Gaussian noise ω, i.e., $y^i = f(\boldsymbol{x}^i) + \omega$ with $\omega \sim \mathcal{N}(0, \sigma^2)$. The observation vector is denoted as $\boldsymbol{y} = [y^1, ..., y^N]^T$ and the input design matrix is denoted as

$\boldsymbol{X} = [\boldsymbol{x}^1, ..., \boldsymbol{x}^N]^T$. At a testing point $\boldsymbol{x}^* \in \mathbb{R}^d$, we would like to predict its corresponding function value $f^* = f(\boldsymbol{x}^*)$ according to the observed training data $\mathcal{D}$. The joint distribution of the training outputs $\boldsymbol{y}$ and the testing output f^* is

$$\begin{bmatrix} \boldsymbol{y} \\ f^* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} \boldsymbol{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma^2 \boldsymbol{I}_N & \boldsymbol{k}(\boldsymbol{X},\boldsymbol{x}^*) \\ \boldsymbol{k}(\boldsymbol{X},\boldsymbol{x}^*)^T & k(\boldsymbol{x}^*,\boldsymbol{x}^*) \end{bmatrix}\right), \tag{A.2}$$

where $\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X})$ is the $N \times N$ kernel matrix whose element is $\boldsymbol{K}_{i,j}(\boldsymbol{X},\boldsymbol{X}) = k(\boldsymbol{x}^i, \boldsymbol{x}^j)$. $\boldsymbol{I}_N$ is the $N \times N$ identity matrix. $\boldsymbol{k}(\boldsymbol{X},\boldsymbol{x}^*)$ is an $N \times 1$ column vector whose element is $\boldsymbol{k}_i(\boldsymbol{X},\boldsymbol{x}^*) = k(\boldsymbol{x}^i, \boldsymbol{x}^*)$.

The probabilistic prediction of f^* is given by the conditional distribution

$$f^* | \boldsymbol{x}^*, \mathcal{D} \sim \mathcal{N}(\mu(\boldsymbol{x}^*), \Sigma(\boldsymbol{x}^*)), \tag{A.3}$$

where $\mu(\boldsymbol{x}^*)$ is the posterior mean function and $\Sigma(\boldsymbol{x}^*)$ is the posterior covariance function given by

$$\begin{aligned} \mu(\boldsymbol{x}^*) &= \boldsymbol{k}(\boldsymbol{X},\boldsymbol{x}^*)^T [\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma^2 \boldsymbol{I}_N]^{-1} \boldsymbol{y}, \\ \Sigma(\boldsymbol{x}^*) &= k(\boldsymbol{x}^*,\boldsymbol{x}^*) - \boldsymbol{k}(\boldsymbol{X},\boldsymbol{x}^*)^T [\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X}) + \sigma^2 \boldsymbol{I}_N]^{-1} \boldsymbol{k}(\boldsymbol{X},\boldsymbol{x}^*). \end{aligned} \tag{A.4}$$

GPs can also be applied to learn n-dimensional vector-valued function $\boldsymbol{f}(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}^n$. In such cases, GPs are adopted to learn each function $f_i(\boldsymbol{x}), i = 1, ..., n$, as $f_i^* | \boldsymbol{x}^*, \mathcal{D} \sim \mathcal{N}(\mu_i(\boldsymbol{x}^*), \Sigma_i(\boldsymbol{x}^*))$ independently. The predictive distribution can be written as

$$\begin{aligned} &\boldsymbol{f}^* | \boldsymbol{x}^*, \mathcal{D} \sim \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{x}^*), \boldsymbol{\Sigma}(\boldsymbol{x}^*)), \\ &\boldsymbol{\mu}(\boldsymbol{x}^*) = [\mu_1(\boldsymbol{x}^*), ..., \mu_n(\boldsymbol{x}^*)]^T, \boldsymbol{\Sigma}(\boldsymbol{x}^*) = \mathrm{diag}[\Sigma_1(\boldsymbol{x}^*), ..., \Sigma_n(\boldsymbol{x}^*)]. \end{aligned} \tag{A.5}$$

Because each dimension is assumed to be independent from each other, the covariance matrix $\boldsymbol{\Sigma}$ is diagonal.

A.2 Model selection and hyperparameter learning

As mentioned in the previous section, the Gaussian process is determined by the covariance function (also called kernel function) in (A.1). A covariance function corresponds to a set of basis feature functions in regression problem. In model selection, we choose the function form of covariance function: how the covariance function depends on the input data and how the covariance function depends on the hyperparameters. The most frequently used covariance function is the squared exponential (SE) function defined as

$$k(\boldsymbol{x}^i, \boldsymbol{x}^j) = \sigma_f^2 \exp(-\frac{1}{2}(\boldsymbol{x}^i - \boldsymbol{x}^j)^T \boldsymbol{M}(\boldsymbol{x}^i - \boldsymbol{x}^j)) + \sigma^2 \delta_{ij}, \tag{A.6}$$

where $\boldsymbol{\theta} = \{\boldsymbol{M}, \sigma_f^2, \sigma^2\}$ is the hyperparameter set. $\delta_{ij} = 1$ if $\boldsymbol{x}^i = \boldsymbol{x}^j$, otherwise $\delta_{ij} = 0$. The SE covariance function is static because it only depends on the distance between two points and ignores the exact location of any point. The fact that SE function value decreases as the distance between two points increases implies that the modeled function tends to have similar function value for close input points. The hyperparameter $\boldsymbol{M}$ determines the speed and direction of this decreasing. Throughout this dissertation, we focus on the SE kernel. The following lemma is given on the upper bound of covariance of SE kernel.

Lemma A.1. *For any testing point $\boldsymbol{x}^* \in \mathbb{R}^d$, the posterior covariance $\Sigma(\boldsymbol{x}^*)$ of SE kernel is bounded by*

$$\Sigma(\boldsymbol{x}^*) \leq \sigma_f^2 + \sigma^2, \tag{A.7}$$

where σ_f and σ are the hyperparameters in the kernel function (A.6). For n-dimensional vector Gaussian processes $\boldsymbol{f}(\cdot)$ defined as (A.5),

$$||\boldsymbol{\Sigma}(\boldsymbol{x}^*)|| \leq \max_{i=1}^{n}(\sigma_{f_i}^2 + \sigma_i^2),$$

where σ_{f_i} and σ_i are hyperparameters for the kernel function corresponding to $f_i(\cdot)$, the ith element of $\boldsymbol{f}(\cdot)$.

Proof. From (A.4) and the positive definiteness of $\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X})+\sigma^2\boldsymbol{I}_N$ and the definition of SE kernel (A.6), we have

$$\Sigma(\boldsymbol{x}^*) \leq k(\boldsymbol{x}^*,\boldsymbol{x}^*) \leq \sigma_f^2 + \sigma^2.$$

Since $\boldsymbol{\Sigma}(\boldsymbol{x}^*) = \mathrm{diag}[\Sigma_1(\boldsymbol{x}^*),...,\Sigma_n(\boldsymbol{x}^*)]$, by the definition of matrix norm

$$||\boldsymbol{\Sigma}(\boldsymbol{x}^*)|| = \lambda_{\max}(\boldsymbol{\Sigma}(\boldsymbol{x}^*)) = \max_{i=1}^{n} \Sigma_i(\boldsymbol{x}^*) \leq \max_{i=1}^{n}(\sigma_{f_i}^2 + \sigma_i^2).$$

This proofs the lemma. □

For hyperparameter learning, the log likelihood of the observed data is optimized with respect to the hyperparameters.

$$\log(\boldsymbol{y}|\boldsymbol{X}) = -\frac{1}{2}\boldsymbol{y}^T(\boldsymbol{K}+\sigma^2\boldsymbol{I}_N)^{-1}\boldsymbol{y} - \frac{1}{2}\log|\boldsymbol{K}+\sigma^2\boldsymbol{I}_N| - \frac{N}{2}\log(2\pi) \tag{A.8}$$

To avoid overfit, regulation terms that penalizing large value of hyperparameters are added to the optimization objectives.

A.3 Boundedness of learning error

Suppose we would like to learn a scalar function $y = f(\boldsymbol{x})$ with input $\boldsymbol{x} \in D \subseteq \mathbb{R}^d$, where D is a compact set and output $y \in \mathbb{R}^1$. The unknown function $f(\boldsymbol{x})$ is in the reproducing kernel Hilbert space (RKHS), denoted as $f \in \mathcal{H}_k(D)$, with bounded RKHS norm $||f||_k$ with respect to the covariance function $k(\boldsymbol{x},\boldsymbol{x}')$. The output observation is with noise $\omega \sim \mathcal{N}(0,\sigma^2)$. The training data set contains N observations of input output pairs $\{\boldsymbol{x}^i, y^i\}_{i=1}^N$. For a testing point $\boldsymbol{x}$, the predictive distribution conditioned on N observations is $\mathcal{N}(\mu(\boldsymbol{x}),\Sigma(\boldsymbol{x}))$. The following lemma gives the learning error bound.

Lemma A.2. *Theorem 6 of [78]. Let $\delta \in (0,1)$, then*

$$\Pr\{|\mu(\boldsymbol{x}) - f(\boldsymbol{x})| \leq \beta\Sigma^{\frac{1}{2}}(\boldsymbol{x})\} \geq 1-\delta \tag{A.9}$$

with $\beta = \sqrt{2||f||_k^2 + 300\gamma \ln^3(\frac{N+1}{\delta})}$. $\gamma \in \mathbb{R}$ *is the maximum information gain defined as*

$$\gamma = \max_{\boldsymbol{x}^1,...,\boldsymbol{x}^{N+1} \in D} \mathbf{I}(y^1, ..., y^{N+1}; f). \tag{A.10}$$

Letting $\boldsymbol{y} := \{y^1, ..., y^{N+1}\}$ denote the $N+1$ observations of function f, and the information gain is defined as

$$\mathbf{I}(\boldsymbol{y}; f) = H(\boldsymbol{y}) - H(\boldsymbol{y}|f) \tag{A.11}$$

to quantify the reduction in uncertainty about f by revealing $\boldsymbol{y}$, where $H(\cdot)$ is the entropy. In GP context, from the prior distribution $\boldsymbol{y} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}+\sigma^2\boldsymbol{I}_{N+1})$ and the conditional distribution $\boldsymbol{y}|f \sim \mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I}_{N+1})$, the entropies are $H(\boldsymbol{y}) = \frac{1}{2}\log\{\det[2\pi e(\boldsymbol{K}+\sigma^2\boldsymbol{I}_{N+1})]\}$ and $H(\boldsymbol{y}|f) = \frac{1}{2}\log(2\pi e\sigma^2)$. Therefore, the information gain is $\mathbf{I}(\boldsymbol{y}; f) = \frac{1}{2}\log\det(\boldsymbol{I}_{N+1} + \sigma^{-2}\boldsymbol{K})$. In GP context, (A.10) can be specifically written as

$$\gamma = \max_{\boldsymbol{x}^1,...,\boldsymbol{x}^{N+1} \in D} \frac{1}{2}\log\det(\boldsymbol{I}_{N+1} + \sigma^{-2}\boldsymbol{K}). \tag{A.12}$$

The maximum information gain γ is obtained by taking optimization with respect to $\boldsymbol{x}^1, ..., \boldsymbol{x}^{N+1} \in D$, and therefore, γ is not dependent on particular chosen of $\boldsymbol{x}^i$. The details and proof of this lemma is referred to [78].

