Understanding Feedforward – Feedback Controller Components In Human Movement Through Optimization-Based Approaches

by

Mahsa Parsapour

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2023

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Wael Suleiman Associate Professor Electrical and Computer Engineering University of Sherbrooke
Supervisors:	Dana Kulic Professor Electrical and Computer Engineering University of Waterloo — Monash University
	Katja Mombaur Professor Systems Design Engineering University of Waterloo
Internal Members:	Christopher Nielsen Professor Electrical and Computer Engineering University of Waterloo
	Yash Vardhan Pant Assistant Professor Electrical and Computer Engineering University of Waterloo
Internal-External Member:	John McPhee Professor Systems Design Engineering University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

This thesis contains some material that was taken from multi-author papers including:

Conference Paper

- M. Parsapour, and D. Kulić, 2021, "Recovery-matrix inverse optimal control for deterministic feedforward-feedback controllers", 2021 American control conference (ACC), pages 4765-4770, IEEE.

Journal Paper

- J. F. S. Lin, P. Carreno-Medrano, M. Parsapour, M. Sakr, and D. Kulić, 2021, "Objective Learning from Human Demonstration". Annual Reviews in Control, volume 51, pages 111-129. https://doi.org/10.1016/j.arcontrol.2021.04.003.

Abstract

Despite many studies in human motion analysis using optimal control theory to understand how movement is generated, less attention is focused on the structure of the optimal controller. The majority of existing studies assume that the person is using a feedforward controller to accomplish the desired task. However, during perturbed motions a feedback controller becomes active, and enables the person to react to unforeseen perturbations and adapt their motions to the environment. As such, understanding controller structure becomes important to analyze human motion more accurately. Three key contributions will be elaborated in this thesis to enable analyzing the human feedforward and feedback controller components.

The first key contribution is the formulation of an inverse optimization problem for trajectories generated by feedforward-feedback controllers for nonlinear systems and feedback controllers for linear systems. We adapt the recovery matrix inverse optimal control approach, originally developed for recovering the cost matrices from trajectories observed under feedforward control, and apply it to analyze trajectories observed from systems controlled in the feedback form plus additional feedforward term. This method also estimates the feedback gain for linear systems where inverse linear quadratic regulator approaches are dependent on the given feedback gain assumption. The perturbation in this study is added as a zero mean Gaussian noise at the state output.

The second key contribution is an algorithm to decompose the controller components for tracking problems. This algorithm uses Bellman optimality condition to form an optimization problem to detect whether the system was disturbed or not. We formulate a constrained optimization problem to estimate the control signal. The identification of controller components is made based on the estimated and the reference experimental trajectories. The proposed approach is tested in simulation, where a perturbation is applied on different nonlinear systems in a continuous form.

The third key contribution is to combine the first two contributions to analyze feedforward and feedback controller components of human movement. First, squat motion is identified as a suitable motion for analyzing human perturbed motion and controller structure. We collected both unperturbed and perturbed squat motions for this study. The perturbation is applied during a short time in a continuous form through a push stick. Then the human body is modeled as a three degrees of freedom system, and the task is modeled by an optimal control problem. By modifying the decomposition algorithm, the trials are classified and the controller components are identified by the inverse optimal control.

Acknowledgements

I would like to thank all the people who made this thesis possible.

My supervisor, Dr. Dana Kulić, for her support, guidance, and expertise. I am really thankful to have regular online meetings even though the time zone difference between Waterloo and Melbourne was always challenging. I am also grateful for her enthusiastic view on the topic.

My co-supervisor, Dr. Katja Mombaur, for giving me the opportunity to join her research group after my proposal comprehensive exam. I am really thankful to have her mentorship from another perspective on this topic. It was a great opportunity to connect with other graduate students and be active in this topic.

My doctorate examining committee, Dr. Wael Suleiman, Dr. John McPhee, Dr. Christopher Nielsen, and Dr. Yash Vardhan Pant for their time in reviewing my thesis and providing insight into the research field.