For n-dimensional vector function $\boldsymbol{f}(\boldsymbol{x})$, where each dimension is independent of each other, Lemma A.2 can be extended to the following lemma:

Lemma A.3. *Lemma 1 of [70].*

$$\Pr\{||\boldsymbol{\mu}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x})|| \leq ||\boldsymbol{\beta}^T\boldsymbol{\Sigma}^{\frac{1}{2}}(\boldsymbol{x})||\} \geq (1-\delta)^n, \tag{A.13}$$

where $\boldsymbol{\mu}(\cdot)$ *and* $\boldsymbol{\Sigma}(\cdot)$ *are defined in (A.5).* $\boldsymbol{\beta}$ *is an* $n \times 1$ *vector with its* i*th element* $\beta_i = \sqrt{2||f_i||_k^2 + 300\gamma_i \ln^3(\frac{N+1}{\delta})}$. γ_i *is the maximum information gain for* f_i.

Proof. Define the probabilistic events

$$\Pi_i = \{\boldsymbol{x} \in D, |\mu_i(\boldsymbol{x}) - f_i(\boldsymbol{x})| \leq \beta_i \Sigma_i^{\frac{1}{2}}(\boldsymbol{x})\} \tag{A.14}$$

for $i = 1, ..., n$. From Lemma A.2, for any $0 < \delta < 1$, we have $\Pr\{\Pi_i\} \geq 1 - \delta$. Since Π_i are independent from each other, $\Pr\{\cap_{i=1}^n \Pi_i\} \geq (1-\delta)^n$. Defining the probabilistic event

$$\Pi = \{\boldsymbol{x} \in D, \|\boldsymbol{\mu}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x})\| \leq \|\boldsymbol{\beta}^T \boldsymbol{\Sigma}^{\frac{1}{2}}(\boldsymbol{x})\|\}, \tag{A.15}$$

we have $\{\cap_{i=1}^n \Pi_i\} \subseteq \Pi$. Therefore $\Pr\{\Pi\} \geq \Pr\{\cap_{i=1}^n \Pi_i\} \geq (1 - \delta)^n$. □

This lemma is first proposed and proven in [70].

Appendix B
Proofs for Results in Chapter 4

B.1 Proof of Lemma 4.1

Using (4.7), we have $\boldsymbol{Y} = \boldsymbol{b}\boldsymbol{X}_d$, where $\boldsymbol{X}_d \in \mathbb{R}^{1\times M}$ is the row vector that contains the variables in $\mathrm{null}(\boldsymbol{H}_\theta)$ corresponding to each data point. Because of dimension of scalar variable x_d, the first eigenvector (i.e., the first and only principal component) by PCA is given as [79]

$$\boldsymbol{w}_1 = \arg_{\|\boldsymbol{w}\|=1} \max\{\boldsymbol{w}^T\boldsymbol{Y}\boldsymbol{Y}^T\boldsymbol{w}\} = \arg_{\|\boldsymbol{w}\|=1} \max\{\|\boldsymbol{b}^T\boldsymbol{w}\|^2\} = \boldsymbol{b},$$

where we use $\|\boldsymbol{b}\| = 1$ in the last step. Therefore, the PCA results in estimate of $\boldsymbol{b}$ exactly and this proves Lemma 4.1.

B.2 Proof of Lemma 4.2

Without loss of generality, we assume that the regression is conducted for each individual dimension. We denote that $\boldsymbol{y} \in \mathbb{R}^{M\times 1}$ be the transpose of one row of $\boldsymbol{Y}$. Since the right hand side of (4.14) is a spline function, we can write an element y of $\boldsymbol{y}$ as $y = y(\xi_1) = \sum_j^N w_j\phi_j(\xi_1) + n = \boldsymbol{\phi}^T(\xi_1)\boldsymbol{w} + n$, where $n \sim \mathcal{N}(0, \sigma^2)$ and N is the dimension of the feature space. Then for M data points, we have

$$\boldsymbol{y} = \boldsymbol{\Phi}^T\boldsymbol{w} + \boldsymbol{\epsilon}, \tag{B.1}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$, $\boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{\phi}(\xi_1(1)) & \cdots & \boldsymbol{\phi}(\xi_1(M)) \end{bmatrix}$ can be written as

$$\boldsymbol{\Phi} = \begin{bmatrix} \phi_1(\xi_1(1)) & \cdots & \phi_1(\xi_1(M)) \\ \phi_2(\xi_1(1)) & \cdots & \phi_2(\xi_1(M)) \\ \cdots & \cdots & \cdots \\ \phi_N(\xi_1(1)) & \cdots & \phi_N(\xi_1(M)) \end{bmatrix}_{N \times M}$$

and $\boldsymbol{\phi}(\xi_1(j)) = [\phi_i(\xi_1(j)) \ \cdots \ \phi_i(\xi_1(j))]^T$, $j = 1, \cdots, M$.

We now apply the minimization to the collected data to obtain parameter $\boldsymbol{\beta} = [\beta_1 \, \beta_2 \, \beta_3]$ as

$$\min_{\boldsymbol{\beta}} p(\boldsymbol{\beta}) := -\log P(\boldsymbol{y}|\Xi_1, \boldsymbol{\beta}) = \frac{1}{2}\boldsymbol{y}^T \boldsymbol{K}_y^{-1} \boldsymbol{y} + \frac{1}{2}\log|\boldsymbol{K}_y| + \frac{M}{2}\log(2\pi), \quad \text{(B.2)}$$

where $\boldsymbol{K}_{i,j} = k(\xi_1(i), \xi_1(j))$ and

$$\boldsymbol{K}_y = \boldsymbol{K} + \beta_3 \boldsymbol{I}_{M \times M} \quad \text{(B.3)}$$

If we choose the Gaussian kernel, we have $k(\xi_1(i), \xi_1(j)) = \beta_1 \exp\{-\frac{\beta_2}{2}\|\xi_1(i) - \xi_1(j)\|^2\}$. At the minimization point, the derivative with respect to $\boldsymbol{K}_y$ should be zero, namely,

$$\frac{\partial p(\boldsymbol{\beta})}{\partial \boldsymbol{K}_y} = \frac{1}{2}\left(-\boldsymbol{K}_y^{-1}\boldsymbol{y}\boldsymbol{y}^T\boldsymbol{K}_y^{-T} + \boldsymbol{K}_y^{-1}\right) = \mathbf{0}. \quad \text{(B.4)}$$

Thus, we have $\boldsymbol{K}_y = \boldsymbol{y}\boldsymbol{y}^T$. However, $\boldsymbol{y}\boldsymbol{y}^T$ is not invertible. To deal with this problem, let's assume that the minimization algorithm finds $\boldsymbol{K}_y = E[\boldsymbol{y}\boldsymbol{y}^T]$ as the solution. From (B.1), we have $\boldsymbol{K}_y = E[\boldsymbol{y}\boldsymbol{y}^T] = \boldsymbol{\Phi}^T \boldsymbol{w}\boldsymbol{w}^T \Phi + \sigma^2 \boldsymbol{I}_{M \times M}$. Comparing with (B.3) we have that $k(\xi_1(i), \xi_1(j)) = \boldsymbol{\phi}^T(\xi_1(i))\boldsymbol{w}\boldsymbol{w}^T\boldsymbol{\phi}(\xi_1(j))$. Once the kernel is determined, the predictive posterior distribution for a new input ξ_* is given by $p(f|\xi_*, \boldsymbol{y}, \Xi_1) = \mathcal{N}(\mu_g, \Sigma_g)$ with

$$\mu_g(\xi_*) = \boldsymbol{k}_*^T(\boldsymbol{K}_y)^{-1}\boldsymbol{y}, \ \Sigma_g(\xi_*) = k_{**} - \boldsymbol{k}_*^T(\boldsymbol{K}_y)^{-1}\boldsymbol{k}_*,$$

where $\boldsymbol{k}_* = [k(\xi_1(1), \xi_*), \cdots, k(\xi_1(M), \xi_*)]^T$ and $k_{**} = k(\xi_*, \xi_*)$. Therefore, we have $\boldsymbol{k}_* = \boldsymbol{\Phi}^T \boldsymbol{w}\boldsymbol{w}^T \boldsymbol{\phi}(\xi_*)$. Multiplying both sides of the mean value prediction function by $\boldsymbol{y}^T$, we obtain $\mu_g(\xi_*)\boldsymbol{y}^T = \boldsymbol{k}_*^T(\boldsymbol{K}_y)^{-1}\boldsymbol{y}\boldsymbol{y}^T$. With equation (B.4), it can be reformed as $\mu_g(\xi_*)\boldsymbol{y}^T = \boldsymbol{k}_*^T = \boldsymbol{\phi}^T(\xi_*)\boldsymbol{w}\boldsymbol{w}^T\boldsymbol{\Phi}$. We combine this relation with (B.1) and have $(\mu_g(\xi_*) - \boldsymbol{\phi}^T(\xi_*)\boldsymbol{w})\boldsymbol{y}^T = -\boldsymbol{\phi}^T(\xi_*)\boldsymbol{w}\boldsymbol{\epsilon}^T$. Because $\boldsymbol{y}^T$ is nonzero, when $\boldsymbol{\epsilon}$ is small, we have $\mu_g(\xi_*) = \boldsymbol{\phi}^T(\xi_*)\boldsymbol{w}$. This implies that by minimizing (B.2), the mean value of the posterior predictive distribution recovers the nominal generating model (B.1). We apply (B.2) to each row of $\boldsymbol{Y}$ to recover each dimension of the equation (4.14). This completes the proof.

Appendix C
Proofs for Results in Chapter 5

C.1 Proof of Lemma 5.1

For the brevity of the derivation, we omit the input $\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{u}_f$ in function $\boldsymbol{\mu}_\alpha$ and $\boldsymbol{\kappa}_\alpha$.

$$\begin{aligned} &\boldsymbol{\mu}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f) - \boldsymbol{\kappa}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f) \\ =&\boldsymbol{\mu}_\alpha(\boldsymbol{v}) - \boldsymbol{\kappa}_\alpha(\boldsymbol{v}) + \boldsymbol{\kappa}_\alpha(\boldsymbol{v}) - \boldsymbol{\kappa}_\alpha(\dot{\boldsymbol{\alpha}}_2) \\ =&\boldsymbol{\mu}_\alpha(\boldsymbol{v}) - \boldsymbol{\kappa}_\alpha(\boldsymbol{v}) - \frac{\partial \boldsymbol{\kappa}_\alpha}{\partial \boldsymbol{v}}(\dot{\boldsymbol{\alpha}}_2 - \boldsymbol{v}) + O(||\dot{\boldsymbol{\alpha}}_2 - \boldsymbol{v}||^2) \\ =&\boldsymbol{\mu}_\alpha(\boldsymbol{v}) - \boldsymbol{\kappa}_\alpha(\boldsymbol{v}) - \frac{\partial \boldsymbol{\kappa}_\alpha}{\partial \boldsymbol{v}}[\boldsymbol{\mu}_\alpha(\boldsymbol{v}) - \boldsymbol{\kappa}_\alpha(\dot{\boldsymbol{\alpha}}_2)] + O(||\dot{\boldsymbol{\alpha}}_2 - \boldsymbol{v}||^2). \end{aligned} \tag{C.1}$$

The second equality sign comes from Taylor expansion of $\boldsymbol{\kappa}_\alpha(\boldsymbol{v}) - \boldsymbol{\kappa}_\alpha(\dot{\boldsymbol{\alpha}}_2)$. We have $O(||\dot{\boldsymbol{\alpha}}_2 - \boldsymbol{v}||^2) \leq c_2||\boldsymbol{e}_\alpha||^2 + c_1||\boldsymbol{e}_\alpha|| + c_0$, with constants $c_0, c_1, c_2 \in \mathbb{R}^+$. It should be noted that if (5.17) is in the form of robot manipulator dynamics, $\boldsymbol{\kappa}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f)$ is exactly linear function of $\dot{\boldsymbol{\alpha}}_2$ and $O(||\dot{\boldsymbol{\alpha}}_2 - \boldsymbol{v}||^2) = 0$. The third equality sign comes from (5.22).

Let $\boldsymbol{A}_\kappa := \boldsymbol{I} + \frac{\partial \boldsymbol{\kappa}_\alpha}{\partial \boldsymbol{v}}$ be the linearization of the left hand side of $\boldsymbol{\alpha}_2$ subdynamics in (5.17) with respect to $\dot{\boldsymbol{\alpha}}_2$. It serves as the inertia matrix of $\boldsymbol{\alpha}_2$ subdynamics in the robot manipulator dynamics. Therefore, $\boldsymbol{A}_\kappa$ is non-singular and positive definite. From (C.1), we have

$$\boldsymbol{\mu}_\alpha(\boldsymbol{v}) - \boldsymbol{\kappa}_\alpha(\dot{\boldsymbol{\alpha}}_2) = \boldsymbol{A}_\kappa^{-1}[O(||\dot{\boldsymbol{\alpha}}_2 - \boldsymbol{v}||^2) + \boldsymbol{\mu}_\alpha(\boldsymbol{v}) - \boldsymbol{\kappa}_\alpha(\boldsymbol{v})]. \tag{C.2}$$

Taking the norm and applying Lemma A.3 to $\boldsymbol{\mu}_\alpha(\boldsymbol{v}) - \boldsymbol{\kappa}_\alpha(\boldsymbol{v})$, we have

$$||\boldsymbol{\mu}_\alpha(\boldsymbol{v}) - \boldsymbol{\kappa}_\alpha(\dot{\boldsymbol{\alpha}}_2)|| \leq \lambda_{\min}^{-1}(\boldsymbol{A}_\kappa)(\sum_{i=0}^{2} c_i||\boldsymbol{e}_\alpha||^i + ||\boldsymbol{\beta}_\alpha^T \boldsymbol{\Sigma}_\alpha^{\frac{1}{2}}(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f)||) \tag{C.3}$$

with the probability $(1-\delta)^n$, where $\lambda_{\min}(\boldsymbol{A}_\kappa) > 0$ is the smallest eigenvalue of $\boldsymbol{A}_\kappa$. Letting $\rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f)$ denote the right hand side of the less or equal sign, i.e.,

$$\rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f) := \lambda_{\min}^{-1}(\boldsymbol{A}_\kappa)(\sum_{i=0}^{2} c_i||\boldsymbol{e}_\alpha||^i + ||\boldsymbol{\beta}_\alpha^T \boldsymbol{\Sigma}_\alpha^{\frac{1}{2}}(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f)||), \tag{C.4}$$

we have $\Pr\{||\boldsymbol{\mu}_\alpha(\boldsymbol{v}) - \boldsymbol{\kappa}_\alpha(\dot{\boldsymbol{\alpha}}_2)|| \leq \rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f)\} \leq (1-\delta)^n$.

C.2 Proof of Lemma 5.2

With $k_p, k_d > 0$ and singular perturbation parameter $\epsilon > 0$, $\boldsymbol{A}$ is Hurwitz. $\boldsymbol{A}$ is with n-eigenvalues at $\frac{\lambda_1}{\epsilon}$ and the other n-eigenvalues at $\frac{\lambda_2}{\epsilon}$, where $\lambda_1 = \frac{1}{2}(-k_d + \sqrt{k_d^2 - 4k_p})$ and $\lambda_2 = \frac{1}{2}(-k_d - \sqrt{k_d^2 - 4k_p})$. We further assume that $k_d^2 - 4k_p > 0$ so that $\lambda_2 < \lambda_1 < 0$. There is an invertible coordinate transformation

$$\boldsymbol{M} = \begin{bmatrix} \epsilon \boldsymbol{I}_n & \epsilon \boldsymbol{I}_n \\ \lambda_1 \boldsymbol{I}_n & \lambda_2 \boldsymbol{I}_n \end{bmatrix}, \boldsymbol{M}^{-1} = \frac{1}{\epsilon(\lambda_2 - \lambda_1)} \begin{bmatrix} \lambda_2 \boldsymbol{I}_n & -\epsilon \boldsymbol{I}_n \\ -\lambda_1 \boldsymbol{I}_n & \epsilon \boldsymbol{I}_n \end{bmatrix} \tag{C.5}$$

such that $\boldsymbol{A} = \boldsymbol{M}\boldsymbol{\Lambda}\boldsymbol{M}^{-1}$, where $\boldsymbol{\Lambda}$ is the diagonalized matrix:

$$\boldsymbol{\Lambda} = \boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{M} = \begin{bmatrix} \frac{\lambda_1}{\epsilon} \boldsymbol{I}_n & \boldsymbol{0} \\ \boldsymbol{0} & \frac{\lambda_2}{\epsilon} \boldsymbol{I}_n \end{bmatrix}. \tag{C.6}$$

In order to assess the converging property of $\boldsymbol{e}_\alpha$, a coordinate transformation is performed to $\boldsymbol{e}_\alpha$. Let $\boldsymbol{e}_\alpha = \boldsymbol{M}\boldsymbol{e}_{\alpha'}$, the $\boldsymbol{e}_\alpha$ dynamics (5.23) can be written in $\boldsymbol{e}_{\alpha'}$ coordinates:

$$\dot{\boldsymbol{e}}_{\alpha'} = \boldsymbol{\Lambda}\boldsymbol{e}_{\alpha'} + \boldsymbol{M}^{-1}\boldsymbol{B}[\boldsymbol{r}(t) + \boldsymbol{\mu}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f) - \boldsymbol{\kappa}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f)]. \tag{C.7}$$

Since $\boldsymbol{\Lambda}$ is Hurwitz, there exists a positive definite matrix $\boldsymbol{P}_{\alpha'}$

$$\boldsymbol{P}_{\alpha'} = \begin{bmatrix} -\frac{\epsilon}{2\lambda_1}\boldsymbol{I}_n & \boldsymbol{0} \\ \boldsymbol{0} & -\frac{\epsilon}{2\lambda_2}\boldsymbol{I}_n \end{bmatrix} \tag{C.8}$$

such that $\boldsymbol{\Lambda}^T\boldsymbol{P}_{\alpha'} + \boldsymbol{P}_{\alpha'}\boldsymbol{\Lambda} = -\boldsymbol{I}_{2n}$. We choose the Lyapunov function $V_{\alpha'} = \boldsymbol{e}_{\alpha'}^T\boldsymbol{P}_{\alpha'}\boldsymbol{e}_{\alpha'}$ for $\boldsymbol{e}_{\alpha'}$ dynamics (C.7). Note that

$$-\frac{\epsilon}{2\lambda_2}||\boldsymbol{e}_{\alpha'}||^2 \leq V_{\alpha'} \leq -\frac{\epsilon}{2\lambda_1}||\boldsymbol{e}_{\alpha'}||^2, \tag{C.9}$$

then

$$\dot{V}_{\alpha'} = -\boldsymbol{e}_{\alpha'}^T\boldsymbol{e}_{\alpha'} + 2(\boldsymbol{B}^T\boldsymbol{M}^{-T}\boldsymbol{P}_{\alpha'}\boldsymbol{e}_{\alpha'})^T[\boldsymbol{r} + \boldsymbol{\mu}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f) - \boldsymbol{\kappa}_\alpha(\boldsymbol{\theta}, \boldsymbol{\alpha}, \dot{\boldsymbol{\alpha}}_2, \boldsymbol{u}_f)]. \tag{C.10}$$

$\boldsymbol{r}(t)$ is designed to be

$$\boldsymbol{r}(t) = \begin{cases} -\rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta})\frac{\boldsymbol{B}^T\boldsymbol{M}^{-T}\boldsymbol{P}_{\alpha'}\boldsymbol{e}_{\alpha'}}{||\boldsymbol{B}^T\boldsymbol{M}^{-T}\boldsymbol{P}_{\alpha'}\boldsymbol{e}_{\alpha'}||}; & \text{if } ||\boldsymbol{B}^T\boldsymbol{M}^{-T}\boldsymbol{P}_{\alpha'}\boldsymbol{e}_{\alpha'}|| > \xi \\ -\frac{\rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta})}{\xi}\boldsymbol{B}^T\boldsymbol{M}^{-T}\boldsymbol{P}_{\alpha'}\boldsymbol{e}_{\alpha'}; & \text{if } ||\boldsymbol{B}^T\boldsymbol{M}^{-T}\boldsymbol{P}_{\alpha'}\boldsymbol{e}_{\alpha'}|| \leq \xi, \end{cases} \tag{C.11}$$

where $\xi > 0$ is designed as $\xi = \frac{\lambda_{\min}(\boldsymbol{A}_\kappa)}{c_2\|\boldsymbol{M}\|^2}$. When $||\boldsymbol{\mu}_\alpha(\boldsymbol{v}) - \boldsymbol{\kappa}_\alpha(\dot{\boldsymbol{\alpha}}_2)|| \leq \rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{v}, \boldsymbol{u}_f)$, according to [80] (Chapter 8.4.1 Theorem 1), we have

$$\begin{aligned} \dot{V}_{\alpha'} &\leq -\|\boldsymbol{e}_{\alpha'}\|^2 + \frac{\xi\rho(\boldsymbol{e}_\alpha, \boldsymbol{\theta})}{2} = -\frac{1}{2}\|\boldsymbol{e}_{\alpha'}\|^2 + \frac{c_1}{2c_2\|\boldsymbol{M}\|}\|\boldsymbol{e}_{\alpha'}\| + \frac{c_0}{2c_2\|\boldsymbol{M}\|^2} + \frac{\|\beta_\alpha^T\boldsymbol{\Sigma}_\alpha^{1/2}\|}{2c_2\|\boldsymbol{M}\|^2} \\ &= -\frac{1}{4}\|\boldsymbol{e}_{\alpha'}\|^2 - \frac{1}{4}\left(\|\boldsymbol{e}_{\alpha'}\| - \frac{c_1}{c_2\|\boldsymbol{M}\|}\right)^2 + c_3 \leq -\frac{1}{4}\|\boldsymbol{e}_{\alpha'}\|^2 + c_3, \end{aligned} \tag{C.12}$$

where positive constant $c_3 = \frac{1}{4}\frac{c_1^2}{c_2^2\|\boldsymbol{M}\|^2} + \frac{1}{2}\frac{c_0}{c_2\|\boldsymbol{M}\|^2} + \frac{1}{2}\frac{\|\beta_\alpha^T\boldsymbol{\Sigma}_\alpha^{1/2}\|}{2c_2\|\boldsymbol{M}\|^2} > 0$. By combining (C.9) with (C.12), we have

$$\dot{V}_{\alpha'} \leq \frac{\lambda_1}{2\epsilon}V_{\alpha'} + c_3 \;\Rightarrow\; V_{\alpha'}(t) \leq V_{\alpha'}(0)e^{\frac{\lambda_1}{2\epsilon}t} - \frac{2\epsilon}{\lambda_1}c_3. \tag{C.13}$$