All members of the Human-Centred Robotics and Machine Intelligence (HCRMI) group, and Adaptive Systems Laboratory for the great moments I had with them during my studies at the university of Waterloo. Especially, I would like to thank Dr. Jonathan Feng-Shun Lin for his support. He would always answer my questions, and share his thoughts on the topic. Also, Robotics group at Monash university for the online group meetings and discussion during Covid. Before joining HCRMI group, online meetings with Monash University was the only platform I had to interact with other researchers, and I am happy to have that opportunity.

Women in engineering (WiE) at the University of Waterloo. I had the pleasure of organizing events with this group. It was really enjoyable to connect with other engineering students and participate in different activities.

UW Varsity Squash team that gave me the opportunity to play matches with other universities in Ontario. Playing squash gave me the momentum to my research. And, all other sport activities with warriors that created joyful moments during my PhD experience.

Finally, I would like to thank my family and friends for the support and persuasion.

Dedication

This thesis is dedicated to my mother, **Mina**, my father, **Kazem**, and my lovely sister, **Maryam**.

I was unlucky to lose my mother in the first year of my PhD studies. However, I can say that I was lucky to be her daughter, and learn from her how to be a good person. I was lucky to be beside her in the last few years of her life. Her bravery to fight cancer taught me that impossible is not impossible, and if I want something in life, there is always a way to reach that goal.

My father is always like a mentor to me in my personal and academic life. His optimism and encouragement always give me positive energy to keep going. Even though there were times that communication from miles away was not possible, but there was always a way that he would listen to me and that is a world to me.

Maryam is not only a sister to me, but also my best friend in the world. Her support during my challenging times is precious and unforgettable.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Humans have a remarkable ability to adapt their motions to the environment and learn how to react and predict unforeseen perturbations. The human central nervous system (CNS) controls body motions and can generate reactive motion in the face of a disturbance. In general, two categories of control models can be considered: open-loop control (feedforward) and closed-loop control (feedback) [178]. When we perform a task, the motor system is thought to deploy both feedforward and feedback control [178]. The first category of models focuses on open-loop control: If the CNS only relies on the learned information, and does not adjust the commands in real-time [121] then the motion is thought to be controlled by a feedforward controller. This controller ignores the role of online sensory feedback, and usually assumes deterministic dynamics. The second category of models focuses on closed-loop control: The CNS can adjust the movement by feedback control law to update the trajectory in the presence of noise, delays, internal fluctuations and unpredictable changes in the environment. The CNS has to constantly update the motor commands to correct for errors during conscious voluntary movements [178]. The sensory information can be received from vision, proprioception, audition, the vestibular system and internal models that can predict the motion [47].

To understand the control principle of natural movements, optimal control theory has been applied to human motion analysis [62]. In optimal control theory it is hypothesized that the CNS produces human motions that minimize certain task specific cost functions. Identifying the control strategy being used by the central nervous system for human movements has been widely studied for motions with feedforward controllers for unperturbed conditions [16, 54, 81, 133]. On the other hand, a few studies have focused on analyzing the role of feedback controllers by adding either mechanical or visual disturbances [121, 173, 181].

Understanding the control principles is not limited to finding the underlying cost functions, but also how the controllers are formed to perform a task. Therefore, in this thesis we address the following questions:

- How can we estimate the underlying cost functions in human motions?
- Is it possible to identify the controller structure which produces the human motion trajectory?
- If so, is it possible to separate out feedforward and feedback components?

The results can be used for applications such as physical therapy and sports medicine, as a rehabilitative modality, and as an assessment tool. It also provides an abstracted representation of the task [140], with the ability to: (1) model and generate new trajectories, (2) provide insight into why a given trajectory was selected, out of all possible trajectories, and (3) generalize motions to other tasks.

1.1 Problem Formulation

When analyzing human motions using objective learning techniques, it is assumed that the human is generating optimal trajectories according to the (unknown) objectives. As illustrated in Figure 1.1, we start with a dynamical system –which may represent the human in an environment – which evolves according to some function dependent on the current system state and the control actions. The system dynamics are assumed to be deterministic and are approximated by a function $x_{k+1} = f(x_k, u_k, d_k)$ that maps the current state of the system $x_k \in \mathcal{X} \subseteq \mathbb{R}^n$ and control action $u_k \in \mathcal{U} \subseteq \mathbb{R}^m$ at time k to a new state x_{k+1}, with some disturbance d_k. The agent can assess its current state x_k and the control action u_k via the objective function $J(x_k, u_k)$.