We define Lyapuov function candidate $V_\alpha(t) = \boldsymbol{e}_\alpha(t)^T\boldsymbol{P}\boldsymbol{e}_\alpha(t)$, where $\boldsymbol{P} = \boldsymbol{M}^{-T}\boldsymbol{P}_{\alpha'}\boldsymbol{M}^{-1}$ is positive definite because of positive definite $\boldsymbol{P}_{\alpha'}$. It is straightforward to check that $V_\alpha(t) = \boldsymbol{e}_{\alpha'}(t)^T \boldsymbol{M}^T \boldsymbol{P} \boldsymbol{M} \boldsymbol{e}_{\alpha'}(t) = \boldsymbol{e}_{\alpha'}(t)^T \boldsymbol{P}_{\alpha'} \boldsymbol{e}_{\alpha'}(t) = V_{\alpha'}(t)$. Defining $\boldsymbol{Q} = \boldsymbol{M}^{-T}\boldsymbol{M}^{-1}$, $\boldsymbol{P}$ is the solution of Lyapunov equation $\boldsymbol{A}^T\boldsymbol{P}+\boldsymbol{P}\boldsymbol{A} = -\boldsymbol{Q}$. Using (C.12), we obtain

$$\dot{V}_\alpha \leq -\frac{1}{4}\boldsymbol{e}_\alpha^T \boldsymbol{Q} \boldsymbol{e}_\alpha + c_3. \tag{C.14}$$

Using $\boldsymbol{P}$ and $\boldsymbol{e}_\alpha$, we write the control input $\boldsymbol{r}(t)$ as in (5.28).

Using the fact that $\lambda_{\min}(\boldsymbol{P})\|\boldsymbol{e}_\alpha\|^2 \leq V_\alpha(t) \leq \lambda_{\max}(\boldsymbol{P})\|\boldsymbol{e}_\alpha\|^2$, from (C.13), we have

$$\begin{aligned} \|\boldsymbol{e}_\alpha(t)\| &\leq \sqrt{\frac{\lambda_{\max}(\boldsymbol{P})}{\lambda_{\min}(\boldsymbol{P})}}\|\boldsymbol{e}_\alpha(0)\| e^{\frac{\lambda_1}{4\epsilon}t} + \sqrt{-\frac{2\epsilon c_3}{\lambda_1 \lambda_{\min}(\boldsymbol{P})}} \\ &= d_1\|\boldsymbol{e}_\alpha(0)\| e^{\frac{\lambda_1}{4\epsilon}t} + d_2 \end{aligned}$$

with d_1 and d_2 are given in the lemma. Then, considering the learning model error bound in c_3 and applying the results in Lemma 5.1, we prove (5.29).

C.3 Proof of Lemma 5.3

The iterative relation of the covariance is given in (5.35). Taking norm on both sides and applying the upper bound of the gradient $||\frac{\partial \boldsymbol{\mu}_{gp_\theta}}{\partial \boldsymbol{\theta}}|| \leq L_1$,

$$\begin{aligned} ||\boldsymbol{\Sigma}_{\hat{\theta}}(k+i+1|k)|| &\leq (||\boldsymbol{F}||^2 + ||\boldsymbol{G}||^2 L_1^2)||\boldsymbol{\Sigma}_{\hat{\theta}}(k+i|k)|| + ||\boldsymbol{G}||^2||\boldsymbol{\Sigma}_{gp_\theta}|| \\ &\leq (||\boldsymbol{F}||^2 + ||\boldsymbol{G}||^2 L_1^2)||\boldsymbol{\Sigma}_{\hat{\theta}}(k+i|k)|| + ||\boldsymbol{G}||^2 \sigma^2_{\boldsymbol{f}\,\max}. \end{aligned} \tag{C.15}$$

Applying (C.15) iteratively with the initial condition $\boldsymbol{\Sigma}_{\hat{\theta}}(k|k) = \boldsymbol{0}$, we obtain

$$||\boldsymbol{\Sigma}_{\hat{\theta}}(k+i|k)|| \leq \frac{1-(||\boldsymbol{F}||^2 + ||\boldsymbol{G}||^2 L_1^2)^i}{1-(||\boldsymbol{F}||^2 + ||\boldsymbol{G}||^2 L_1^2)}||\boldsymbol{G}||^2 \sigma^2_{\boldsymbol{f}\,\max}. \tag{C.16}$$

The norms of $\boldsymbol{F}$ and $\boldsymbol{G}$ are calculated as

$$\|\boldsymbol{F}\| = \sqrt{1 + \frac{\Delta t}{2}\left(\Delta t + \sqrt{(\Delta t)^2 + 4}\right)}, \ \|\boldsymbol{G}\| = \Delta t.$$

For $\Delta t \ll 1$, taking approximation $\|\boldsymbol{F}\| \approx 1$ and $(1+x)^n \approx 1 + nx$ with $|x| \ll 1$, we obtain the upper-bound as shown in the lemma.

C.4 Proof of Lemma 5.4

It is straightforward to obtain

$$\begin{aligned} l_f(k+H+2) &= l_f^*(k+H+2) + \mathrm{tr}(\boldsymbol{Q}_3 \boldsymbol{\Sigma}_{\hat{\theta}}(k+H+2)) \\ &\leq l_f^*(k+H+1) - l_s^*(k+H+1) + \mathrm{tr}(\boldsymbol{Q}_3 \boldsymbol{\Sigma}_{\hat{\theta}}(k+H+2)) \\ &\leq l_f(k+H+1) - l_s^*(k+H+1) + \mathrm{tr}(\boldsymbol{Q}_3 \boldsymbol{\Sigma}_{\hat{\theta}}(k+H+2)). \end{aligned}$$

Under (5.48), we obtain $l_s(k+H+1) \leq l_s^*(k+H+1) + \mathrm{tr}(\boldsymbol{Q}_1 \boldsymbol{\Sigma}_{\hat{\theta}}(k+H+1))$ and combining with the above inequality, the proof is completed.

C.5 Proof of Lemma 5.5

We first show the decreasing property of the proposed Lyapunov function candidate $J^k_{\hat{\theta}^0, \hat{W}^0_H}$. To do that, we take the technique to construct an intermediary policy $\hat{\boldsymbol{W}}^e_H(k+1)$ (defined below) extended from $\hat{\boldsymbol{W}}^0_H(k)$ as shown in [65]. From the definition of $\hat{\boldsymbol{W}}^0_H(k) = \{\hat{\boldsymbol{\alpha}}^0(k), \hat{\boldsymbol{w}}^0(k+i), \boldsymbol{u}_f^0(k+i), i = 0, \ldots, H\}$, we introduce an extended control input at the $(k+1)$th step as

$$\hat{\boldsymbol{W}}^e_H(k+1) = \{\hat{\boldsymbol{\alpha}}^e(k+1), \hat{\boldsymbol{w}}^e(k+i+1), \boldsymbol{u}_f^e(k+i+1), i = 0, \ldots, H\},$$

where $\hat{\boldsymbol{\alpha}}_1^e(k+1) = \hat{\boldsymbol{\alpha}}_1^0(k) + \hat{\boldsymbol{\alpha}}_2^0(k)\Delta t$, $\hat{\boldsymbol{\alpha}}_2^e(k+1) = \hat{\boldsymbol{\alpha}}_2^0(k) + \hat{\boldsymbol{w}}^0(k)\Delta t$, and $\hat{\boldsymbol{w}}^e(k+i) = \hat{\boldsymbol{w}}^0(k+i)$, $\boldsymbol{u}_f^e(k+i) = \boldsymbol{u}_f^0(k+i)$ for $i = 1, ..., H$, and $\hat{\boldsymbol{w}}^e(k+H+1)$ and $\boldsymbol{u}_f^e(k+H+1)$ satisfy (5.48). The choice of such design guarantees that inputs $\{\hat{\boldsymbol{\alpha}}^e(k+i)$,$\hat{\boldsymbol{w}}^e(k+i)$,$\boldsymbol{u}_f^e(k+i)\}$ of $\hat{\boldsymbol{W}}^e_H(k+1)$ are the same as $\{\hat{\boldsymbol{\alpha}}^0(k+i)$,$\hat{\boldsymbol{w}}^0(k+i)$,$\boldsymbol{u}_f^0(k+i)\}$ of $\hat{\boldsymbol{W}}^0_H(k)$ for $i = 1, \ldots, H$. Consequently, the predicted states $\boldsymbol{\mu}^e_{\hat{\theta}}(k+i)$, $\boldsymbol{\Sigma}^e_{\hat{\theta}}(k+i)$ by (5.35) under $\hat{\boldsymbol{W}}^e_H(k+1)$ are the same as those under control $\hat{\boldsymbol{W}}^0_H(k)$ at those steps. Let $l_s^e(k+i)$ ($l_f^e(k+i)$) and $l_s^0(k+i)$ ($l_f^0(k+i)$) denote the stage and terminal costs under controls $\hat{\boldsymbol{W}}^e_H(k+1)$ and $\hat{\boldsymbol{W}}^0_H(k)$, respectively. It is then straightforward to obtain that