The goal of the *forward* problem is to find a control policy π that optimizes the expected cumulative return starting from an initial state x_0. The policy π is a function that maps a sequence of states and control actions $\zeta_{0:k} = \{(x_0, u_0), \ldots, (x_k, u_k)\}$ up to time k to a new control action. The cumulative return is given by:

$$V^{\pi}(x_0) = \sum_{k=0}^{T} J(x_k, u_k)$$

where T is the duration of the trajectory. $V^{\pi}(x) : \mathcal{X} \to \mathbb{R}$ is also known as the *state-value function* (or simply value function) under π.

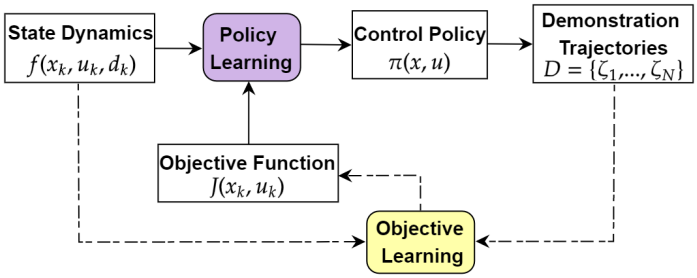


Figure 1.1: Graphic depiction of the policy learning (forward) (solid arrows) and objective learning (inverse) problems (dashed arrows).

Formally, when the objective function is known, it is hypothesized that the human motion solves the following optimization problem:

$$\begin{aligned} \min_{\pi} \quad & V^{\pi}(x_0) \\ \text{s.t.} \quad & x_{k+1} = f(x_k, u_k, d_k) \\ & u_k = \pi(\zeta_{0:k}) \\ & g(x_k) \leq 0, \end{aligned} \tag{1.1}$$

If the initial state is known, it is represented as a constraint on the initial state of the system. From Equation (1.1), we observe that policy learning algorithms aim to optimize the *objective function* $J(x_k, u_k)$ with respect to the control actions u_k, subject to the *state dynamics* $f(x_k, u_k, d_k)$ and any additional constraints $g(x_k)$, and where the *policy* $\pi(\zeta_{0:k})$ is the decision variable.

Given these key elements, we now introduce the objective learning problem, also known as the *inverse* problem (*i.e.*, inverse optimal control/inverse reinforcement learning). Given an observed set of motions $\mathcal{D} = \{\zeta^1, \ldots, \zeta^N\}$ sampled from the unknown control policy of the human expert π^E, objective learning algorithms seek to find an objective function $\hat{J}(x, u)$ such that any optimal trajectory $\hat{\zeta}$ generated with respect to this function would match in some way those provided by the expert. With $\hat{\pi}$ as the optimal control policy learned from the candidate objective function $\hat{J}(x, u)$, we formally introduce the objective learning problem as follows:

$$\hat{J}(x, u) = \text{argmax}_{J(x,u)} \sum_{\zeta \in \mathcal{D}} S(\hat{\zeta}, \zeta), \tag{1.2}$$

where $S(\hat{\zeta}, \zeta)$ corresponds to a similarity criterion used to guide the search for the expert's unknown objective function. We note that objective learning algorithms may also have knowledge of the dynamics of the system.

1.2 Thesis Contributions

This thesis develops optimization-based methods that identify the underlying objective of a given trajectory and analyze its structure. The three main contributions of this thesis are:

- **Controller identification through inverse optimal control**

  In the first study, we propose an algorithm based on recovery-matrix inverse optimal control [77] to infer the underlying controller. In this study, we adapt the recovery matrix inverse optimal control approach, originally developed for recovering the cost matrices from trajectories observed under feedforward control, and apply it to trajectories observed from systems controlled in the feedback form, with an additional feedforward term. This algorithm enables objective function recovery from trajectories generated by controllers with both feedback and feedforward components. Accurate cost function recovery is demonstrated via simulation examples with both linear and nonlinear systems.