$l^e_s(k+i) = l^0_s(k+i)$ for $i = 1, \ldots, H$ and $l^e_f(k+H+1) = l^0_f(k+H+1)$. Therefore,

$$\begin{aligned} J^{k+1}_{\hat{\theta}^e, \hat{W}^e_H} - J^{k}_{\hat{\theta}^0, \hat{W}^0_H} &= l^e_s(k+H+1) + l^e_f(k+H+2) \\ &\quad - l^0_s(k) - l^e_f(k+H+1) + \nu \Delta \mathbf{\Sigma}^{e0}_{dk} + \Delta \hat{\boldsymbol{\alpha}}^0_{\boldsymbol{Q}_2 k}, \end{aligned}$$

where $\Delta \mathbf{\Sigma}^{e0}_{dk} = \|\mathbf{\Sigma}_d(\hat{W}^e_H(k+1))\| - \|\mathbf{\Sigma}_d(\hat{W}^0_H(k))\|$ and $\Delta \hat{\boldsymbol{\alpha}}^0_{\boldsymbol{Q}_2 k} = \|\hat{\boldsymbol{\alpha}}^e(k+1)\|^2_{\boldsymbol{Q}_2} - \|\hat{\boldsymbol{\alpha}}^0(k)\|^2_{\boldsymbol{Q}_2}$. Noticing that $\hat{\boldsymbol{w}}^e(k+H+1)$ and $\boldsymbol{u}^e_f(k+H+1)$ satisfy (5.48), by Lemma 5.4, we have

$$\begin{aligned} J^{k+1}_{\hat{\theta}^e, \hat{W}^e_H} - J^{k}_{\hat{\theta}^0, \hat{W}^0_H} &\leq -l^0_s(k) + \nu \Delta \mathbf{\Sigma}^{e0}_{dk} + \Delta \hat{\boldsymbol{\alpha}}^0_{\boldsymbol{Q}_2 k} \\ &\quad + \operatorname{tr}(\boldsymbol{Q}_1 \mathbf{\Sigma}_{\hat{\theta}}(k+H+1)) + \operatorname{tr}(\boldsymbol{Q}_3 \mathbf{\Sigma}_{\hat{\theta}}(k+H+2)). \end{aligned}$$

From the optimality of $J^{k+1}_{\hat{\theta}^0, \hat{W}^0_H}$, we have $J^{k+1}_{\hat{\theta}^0, \hat{W}^0_H} \leq J^{k+1}_{\hat{\theta}^e, \hat{W}^e_H}$ and then

$$\begin{aligned} J^{k+1}_{\hat{\theta}^0, \hat{W}^0_H} &- J^{k}_{\hat{\theta}^0, \hat{W}^0_H} \leq -\lambda_{\min}(\boldsymbol{Q}_1)\|\boldsymbol{e}_\theta(k)\|^2 + \nu \Delta \mathbf{\Sigma}^{e0}_{dk} \\ &+ \Delta \hat{\boldsymbol{\alpha}}^0_{\boldsymbol{Q}_2 k} + \operatorname{tr}(\boldsymbol{Q}_1 \mathbf{\Sigma}_{\hat{\theta}}(k+H+1)) + \operatorname{tr}(\boldsymbol{Q}_3 \mathbf{\Sigma}_{\hat{\theta}}(k+H+2)). \end{aligned} \tag{C.17}$$

From Lemma 5.3, $\mathbf{\Sigma}_{\hat{\theta}}(k+H+1) \leq (H+1)(\Delta t)^2 \sigma^2_{f\,\max}$. From Lemma A.1, $\Delta \mathbf{\Sigma}^{e0}_{dk} \leq \|\mathbf{\Sigma}_d(\hat{W}^e_H(k+1))\| \leq \max_{1 \leq i \leq n}(\sigma^2_{\alpha_i} + \sigma^2_i) := \sigma^2_{\kappa\,\max}$. Letting $\|\boldsymbol{\alpha}(k+1)\|^2_{\boldsymbol{Q}_2} \leq \alpha^2_{\max}$ as the constant upper-bound, we have $\Delta \hat{\boldsymbol{\alpha}}^0_{\boldsymbol{Q}_2 k} \leq \|\hat{\boldsymbol{\alpha}}^e(k+1)\|^2_{\boldsymbol{Q}_2} \leq \alpha^2_{\max}$ and

$$\begin{aligned} J^{k+1}_{\hat{\theta}^0, \hat{W}^0_H} &- J^{k}_{\hat{\theta}^0, \hat{W}^0_H} \leq -\lambda_{\min}(\boldsymbol{Q}_1)\|\boldsymbol{e}_\theta(k)\|^2 + \nu \sigma^2_{\boldsymbol{\kappa}\,\max} \\ &+ \alpha^2_{\max} + m[\lambda_{\max}(\boldsymbol{Q}_1) + \lambda_{\max}(\boldsymbol{Q}_3)](H+2)(\Delta t)^2 \sigma^2_{\boldsymbol{f}\,\max}. \end{aligned} \tag{C.18}$$

By the monotonicity of the value function (Lemma 2.15 in [65]), we have

$$\begin{aligned} J^{k}_{\hat{\theta}^0, \hat{W}^0_H} &\leq l_f(k) + \|\hat{\boldsymbol{\alpha}}^0(k)\|^2_{\boldsymbol{Q}_2} + \nu\|\mathbf{\Sigma}^{\hat{W}^0}_d(k))\| \\ &\leq \lambda_{\max}(\boldsymbol{Q}_3)\|\boldsymbol{e}_\theta(k)\|^2 + \alpha^2_{\max} + \nu \sigma^2_{\kappa\,\max}. \end{aligned}$$

Substituting the above inequalities into (C.18) to cancel $\|\boldsymbol{e}_\theta(k)\|^2$ on the right hand side of the inequality sign, we then obtain $J^{k+1}_{\hat{\theta}^0, \hat{W}^0_H} \leq d_3 J^{k}_{\hat{\theta}^0, \hat{W}^0_H} + d_4$ with $d_3 = 1 -$

$\frac{\lambda_{\min}(\boldsymbol{Q}_1)}{\lambda_{\max}(\boldsymbol{Q}_3)} < 1$ and $d_4 = (1 + \frac{\lambda_{\min}(\boldsymbol{Q}_1)}{\lambda_{\max}(\boldsymbol{Q}_3)})(\nu\sigma^2_{\kappa\max} + \alpha^2_{\max}) + m(\lambda_m)(H+2)(\Delta t)^2\sigma^2_{\boldsymbol{f}\max}$ where $\lambda_m = \lambda_{\max}(\boldsymbol{Q}_1) + \lambda_{\max}(\boldsymbol{Q}_3)$. Therefore,

$$J^{k+i}_{\hat{\theta}^0,\hat{W}^0_H} \leq d_3^i J^k_{\hat{\theta}^0,\hat{W}^0_H} + d_4 \frac{1-d_3^i}{1-d_3}, \tag{C.19}$$

and consequently, $\|\boldsymbol{e}_{\boldsymbol{\mu}_{\hat{\theta}}}(k+i)\| \leq a_4(i)\|\boldsymbol{e}_\theta(k)\| + a_5(i)$ where $a_4(i) = d_3^{\frac{i}{2}}\sqrt{\frac{\lambda_{\max}(\boldsymbol{Q}_3)}{\lambda_{\min}(\boldsymbol{Q}_1)}}$ and $a_5(i) = \sqrt{\frac{d_3^i(\alpha^2_{\max} + \nu\sigma^2_{\kappa\max}) + d_4\frac{1-d_3^i}{1-d_3}}{\lambda_{\min}(\boldsymbol{Q}_1)}}$.

C.6 Proof of Lemma 5.6

Plugging the iterative relation (5.35) for $\boldsymbol{\mu}_{\hat{\theta}}(k+i|k)$ and counterpart for $\boldsymbol{\mu}_\theta(k+i|k)$ into $\tilde{\boldsymbol{\mu}}_\theta(k+i)$, the difference is then

$$\begin{aligned}
\tilde{\boldsymbol{\mu}}_\theta(k+i) &= \|\boldsymbol{F}\|\|\tilde{\boldsymbol{\mu}}_\theta(k+i-1)\| + \|\boldsymbol{G}\|\|\boldsymbol{\mu}_{gp_\theta}(\boldsymbol{\mu}_\theta(k+i-1), \boldsymbol{\alpha}(k+i-1)) \\
&\quad -\boldsymbol{\mu}_{gp_\theta}(\boldsymbol{\mu}_{\hat{\theta}}(k+i-1|k), \hat{\boldsymbol{\alpha}}(k+i-1|k))\| \\
&\leq \|\boldsymbol{F}\|\|\tilde{\boldsymbol{\mu}}_\theta(k+i-1)\| + \|\boldsymbol{G}\|(L_3\|\tilde{\boldsymbol{\mu}}_\theta(k+i-1)\| + L_2\|\boldsymbol{e}_\alpha(k+i-1)\|) \\
&= (\|\boldsymbol{F}\| + L_3\|\boldsymbol{G}\|)\|\tilde{\boldsymbol{\mu}}_\theta(k+i-1)\| + L_2\|\boldsymbol{G}\|\|\boldsymbol{e}_\alpha(k+i-1)\|.
\end{aligned}$$

In the above derivations, we use the Lipshitz assumption. For small sampling period $\Delta t \ll 1$, $\|\boldsymbol{F}\| \approx 1$ and $\|\boldsymbol{G}\| = \Delta t$, when $i = 1$, with the fact that $\boldsymbol{\mu}_\theta(k|k) = \boldsymbol{\mu}_{\hat{\theta}}(k|k)$, we have $\tilde{\boldsymbol{\mu}}_\theta(k+1) \leq L_2\Delta t\|\boldsymbol{e}_\alpha(k)\|$. For $i \geq 2$, applying the above process iteratively, we obtain

$$\tilde{\boldsymbol{\mu}}_\theta(k+i) \;\leq\; \sum_{j=0}^{i-1}(\|\boldsymbol{F}\| + L_3\|\boldsymbol{G}\|)^{i-j-1}L_2\|\boldsymbol{G}\|\Big[d_1e^{\frac{\lambda_1}{4\epsilon}j\Delta t}\|\boldsymbol{e}_\alpha(k)\| + d_2\Big],$$

where the results in Lemma 5.2 are used to obtain $\|\boldsymbol{e}_\alpha(k+j)\| \leq d_1e^{\frac{\lambda_1}{4\epsilon}j\Delta t}\|\boldsymbol{e}_\alpha(k)\| + d_2$. Using approximation $(1 + L_3\Delta t)^{i-j-1} \approx 1 + (i-j-1)L_3\Delta t$ for small $L_3\Delta t \ll 1$, we obtain the upper-bound as shown in the lemma.

C.7 Proof of Lemma 5.7

Substituting the iterative model similar to (5.35) for both $\boldsymbol{\mu}_\theta(k+i|k)$ and $\boldsymbol{\theta}(k+i)$, the error calculation is reduced to

$$\boldsymbol{\theta}_\mu(k+i) \;=\; \boldsymbol{F}\boldsymbol{\theta}_\mu(k+i-1)+\boldsymbol{G}[\boldsymbol{\mu}_{gp_\theta}(k+i-1|k)-\boldsymbol{f}_\theta(\boldsymbol{\theta}(k+i-1))].$$

Taking the norm on both sides of the above equation, using approximation $\|\boldsymbol{F}\|\approx 1$ and $\|\boldsymbol{G}\|=\Delta t$, and applying the results in Lemma A.3 on $\|\boldsymbol{\mu}_{gp_\theta}-\boldsymbol{f}_\theta\|$, we obtain the iterative relationship of the error bound

$$\Pr\{\|\boldsymbol{\theta}_\mu(k+i)\|\le\|\boldsymbol{\theta}_\mu(k+i-1)\|+\Delta t\|\boldsymbol{\beta}_\theta^T\boldsymbol{\Sigma}_{gp_\theta}^{\frac{1}{2}}(k+i-1|k)\|\}\ge(1-\delta)^m.$$

With the initial condition $\boldsymbol{\mu}_\theta(k|k)=\boldsymbol{\theta}(k)$, by applying the above iteration i times and noting the independence of each event $\boldsymbol{\Pi}_{\mu_\theta}^\theta(k+i)$, we then obtain the results in the lemma.