  The proposed algorithm also estimates the controller gain for inverse linear quadratic regulator problems. In previous inverse optimal control approaches for linear systems [29, 123, 154, 196], it is assumed that the feedback gain is known. Without this assumption, the inverse linear quadratic regulator algorithms fail to provide the estimated weights for the underlying objective functions. Our algorithm removes the assumption of knowing the feedback gain.

- **Controller Decomposition Algorithm for Tracking Problems**

  In the second study, we present an approach to study the structure of controllers for nonlinear systems which may or may not be subject to a disturbance. We address whether the motion of nonlinear systems were generated by a feedforward controller or a combination of feedforward and feedback controller. Specifically, we are interested in understanding if the feedback is optimal or non-optimal. Our proposed method is based on the value function and Bellman's optimality condition. We formulate a constrained optimization problem to estimate the control signal, and the

decision on the structure is made based on comparing the estimated and the reference experimental trajectories. We assume four structures as a reference to decide about the controller. The performance of the algorithm is illustrated on an inverted pendulum and a two-link leg model in simulation.

- **Controller Analysis of Human Motion**

  In the third study, we adapt the approaches used in the first two studies to analyze the controller components on a real human dataset. We first identify and select the squat motion for the analysis. A simple experiment is designed and data is collected from one participant to study the differences between motions where the feedback is active or not. The activation of feedback is generated by perturbing the participant.

  We present an algorithm to classify perturbed and unperturbed trials, and then extract the information regarding the feedback and feedforward components through inverse optimal control results. We model the squat motion and identify potential objective functions to define the optimal control problem. This version of inverse optimal control approach is applied for the first time for understanding the differences between perturbed and unperturbed motions.

1.3 Thesis Outline

The rest of the thesis is organized as follows: Chapter 2 provides a summary of related works on optimization techniques for forward and inverse problems. It also overviews the methods used to analyze feedforward and feedback controllers in human motion analysis, and the type of perturbations that have been applied for studying human motion in practice.

Chapter 3 formulates the inverse optimization problem for trajectories generated by iterative linear quadratic regulators for nonlinear systems. The control policy in this type of formulation has both feedforward and feedback components and is locally-optimal for nonlinear systems. We show how to infer the underlying controller for this type of problem.

Chapter 4 proposes a decomposition algorithm using value function approximation. An optimization problem is formulated using Bellman optimality conditions to estimate the control signal. Assumptions are made on the structure of controllers in the feedforward and feedback forms in order to test the idea in simulation. The performance of the algorithm is shown on inverted pendulum and two-link leg model in simulation.

Chapter 5 modifies the algorithm proposed in chapter 4, and extends the idea to explain human movement analysis in practice. We explain the analysis of feedforward and feedback

controllers for squat motion. We present the squat motion dataset collected for analyzing the structure of controllers. This dataset includes both perturbed and unperturbed trials. Then, the task is formulated through an optimal control problem, and the inverse optimal control approach is explained for estimating the underlying objective functions. At the end, the controller of the squat motion is analyzed by the proposed algorithm and the feedforward and feedback components are identified.

Chapter 6 summarizes the results, and suggests possible extensions for future work.

Chapter 2

Related Works

In this Chapter we first overview the forward problem (Section 2.1) and inverse problem (Section 2.2 [1]) that was defined in Section 1.1. Then, we provide a summary of studies done on human perturbed motions in Section 2.3. Lastly, we talk about the core elements needed for this thesis from the related work in Section 2.4.

2.1 Forward problem

In forward problems, the objective functions are known, and the goal is to find a control policy by solving an optimal control problem. In general two types of controllers have been studied for human motion analysis: feedforward controllers and feedback controllers. Better feedforward control enables successful open-loop control during fast motions, whereas the feedback controller is necessary to make online error corrections in the presence of large internal fluctuations, noise, delays, and unpredictable changes in the environment. Humans have a remarkable ability to adjust movements to novel tasks or environments. When we perform a task, the motor system is thought to deploy both feedforward and feedback control [88]. Currently, little is known about how the learning of these two mechanisms relate to each other [88]. Performing some tasks cannot be done with only the feedback information from the sensory receptors [89]. A combination of both feedback and

[1] The text in this section is extracted from our survey paper "Objective Learning from Human Demonstrations" [113] that was a collaboration with Jonathan Feng-Shun Lin, Pamela Carreno-Medrano, Maram Sakr, and Dana Kulić.

feedforward processes is likely to be involved for most optimal movement control tasks. Especially in the context of adaptation to new tasks or new dynamical environments where the motion might have been perturbed, feedback is needed.

2.1.1 Feedforward Controllers

In feedforward models, the central nervous system (CNS) only recalls the learned information, and does not intelligently adjust the commands in real-time [121]. The majority of existing optimality models in motor control have been formulated in feedforward and assume deterministic dynamics. These models predict average movement trajectories or muscle activity by optimizing cost functions that correspond to what the sensorimotor system is trying to achieve [178]. For example, [59] studied the coordination of the human arm by minimizing hand jerk subject to boundary constraints such as hand position, velocity, and acceleration at the beginning and end of the movement. [148] studied maximum-height jumping by formulating an optimal control problem which maximizes the height reached by the center of mass of the body. The constraints include limitation on the magnitude of the incoming neural control signal, body-segmental, musculotendon, and activation dynamics, a zero vertical ground reaction force at lift-off. [185] formulates an optimal control problem to study human stand-to-sit movement. The optimization criterion minimizes three kind of energy costs, a center of gravity cost, and an input cost.

2.1.2 Feedback Controllers

To adjust the movement, the CNS needs sensory (feedback) information to update the trajectory and has to constantly update the motor commands to correct for errors during conscious voluntary movements [178] and reject unpredictable perturbations. The sensory information can be received from vision, proprioception, audition, the vestibular system and internal models that can predict the motion [47]. There exist some studies on understanding the role of feedback control on hand movement in the horizontal plane [48, 118, 192], and hand reaching motion in the vertical plane [142]. From these studies, it appears that both the feedforward and feedback controllers are responsible for generating motions in our body, but the relative contributions of the two components are not yet fully understood. An improved understanding of the relative importance of feedback and feedforward contributions could be helpful for understanding human movement. For example, if the difference in the motions of novices and experts can be quantified in terms of the difference in control methods, the stage of learning that a person has reached

can be identified, and a learner could be given proper guidance. By analyzing the motion control of a person with a movement disorder, the movement disorder would be understood correctly and a clue to treatment may be established [89].

Optimal feedback control theory can be used to work on feedback models. Optimal feedback control on linear models such as linear quadratic Gaussian (LQG) have been applied to linear models before [48,192]. These methods require a state-estimation process to estimate the current state of the body. The models have assumed that there is either no noise, or constant noise on the sensory inputs. [192] developed an optimal feedback control framework based on Bellman equation in which the cost is comprised of accuracy and effort. As the sensory input is state-dependent, the control policy is a combination of feedforward and feedback commands. This work developed sensorimotor strategies to reduce the overall cost. The optimal control model is able to reproduce the behavior of human arm reaching motion in the horizontal plane. [48] studied the problem of coordination in human arm reaching motion. The goal was to use optimal feedback control theory that predicts task demands changes in feedback control, and the correlation of different effectors that work together to achieve a goal.

For nonlinear systems, LQG methods cannot be used. [181] developed an iterative LQG for locally optimal feedback control of nonlinear stochastic systems subject to control constraints. This method constructs an affine feedback control law, obtained by minimizing a quadratic approximation to the optimal cost-to-go function. The performance of this method is shown on a human arm with 10 state dimensions and 6 muscle actuators in simulation. Later, the online version of iLQG was developed in [173] as online trajectory optimization to analyze complex humanoid robots to get up from an arbitrary pose on the ground and recovering from large disturbances using dexterous acrobatic maneuvers. Online trajectory optimization is known as model predictive control (MPC) where it repeatedly solves a finite-horizon optimal control problem. MPC enables to solve iLQG in real-time.