C.8 Proof of Lemma 5.9

To assess $\bar{J}^k_{\hat\theta^0,\hat W^0_H}-\bar{J}^k_{\theta,\hat W^0_H}$, we substitute the definition (5.37) and (5.50) and use calculations (5.39) and (5.40)

$$\begin{aligned}
&\bar{J}^k_{\hat\theta^0,\hat W^0_H}-\bar{J}^k_{\theta,\hat W^0_H}\\
=\;&\sum_{i=0}^{H}\{\mathbb{E}[\|\boldsymbol{e}_{\hat\theta}(k+i|k)\|^2_{\boldsymbol{Q}_1}]-\|\boldsymbol{e}_\theta(k+i)\|^2_{\boldsymbol{Q}_1}\}+\mathbb{E}[\|\boldsymbol{e}_{\hat\theta}(k+H+1)\|^2_{\boldsymbol{Q}_3}]\\
&-\|\boldsymbol{e}_\theta(k+H+1)\|^2_{\boldsymbol{Q}_3}\\
=\;&\sum_{i=0}^{H}\{\|\boldsymbol{e}_{\mu_{\hat\theta}}(k+i)\|^2_{\boldsymbol{Q}_1}+\mathrm{tr}(\boldsymbol{Q}_1\boldsymbol{\Sigma}_{\hat\theta}(k+i))-\|\boldsymbol{e}_\theta(k+i)\|^2_{\boldsymbol{Q}_1}\}\\
&+\|\boldsymbol{e}_{\mu_{\hat\theta}}(k+H+1)\|^2_{\boldsymbol{Q}_3}+\mathrm{tr}(\boldsymbol{Q}_3\boldsymbol{\Sigma}_{\hat\theta}(k+H+1))-\|\boldsymbol{e}_\theta(k+H+1)\|^2_{\boldsymbol{Q}_3}\\
=\;&\sum_{i=0}^{H}\{-\|\tilde{\boldsymbol{\theta}}_\mu(k+i)\|^2_{\boldsymbol{Q}_1}+\mathrm{tr}(\boldsymbol{Q}_1\Sigma_{\hat\theta}(k+i))+2\tilde{\boldsymbol{\theta}}_\mu^T(k+i)\boldsymbol{Q}_1\boldsymbol{e}_{\boldsymbol{\mu}_{\hat\theta}}(k+i)\}\\
&-\|\tilde{\boldsymbol{\theta}}_\mu(k+H+1)\|^2_{\boldsymbol{Q}_3}+\mathrm{tr}(\boldsymbol{Q}_3\boldsymbol{\Sigma}_{\hat\theta}(k+H+1))\\
&+2\tilde{\boldsymbol{\theta}}_\mu^T(k+H+1)\boldsymbol{Q}_3\boldsymbol{e}_{\mu_{\hat\theta}}(k+H+1).
\end{aligned}\qquad\text{(C.20)}$$

The above last equality comes from the observation:

$$\begin{aligned} &\|\boldsymbol{e}_{\mu_{\hat{\theta}}}\|^2_{\boldsymbol{Q}_1} - \|\boldsymbol{e}_{\theta}\|^2_{\boldsymbol{Q}_1} = \|\boldsymbol{e}_{\mu_{\hat{\theta}}}\|^2_{\boldsymbol{Q}_1} - \|\boldsymbol{e}_{\mu_{\hat{\theta}}} - \tilde{\boldsymbol{\theta}}_{\mu}\|^2_{\boldsymbol{Q}_1} \\ = \ &\|\boldsymbol{e}_{\mu_{\hat{\theta}}}\|^2_{\boldsymbol{Q}_1} - \|\boldsymbol{e}_{\mu_{\hat{\theta}}}\|^2_{\boldsymbol{Q}_1} - \|\tilde{\boldsymbol{\theta}}_{\mu}\|^2_{\boldsymbol{Q}_1} + 2\tilde{\boldsymbol{\theta}}^T_{\mu}\boldsymbol{Q}_1\boldsymbol{e}_{\mu_{\hat{\theta}}} = -\|\tilde{\boldsymbol{\theta}}_{\mu}\|^2_{\boldsymbol{Q}_1} + 2\tilde{\boldsymbol{\theta}}^T_{\mu}\boldsymbol{Q}_1\boldsymbol{e}_{\mu_{\hat{\theta}}} \end{aligned}$$

with $\boldsymbol{e}_{\mu_{\hat{\theta}}} = \boldsymbol{\mu}_{\hat{\theta}} - \boldsymbol{\theta}_d$, $\tilde{\boldsymbol{\theta}}_{\mu} = \boldsymbol{\mu}_{\hat{\theta}} - \boldsymbol{\theta}$ and $\boldsymbol{e}_{\theta} = \boldsymbol{\theta} - \boldsymbol{\theta}_d$. The reason for using the above manipulation is to put a bound on $\bar{J}^k_{\hat{\theta}^0,\hat{W}^0_H} - J^k_{\theta,\hat{W}^0_H}$ by terms $\|\tilde{\boldsymbol{\theta}}_{\mu}\|$ and $\|\boldsymbol{e}_{\mu_{\hat{\theta}}}\|$, which are discussed in Lemmas 5.8 and 5.5, respectively.

Since $\lambda_{\max}(\boldsymbol{Q}_1) < \lambda_{\max}(\boldsymbol{Q}_3)$, from (C.20), we have

$$\begin{aligned} |\bar{J}^k_{\hat{\theta}^0,\hat{W}^0_H} - \bar{J}^k_{\theta,\hat{W}^0_H}| \leq \lambda_{\max}(\boldsymbol{Q}_3) \sum_{i=0}^{H+1} \{ &\|\tilde{\boldsymbol{\theta}}_{\mu}(k+i)\|^2 + \mathrm{tr}(\boldsymbol{\Sigma}_{\hat{\theta}}(k+i)) \\ &+ 2\|\boldsymbol{e}_{\mu_{\hat{\theta}}}(k+i)\|\|\tilde{\boldsymbol{\theta}}_{\mu}(k+i)\|\}. \end{aligned} \tag{C.21}$$

It is clear that the right-hand side of (C.21) is a function of $(H+2)$ probabilistic variables $\tilde{\boldsymbol{\theta}}_{\mu}$ and the joint distribution of these variables is given by Lemma 5.8, that is, events $\cap_{j=0}^{H+1}\boldsymbol{\Pi}^{\theta}_{\mu_{\hat{\theta}}}(k+j)$. By the definition of $\boldsymbol{\Pi}^{\theta}_{\mu_{\hat{\theta}}}(k+i)$, if $\boldsymbol{\Pi}^{\theta}_{\mu_{\hat{\theta}}}(k+i)$ is true, then $\|\tilde{\boldsymbol{\theta}}_{\mu}(k+i)\| \leq \varrho_{\hat{\theta}}(i)\|\boldsymbol{e}_{\alpha}(k)\| + a_2(i)$. From Lemma 5.5, $\|\boldsymbol{e}_{\mu_{\hat{\theta}}}(k+i)\| \leq a_4(i)\|\boldsymbol{e}_{\theta}(k)\| + a_5(i)$. From Lemma 5.3, $\|\boldsymbol{\Sigma}_{\hat{\theta}}(k+i)\| \leq i(\Delta t)^2\sigma^2_{\boldsymbol{f}\,\max}$. Noting that matrix $\boldsymbol{\Sigma}_{\hat{\theta}}(k+i) \in \mathbb{R}^{m\times m}$ is diagonal, we obtain $\mathrm{tr}(\boldsymbol{\Sigma}_{\hat{\theta}}(k+i)) \leq m\|\boldsymbol{\Sigma}_{\hat{\theta}}(k+i)\| \leq im(\Delta t)^2\sigma^2_{\boldsymbol{f}\,\max}$. Adding all of the above upper bounds for each term in the right-hand side of (C.21), we obtain that $|\bar{J}^k_{\hat{\theta}^0,\hat{W}^0} - \bar{J}^k_{\theta,\hat{W}^0}| \leq \rho_J(\boldsymbol{e}_{\alpha},\boldsymbol{e}_{\theta})$ given by (5.54). Therefore, events $\cap_{j=0}^{H+1}\boldsymbol{\Pi}^{\theta}_{\mu_{\hat{\theta}}}(k+j)$ implies event $\boldsymbol{\Pi}^J_{\bar{J}}$, and then we obtain

$$\Pr\{\boldsymbol{\Pi}^J_{\bar{J}}\} \geq \Pr\{\cap_{j=0}^{H+1}\boldsymbol{\Pi}^{\theta}_{\mu_{\hat{\theta}}}(k+j)\} \geq (1-\delta)^{m(H+1)+nH}.$$

This completes the proof.

C.9 Proof of Theorem 5.1

First, it is straightforward to obtain the lower-bound of $V(k)$ by the fact that $V(k) \geq \lambda_{\min}(\boldsymbol{Q}_1)\|\boldsymbol{e}_{\theta}(k)\|^2 + \zeta\lambda_{\min}(\boldsymbol{Q})\|\boldsymbol{e}_{\alpha}(k)\|^2 \geq \underline{\lambda}\|\boldsymbol{e}(k)\|^2$ and similarly for upper-bound

$V(k) \leq \overline{\lambda}\|\boldsymbol{e}(k)\|^2$. Therefore, we have

$$\underline{\lambda}\|\boldsymbol{e}(k)\|^2 \leq V(k) \leq \overline{\lambda}\|\boldsymbol{e}(k)\|^2. \tag{C.22}$$

From (5.51), we have

$$\begin{aligned} \Delta V(k) &\leq |\bar{J}^{k+1}_{\hat{\theta}^0,\hat{W}^0_H} - \bar{J}^{k+1}_{\theta,\hat{W}^0_H}| + |\bar{J}^{k}_{\hat{\theta}^0,\hat{W}^0_H} - \bar{J}^{k}_{\theta,\hat{W}^0_H}| + \zeta\left[V_\alpha(k+1) - V_\alpha(k)\right] \\ &+ \nu\big[\|\boldsymbol{\Sigma}_d(\hat{W}^0_H(k))\| - \|\boldsymbol{\Sigma}_d(\hat{W}^0_H(k+1))\|\big] + \Big(J^{k+1}_{\hat{\theta}^0,\hat{W}^0_H} - J^{k}_{\hat{\theta}^0,\hat{W}^0_H}\Big) \end{aligned} \tag{C.23}$$