The iLQG solves the linearized model for the optimization at the end. Another method to use nonlinear models, is nonlinear model predictive controllers (NMPC) with a finite prediction horizon. NMPC was employed in [121] for planar human arm reaching motion. The NMPC uses an internal model to predict a future trajectory, and feedback information to correct the prediction errors. The methods used by [121] can correct the tracking errors for static targets or can follow a moving target seamlessly. The NMPC prediction horizon can represent the time horizon for which the CNS minimizes a physiological cost function.

2.2 Inverse Problem

In inverse problems, the objective functions are unknown. To infer the objective of a demonstration, the trajectory executed by the human is assumed to be optimal with respect to some unknown objective function. Objective learning methods have been developed from two research communities: the control community, where they are known as *inverse optimal control* (IOC) and the machine learning community, where they are known as *inverse reinforcement learning* (IRL) methods.

The majority of objective learning algorithms assume that the objective function $J(x, u)$ is given by a linear combination of basis features $\boldsymbol{\phi}(x, u) \in \mathbb{R}^k$ with weight parameters $\boldsymbol{\theta}$ such that

$$J^{\boldsymbol{\theta}}(x, u) = \boldsymbol{\theta}^T \boldsymbol{\phi}(x, u) \tag{2.1}$$

The objective learning problem then corresponds to estimating the weight vector $\boldsymbol{\theta}$. However, it is not immediately clear how to select these basis features. Most works manually identify features relevant to the specific task studied. For example, [151] minimized the combination of total force and moment of force for 4-finger pressing tasks, [152] considered torque minimization, pelvis position and velocity, joint angle regularization, foot motion periodicity and arm swing features for locomotion.

[16] summarized four types of features that are commonly included for human motion analysis: *kinematic features* such as velocity, acceleration and jerk, *dynamic features* such as torque and torque change, *geodesic features* such as path length, and *energy features* such as kinetic energy, work, positive work, and total absolute work.

If the system dynamics and optimal controller are assumed to be linear, the objective function is often assumed to penalize the states and control signal in quadratic form [52, 53, 112, 154, 182] to facilitate analytic solutions to the policy learning problem.

More recently, researchers have proposed approaches to relax this assumption on the structure of the objective function by using non-parametric models such as radial basis functions [112, 175], Gaussian processes [83, 107, 156] or neural networks [58, 63, 189]. Although Gaussian processes can capture complex relationships between features and scalar-value rewards as well as determine the saliency of each feature with respect to the relevance of the expert's demonstrations, they suffer from poor scaling with the number of samples. Neural networks also allow to model complex, nonlinear objective functions with the additional advantages of a favorable computational complexity and good scaling to problems with large, potentially high-dimensional state spaces [189]. However, they lack the structure

typically encoded in hand-engineered features and thus require additional regularization techniques [58] or the inclusion of a discriminator [63] in order to robustly scale to complex tasks. While these approaches can represent a richer and nonlinear objective function structure, the interpretability of the objectives may be lost.

The above algorithms assume that a single reward function guides the demonstrated behavior. However, for longer demonstrations, multiple sub-goals may partition the demonstration. A number of works have examined how to automatically detect and identify these sub-goals. The assumption that the whole trajectory shares a common objective function can be relaxed by applying objective learning on a fixed-length sliding window [114] or with a dynamically sized window based on the recoverability of the windowed data [81].

If demonstration trajectories were generated by different experts, some experts might follow different objective criteria, resulting in multi-intent problems. [13] consider the case when there are multiple demonstration trajectories, corresponding to multiple intents. The objective is to simultaneously cluster the trajectories by intent, and estimate the intents. Their approach requires the number of clusters to be specified *a priori* .

As shown in Figure 2.1, at a high-level, estimating an objective function from demonstrated expert behavior can be framed as an iterative process. Starting from an initial guess of the objective function, the estimate is improved in a two-step process: (1) a comparison step in which the similarity between the behavior induced by the current estimate of the objective function and the observed demonstrations is measured; and (2) an update step in which the current estimate is modified so as to increase the similarity between the induced and observed behaviors. Thus to solve the inverse problem, an accurate measure of similarity between the induced and observed behavior is critical. Note that the framing in Figure 2.1 also implies that a solver for the forward problem for each new objective function candidate is required.