Assuming that probabilistic events $\boldsymbol{\Pi}^J_{\bar{J}}(k+1)$ and $\boldsymbol{\Pi}^J_{\bar{J}}(k)$ are independent, we therefore apply the results in Lemma 5.9 to the first two terms in the above equation. For the third difference term, from Lemma 5.2, we take into the discrete-time case for (C.14) and obtain

$$\begin{aligned} V_\alpha(k+1) \quad - \quad V_\alpha(k) &\leq -\frac{1}{4}\Delta t \boldsymbol{e}^T_\alpha(k)\boldsymbol{Q}\boldsymbol{e}_\alpha(k) + c_3\Delta t \\ &\leq -\frac{1}{4}\Delta t \lambda_{\min}(\boldsymbol{Q})\|\boldsymbol{e}_\alpha(k)\|^2 + c_3\Delta t. \end{aligned}$$

For the last two difference terms in (C.23), by (C.17), we have

$$\begin{aligned} & J^{k+1}_{\hat{\theta}^0,\hat{W}^0_H} - J^{k}_{\hat{\theta}^0,\hat{W}^0_H} + \nu\left[\|\boldsymbol{\Sigma}_d(\hat{W}^0_H(k))\| - \|\boldsymbol{\Sigma}_d(\hat{W}^0_H(k+1))\|\right] \\ \leq\ & -\lambda_{\min}(\boldsymbol{Q}_1)\|\boldsymbol{e}_\theta(k)\|^2 + \Delta\hat{\boldsymbol{\alpha}}^0_{Q_2 k} + \nu\big[\|\boldsymbol{\Sigma}_d(\hat{W}^e_H(k+1))\| - \|\boldsymbol{\Sigma}_d(\hat{W}^0_H(k+1))\|\big] \\ & + \mathrm{tr}(\boldsymbol{Q}_1\boldsymbol{\Sigma}_{\hat{\theta}}(k+H+1)) + \mathrm{tr}(\boldsymbol{Q}_3\boldsymbol{\Sigma}_{\hat{\theta}}(k+H+2)) \\ \leq\ & -\lambda_{\min}(\boldsymbol{Q}_1)\|\boldsymbol{e}_\theta(k)\|^2 + \|\hat{\boldsymbol{\alpha}}^0(k+1)\|^2_{Q_2} + \nu\|\boldsymbol{\Sigma}_d(\hat{W}^e_H(k+1))\| + \\ & \mathrm{tr}(\boldsymbol{Q}_1\boldsymbol{\Sigma}_{\hat{\theta}}(k+H+1)) + \mathrm{tr}(\boldsymbol{Q}_3\boldsymbol{\Sigma}_{\hat{\theta}}(k+H+2)) \\ \leq\ & -\lambda_{\min}(\boldsymbol{Q}_1)\|\boldsymbol{e}_\theta(k)\|^2 + \hat{\alpha}^2_{\max} + \nu\sigma^2_{\boldsymbol{\kappa}\max} + m\lambda_m(H+2)(\Delta t)^2\sigma^2_{\boldsymbol{f}\max}. \end{aligned}$$

In the above last inequality, we use the facts that $l^0(k) \geq \lambda_{\min}(\boldsymbol{Q}_1)\|\boldsymbol{e}_\theta(k)\|^2$, $\|\hat{\boldsymbol{\alpha}}^0(k+1)\|^2_{\boldsymbol{Q}_2} \leq \hat{\alpha}^2_{\max}$, and $\|\boldsymbol{\Sigma}_d(\hat{W}^e_H(k+1))\| \leq \sigma^2_{\boldsymbol{\kappa}\max}$.

Substituting the above derivations into (C.23), we obtain

$$\begin{aligned}\Delta V(k) \leq\ & \xi_1\|\boldsymbol{e}_\alpha(k)\|^2 + \xi_2\|\boldsymbol{e}_\alpha(k)\|\|\boldsymbol{e}_\theta(k)\| + \xi_3\|\boldsymbol{e}_\alpha(k)\| + \xi_4\|\boldsymbol{e}_\theta(k)\| + \xi_5 \\ & -\frac{\zeta}{4}\Delta t\lambda_{\min}(\boldsymbol{Q})\|\boldsymbol{e}_\alpha(k)\|^2 + \zeta c_3\Delta t - \lambda_{\min}(\boldsymbol{Q}_1)\|\boldsymbol{e}_\theta(k)\|^2 + \hat{\alpha}^2_{\max} + \nu\sigma^2_{\boldsymbol{\kappa}\max} \\ & + m\lambda_m(H+2)(\Delta t)^2\sigma^2_{\boldsymbol{f}\max} \\ =\ & -\frac{1}{2}\Big(\gamma_3\|\boldsymbol{e}_\theta(k)\| - \frac{\xi_2}{\gamma_3}\|\boldsymbol{e}_\alpha(k)\|\Big)^2 - \frac{\gamma_3^2}{4}\|\boldsymbol{e}(k)\|^2 - (\gamma_1\|\boldsymbol{e}_\alpha(k)\| - \gamma_2)^2 \\ & -\Big(\frac{\gamma_3}{2}\|\boldsymbol{e}_\theta(k)\| - \gamma_4\Big)^2 + \gamma_5 \qquad \text{(C.24)}\end{aligned}$$

if (5.56) is held. Considering (C.24) and (C.22), we have

$$V(k+1) \leq (1 - \frac{\gamma_3^2}{4\bar{\lambda}})V(k) + \gamma_5 \qquad \text{(C.25)}$$

Defining event $\mathbf{\Pi}_e = \{V(k+1) \leq (1 - \frac{\gamma_3^2}{4\bar{\lambda}})V(k) + \gamma_5\}$, then from (C.23), it is straightforward to obtain $\mathbf{\Pi}^J_{\hat{J}}(k+1) \cap \mathbf{\Pi}^J_{\hat{J}}(k+1)$ induces $\mathbf{\Pi}_e$ and, therefore, for $0 < \delta < 1$, $\Pr\{\mathbf{\Pi}_e\} \geq \Pr\left\{\mathbf{\Pi}^J_{\bar{J}}(k+1) \cap \mathbf{\Pi}^J_{\bar{J}}(k)\right\} \geq (1-\delta)^{(m+n)H+2m+n}$.

References

[1] Y. Zhang, K. Chen, J. Yi, and L. Liu, "Pose estimation in physical human-machine interactions with application to bicycle riding," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2014, pp. 3333–3338.

[2] K. Chen, M. Trkov, J. Yi, T. Liu, Y. Zhang, and D. Song, "A robotic bipedal model for human walking with slips," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 6301–6306.

[3] I. Fantoni, R. Lozano, and M. W. Spong, "Energy based control of the pendubot," *IEEE Trans. Automat. Contr.*, vol. 45, no. 4, pp. 725–729, 2000.

[4] M. W. Spong, "Energy based control of a class of underactuated mechanical systems," in *IFAC World Congress*, vol. 29, no. 1, 1996, pp. 2828 – 2832.

[5] Y. Zhang, P. Wang, J. Yi, D. Song, and T. Liu, "Stationary balance control of a bikebot," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 6706–6711.

[6] R. Olfati-Saber, "Nonlinear control of underactuated mechanical systems with application to robotics and aerospace vehicles," Ph.D. dissertation, Dept. of Electrical Engineering and Computer Science, MIT, 2000.

[7] S. Feng, E. Whitman, X. Xinjilefu, and C. G. Atkeson, "Optimization-based full body control for the darpa robotics challenge," *J. Field Robotics*, vol. 32, no. 2, pp. 293–312, 2015.

[8] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot," *Auton. Robots*, vol. 40, no. 3, pp. 429–455, 2016.

[9] J. Englsberger, C. Ott, and A. Albu-Schäffer, "Three-dimensional bipedal walking control based on divergent component of motion," *IEEE Trans. Robotics*, vol. 31, no. 2, pp. 355–368, 2015.

[10] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 2, 2003, pp. 1620–1626.

[11] E. R. Westervelt, J. W. Grizzle, C. Chevallereau, J. H. Choi, and B. Morris, *Feedback Control of Dynamic Bipedal Robot Locomotion*. CRC Press, 2007.

[12] J. W. Grizzle, M. D. Di Benedetto, and F. Lamnabhi-Lagarrigue, “Necessary conditions for asymptotic tracking in nonlinear systems,” *IEEE Trans. Automat. Contr.*, vol. 39, no. 9, pp. 1782–1794, 1994.

[13] N. H. Getz, “Dynamic inversion of nonlinear maps with applications to nonlinear control and robotics,” Ph.D. dissertation, Dept. of Electrical Engineering and Computer Science, University of California, Berkeley, 1995.

[14] J. M. Wang, D. J. Fleet, and A. Hertzmann, “Gaussian process dynamical models for human motion,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 30, no. 2, pp. 283–298, 2008.

[15] I. Havoutis and S. Ramamoorthy, “Motion planning and reactive control on learnt skill manifolds,” *Int. J. Robot. Res.*, vol. 32, no. 9-10, pp. 1120–1150, 2013.

[16] N. D. Lawrence, “Gaussian process latent variable models for visualisation of high dimensional data.” in *Adv. Neural. Inform. Process. Syst. (NIPS)*, vol. 2, 2003, pp. 329–336.

[17] S. T. Roweis and L. K. Saul, “Nonlinear Dimensionality Reduction by Locally Linear Embedding,” *Science*, vol. 290, pp. 2323–2326, 2000.

[18] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popović, “Style-based inverse kinematics,” in *ACM Trans. on Graphics*, vol. 23, no. 3, 2004, pp. 522–531.

[19] R. Urtasun, D. J. Fleet, and P. Fua, “3d people tracking with gaussian process dynamical models,” in *Proc. IEEE Conf. Comp. Vis. Pattern Recog.*, vol. 1, 2006, pp. 238–245.

[20] R. Urtasun, D. J. Fleet, A. Geiger, J. Popović, T. J. Darrell, and N. D. Lawrence, “Topologically-constrained latent variable models,” in *Proc. Int. Conf. Machine Learning*. ACM, 2008, pp. 1080–1087.

[21] A. Elgammal and C.-S. Lee, “Tracking people on a torus,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 31, no. 3, pp. 520–538, 2009.

[22] B. Ferris, D. Fox, and N. Lawrence, “Wifi-slam using gaussian process latent variable models,” in *Proc. Int. Joint Conf. on Artifical Intelligence*, 2007, pp. 2480–2485.

[23] J. Ko and D. Fox, “Learning gp-bayesfilters via gaussian process latent variable models,” *Auton. Robots*, vol. 30, no. 1, pp. 3–23, 2011.

[24] D. J. Reinkensmeyer and J. L. Patton, “Can robots help the learning of skilled actions?” *Exercise and Sport Sciences Reviews*, vol. 37, pp. 43–51, 2009.

[25] S. Ikemoto, H. B. Amor, T. Minato, B. Jung, and H. Ishiguro, “Physical human-robot interaction: Mutual learning and adaptation,” *IEEE Robot. Automat. Mag.*, vol. 19, pp. 24–35, 2012.

[26] A. Safonova, J. K. Hodgins, and N. S. Pollard, “Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces,” *ACM Trans. on Graphics*, vol. 23, pp. 514–521, 2004.

[27] B. Rosenhahn, R. Klette, and D. Metaxas, *Human Motion Understanding, Modeling, Capture*. Springer, 2008.

[28] P. K. Artemiadis and K. J. Kyriakopoulos, “Emg-based control of a robot arm using low-dimensional embeddings,” *IEEE Trans. Robotics*, vol. 26, no. 2, pp. 393–398, 2010.

[29] I. Havoutis and S. Ramamoorthy, “Geodesic trajectory generation on learnt skill manifolds,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2010, pp. 2946–2952.

[30] R. Chalodhorn, D. B. Grimes, K. Grochow, and R. P. Rao, “Learning to walk by imitation in low-dimensional subspaces,” *Advanced Robotics*, vol. 24, no. 1-2, pp. 207–232, 2010.

[31] X. Zhao, Y. Fu, and Y. Liu, “Human motion tracking by temporal-spatial local gaussian process experts,” *IEEE Trans. Image Processing*, vol. 20, no. 4, pp. 1141–1151, 2011.

[32] L. A. Schwarz, D. Mateus, and N. Navab, “Recognizing multiple human activities and tracking full-body pose in unconstrained environments,” *Pattern Recognition*, vol. 45, no. 1, pp. 11–23, 2012.

[33] T. Feix, J. Romero, C. H. Ek, H.-B. Schmiedmayer, and D. Kragic, “A metric for comparing the anthropomorphic motion capability of artificial hands,” *IEEE Trans. Robotics*, vol. 29, no. 1, pp. 82–93, 2013.

[34] J. Romero, T. Feix, C. H. Ek, H. Kjellstrom, and D. Kragic, “Extracting postural synergies for robotic grasping,” *IEEE Trans. Robotics*, vol. 29, no. 6, pp. 1342–1352, 2013.

[35] J. Ko and D. Fox, “Gp-bayesfilters: Bayesian filtering using gaussian process prediction and observation models,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2008, pp. 3471–3476.

[36] Y. Zhang, K. Chen, and J. Yi, “Rider trunk and bicycle pose estimation with fusion of force/inertial sensors,” *IEEE Trans. Biomed. Eng.*, vol. 60, no. 9, pp. 2541–2551, 2013.

[37] K. Chen, Y. Zhang, and J. Yi, “Modeling of rider-bicycle interactions with learned dynamics on constrained embedding manifolds,” in *Proc. IEEE/ASME Int. Conf. Adv. Intelli. Mechatronics*, 2013, pp. 442–447.

[38] J. Yi, D. Song, A. Levandowski, and S. Jayasuriya, “Trajectory tracking and balance stabilization control of autonomous motorcycles,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2006, pp. 2583–2589.

[39] P. Wang and J. Yi, “Dynamic stability of a rider-bicycle system: Analysis and experiments,” in *Proc. Amer. Control Conf.*, 2015, pp. 1161–1166.

[40] J. A. Stevens, P. S. Corso, E. A. Finkelstein, and T. R. Miller, “The costs of fatal and non-fatal falls among older adults,” *Injury Prevention*, vol. 12, no. 5, pp. 290–295, 2006.

[41] F. Yang and Y.-C. Pai, “Reactive control and its operation limits in responding to a novel slip in gait,” *Annals of Biomedical Engineering*, vol. 38, no. 10, pp. 3246–3256, 2010.

[42] J. W. Grizzle, C. Chevallereau, R. W. Sinnet, and A. D. Ames, “Models, feedback control, and open problems of 3d bipedal robotic walking,” *Automatica*, vol. 50, no. 8, pp. 1955–1988, 2014.

[43] S. Srinivasan, I. Raptis, and E. R. Westervelt, “Low-dimensional sagittal plane model of normal human walking,” *Journal of Biomechanical Engineering*, vol. 130, no. 5, pp. 051 017–051 017–11, 2008.

[44] A. E. Martin, D. C. Post, and J. P. Schmiedeler, “Design and experimental implementation of a hybrid zero dynamics-based controller for planar bipeds with curved feet,” *Int. J. Robot. Res.*, vol. 33, no. 7, pp. 988–1005, 2014.

[45] K. Chen, M. Trkov, S. Chen, J. Yi, and T. Liu, “Balance recovery control of human walking with foot slip,” in *Proc. Amer. Control Conf.*, 2016, pp. 4385–4390.

[46] E. R. Westervelt, J. W. Grizzle, and D. E. Koditschek, “Hybrid zero dynamics of planar biped walkers,” *IEEE Trans. Automat. Contr.*, vol. 48, no. 1, pp. 42–56, 2003.

[47] C. Chevallereau, J. W. Grizzle, and C.-L. Shih, “Asymptotically stable walking of a five-link underactuated 3-d bipedal robot,” *IEEE Trans. Robotics*, vol. 25, no. 1, pp. 37–50, 2009.

[48] B. Morris and J. W. Grizzle, “Hybrid invariant manifolds in systems with impulse effects with application to periodic locomotion in bipedal robots,” *IEEE Trans. Automat. Contr.*, vol. 54, no. 8, pp. 1751–1764, 2009.

[49] K. Sreenath, H.-W. Park, I. Poulakakis, and J. W. Grizzle, “A compliant hybrid zero dynamics controller for stable, efficient and fast bipedal walking on mabel,” *Int. J. Robot. Res.*, vol. 30, no. 9, pp. 1170–1193, 2011.

[50] M. Trkov, J. Yi, T. Liu, and K. Li, “Shoe-floor interactions during human slip and fall: Modeling and experiments,” in *Proc. ASME Dyn. Syst. Control Conf.*, 2014, pp. V001T04A004–V001T04A004.

[51] L. B. Freidovich, U. Mettin, A. S. Shiriaev, and M. W. Spong, “A passive 2-dof walker: Hunting for gaits using virtual holonomic constraints,” *IEEE Trans. Robotics*, vol. 25, no. 5, pp. 1202–1208, 2009.

[52] S. Srinivasan, “Low-dimensional modeling and analysis of human gait with application to the gait of transtibial prosthesis users,” Ph.D. dissertation, Dept. of Mechanical Engineering, The Ohio State University, 2007.

[53] Y. Ye and C. K. Liu, “Optimal feedback control for character animation using an abstract model,” in *ACM Trans. Graphics*, vol. 29, no. 4, 2010, pp. 74:1–74:9.

[54] ——, “Synthesis of responsive motion using a dynamic model,” in *Computer Graphics Forum*, vol. 29, no. 2. Wiley Online Library, 2010, pp. 555–562.

[55] K. Chen, Y. Zhang, and J. Yi, “An integrated physical-learning model of physical human-robot interactions: A bikebot riding example,” in *Proc. ASME Dyn. Syst. Control Conf.*, 2014, pp. V003T42A002–V003T42A002.

[56] N. D. Lawrence and J. Quiñonero-Candela, “Local distance preservation in the gp-lvm through back constraints,” in *Proc. Int. Conf. Machine Learning*, 2006, pp. 513–520.

[57] P. Wang and J. Yi, “Balance equilibrium manifold and control of rider-bikebot systems,” in *Proc. Amer. Control Conf.*, 2016, pp. 2168–2174.

[58] C. E. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[59] G. Cao, E. M.-K. Lai, and F. Alam, “Gaussian process model predictive control of an unmanned quadrotor,” *J. Intelli. Robot. Syst.*, vol. 88, no. 1, pp. 147–162, 2017.

[60] J. Ko, D. J. Klein, D. Fox, and D. Haehnel, “Gaussian processes and reinforcement learning for identification and control of an autonomous blimp,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2007, pp. 742–747.

[61] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, “Gaussian processes for data-efficient learning in robotics and control,” *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 37, no. 2, pp. 408–423, 2015.

[62] J. Kocijan, R. Murray-Smith, C. E. Rasmussen, and A. Girard, “Gaussian process model based predictive control,” in *Proc. Amer. Control Conf.*, vol. 3, 2004, pp. 2214–2219.

[63] C. J. Ostafew, A. P. Schoellig, and T. D. Barfoot, “Robust constrained learning-based nmpc enabling reliable mobile robot path tracking,” *Int. J. Robot. Res.*, vol. 35, no. 13, pp. 1547–1563, 2016.

[64] R. Murray-Smith, D. Sbarbaro, C. E. Rasmussen, and A. Girard, "Adaptive, cautious, predictive control with gaussian process priors," *IFAC Proceedings Volumes*, vol. 36, no. 16, pp. 1155–1160, 2003.

[65] J. Rawlings and D. Mayne, *Model Predictive Control: Theory and Design*. Nob Hill Pub., 2009.

[66] J. Lee, R. Mukherjee, and H. K. Khalil, "Output feedback stabilization of inverted pendulum on a cart in the presence of uncertainties," *Automatica*, vol. 54, pp. 146 – 157, 2015.

[67] K. Chen, J. Yi, and T. Liu, "Learning-based modeling and control of underactuated balance robotic systems," in *Proc. IEEE Conf. Automat. Sci. Eng.*, 2017, pp. 1118–1123.

[68] D. Nguyen-Tuong, J. Peters, M. Seeger, and B. Schölkopf, "Learning inverse dynamics: a comparison," in *European Symposium on Artificial Neural Networks*, no. EPFL-CONF-175477, 2008.

[69] D. Nguyen-Tuong, M. Seeger, and J. Peters, "Model learning with local gaussian process regression," *Advanced Robotics*, vol. 23, no. 15, pp. 2015–2034, 2009.

[70] T. Beckers, J. Umlauft, D. Kulic, and S. Hirche, "Stable gaussian process based tracking control of lagrangian systems," in *Proc. IEEE Conf. Decision Control*, 2017, pp. 5180–5185.

[71] M. K. Helwa, A. Heins, and A. P. Schoellig, "Provably robust learning-based approach for high-accuracy tracking control of lagrangian systems," *arXiv preprint arXiv:1804.01031*, 2018.

[72] S. Zhou, M. K. Helwa, and A. P. Schoellig, "Design of deep neural networks as add-on blocks for improving impromptu trajectory tracking," in *Proc. IEEE Conf. Decision Control*, 2017, pp. 5201–5207.

[73] ——, "An inversion-based learning approach for improving impromptu trajectory tracking of robots with non-minimum phase dynamics," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1663–1670, 2018.

[74] P. Abbeel, A. Coates, and A. Y. Ng, "Autonomous helicopter aerobatics through apprenticeship learning," *Int. J. Robot. Res.*, vol. 29, no. 13, pp. 1608–1639, 2010.

[75] H. K. Khalil, *Noninear Systems*, 2nd ed. Prentice-Hall, 2002.

[76] H. Chen and F. Allgower, "A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability," *Automatica*, vol. 34, no. 10, pp. 1205 – 1217, 1998.

[77] P. Wang, J. Yi, T. Liu, and Y. Zhang, "Trajectory tracking and balance control of an autonomous bikebot," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 2414–2419.

[78] N. Srinivas, A. Krause, S. M. Kakade, and M. W. Seeger, “Information-theoretic regret bounds for gaussian process optimization in the bandit setting,” *IEEE Trans. Inform. Theory*, vol. 58, no. 5, pp. 3250–3265, 2012.

[79] I. Jolliffe, *Principal Component Analysis*. Wiley Online Library, 2002.

[80] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*, 3rd ed. New York: Wiley, 2006.