2.2.1 Inverse Optimal Control

IOC was first proposed by Kalman [87], and then applied to tasks such as human locomotion [5,133,155], human arm movement [8,16,17,182], and squat motions [81,114]. We describe the existing IOC methods as follows:

Bi-level IOC

In the bi-level IOC approach [133], as illustrated in Figure 2.1, the update step is implemented via an "upper-level" optimization, minimizing the error between the demonstration

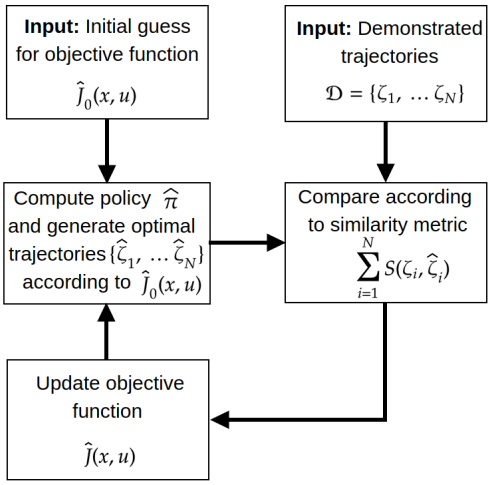


Figure 2.1: High level overview of typical objective learning algorithms.

trajectory and a simulated trajectory. A nested "lower-level" optimization solves the forward problem to generate the simulated trajectory. Due to the nested forward problem, the upper-level problem is both nonlinear and lacks an analytical gradient, thus requiring a derivative-free optimization method.

Formally, the upper-level is formulated as the minimization of the error between $\hat{\zeta}(x,u,\boldsymbol{\theta})$, the trajectory resulting from the minimization of the objective expected cumulative cost $V(x,u,\theta)$, and the demonstration trajectory ζ, where $\boldsymbol{\theta}$ denotes the parameters of the objective function, i.e. the basis function $\boldsymbol{\phi}(x,u)$ weights

$$\min_{\boldsymbol{\theta}} \|\hat{\zeta}(x,u,\theta) - \zeta\|^2 \tag{2.2}$$

while the lower-level solves the forward problem to generate $\hat{\zeta}(x,u,\boldsymbol{\theta})$

$$\min_{\hat{\zeta}} V(x,u,\boldsymbol{\theta}) := \sum_{t=0}^{T} \boldsymbol{\theta}^T \boldsymbol{\phi}(x_t,u_t) \tag{2.3}$$

$$\text{s.t.} \quad x_{t+1} = f(x_t,u_t), \quad t = 0,\ldots,T \tag{2.4}$$

where $f_j(x,u)$ is the deterministic and continuous time version of the dynamic function first introduced in Equation (1.1). Equation (2.2) can then be solved to obtain the objective function [133].

Alternative formulations include minimizing trajectory error while calculating the weights using Pontryagin's maximum principal [78] or linear quadratic regulator (LQR) [52,53,182] as an optimization framework.

One-level IOC

Instead of the bi-level approach, [66] proposed replacing the lower-level direct problem with optimality conditions, to combine the two stages into a single step. This formulation consists of utilizing Lagrangian multipliers [174] or Karush-Kuhn-Tucker (KKT) conditions [8,66] to solve the lower-level forward problem, allowing the two levels to be combined into a single level optimization problem. Given the Lagrangian $\mathcal{L} := V + \boldsymbol{\lambda}_{\text{eq}}^T \boldsymbol{r}_{\text{eq}} + \boldsymbol{\lambda}_{\text{ineq}}^T \boldsymbol{r}_{\text{ineq}}$ where $\boldsymbol{\lambda}_{\text{eq}}$ and $\boldsymbol{\lambda}_{\text{ineq}}$ denote the Lagrangian multipliers associated to the equality $\boldsymbol{r}_{\text{eq}}$ and inequality $\boldsymbol{r}_{\text{ineq}}$ constraints respectively, the optimization problem is formulated as follows:

$$
\begin{aligned}
\min_{(x,u,\theta,\lambda_{\text{eq}},\lambda_{\text{ineq}})} \quad & \sum_{t=0}^{T-1} (\hat{\zeta}(x_t, u_t, \boldsymbol{\theta}) - \zeta_t)^2 \\
\text{s.t.} \quad & x_{t+1} = f(x_t, u_t), \\
& 0 = \boldsymbol{r}_{\text{eq}}(x_t, u_t), \\
& 0 = \boldsymbol{r}_{\text{ineq}}(x_t, u_t), \\
& 0 = \nabla_{(x,u,q)} \mathcal{L}(x_t, u_t, \theta, \boldsymbol{\lambda}_{\text{eq}}, \boldsymbol{\lambda}_{\text{ineq}}), \\
& 0 \leq \boldsymbol{\lambda}_{\text{ineq}}, \\
& 0 = \boldsymbol{\lambda}_{\text{ineq}}^T \boldsymbol{r}_{\text{ineq}}(x_t, u_t), \quad t = 0, \ldots, T
\end{aligned}
\tag{2.5}
$$

where ζ_t indicates the observed state-control pair at time t in the demonstration trajectory ζ. The one-level method's main advantage is that the problem can be formulated into a single stage, reducing the problem complexity during implementation.

Optimality-based IOC

The most indirect set of IOC methodologies do not assess similarity directly, but instead minimize the residual or optimality violations. Since the input trajectory is assumed to be optimally generated from a dynamic system and a set of cost functions, the generated trajectory should be optimal with respect to the cost function, and therefore satisfy optimality criteria. However, practical factors like noise in the trajectory or uncertainty in the

cost function formulation lead to deviations from the optimal trajectory. In this class of approaches, these optimality violations are minimized to recover the objective function.

These methods are computationally fast because they do not require the trajectory or features to be computed, i.e., they avoid the need for solving the forward problem. However, the majority of these methods require computing the cost function gradient along the trajectory, thus requiring higher order derivatives of the system dynamics and features to be available. The minimization of the gradient may not also lead to a minimization of the cost function value [19].

Karush-Kuhn-Tucker Conditions: A common approach is to rely on the KKT conditions [28], which specify that the gradient of the objective function should be zero along the optimal trajectory. To calculate the cost weights, the KKT equations can be re-formulated into a linear residual matrix and minimized. Given an objective function modeled as a weighted sum of basis cost functions $\boldsymbol{\phi}(\mathbf{x})$ to be minimized with respect to some given equality $\boldsymbol{r}_{\text{eq}}$ (inequality terms excluded for brevity):

$$\begin{aligned} \min_{\mathbf{x}^*} \; & V(\mathbf{x}^*) = \boldsymbol{\theta}^T \boldsymbol{\phi}(\mathbf{x}^*) \\ \text{s.t.} \quad & \boldsymbol{r}_{\text{eq}}(\mathbf{x}^*) = 0 \end{aligned} \tag{2.6}$$

the KKT Lagrangian $L(\mathbf{x})$ and gradient $\nabla_{\mathbf{x}} L(\mathbf{x})$ are defined as:

$$L(\mathbf{x}^*) = \boldsymbol{\theta}^T \boldsymbol{\phi}(\mathbf{x}^*) + \boldsymbol{\lambda}_{\text{eq}} \boldsymbol{r}_{\text{eq}}(\mathbf{x}^*) \tag{2.7}$$

$$\nabla_{\mathbf{x}} L(\mathbf{x}^*) = \boldsymbol{\theta}^T \nabla_{\mathbf{x}} \boldsymbol{\phi}(\mathbf{x}^*) + \boldsymbol{\lambda}_{\text{eq}} \nabla_{\mathbf{x}} \boldsymbol{r}_{\text{eq}}(\mathbf{x}^*) \tag{2.8}$$

where the partial differential of the gradient $\nabla_{\mathbf{x}}$ is calculated with respect to the state $\mathbf{x}$, λ_{eq} are the Lagrangian multipliers on $r_{\text{eq}}(\mathbf{x}^*)$. The condition that must be met to ensure optimality is:

$$\nabla_{\mathbf{x}} L(\mathbf{x}^*) = 0 \tag{2.9}$$

If it is assumed that the system is not strictly optimal, but rather only approximately optimal [91], then Equation (2.10) is minimized but is not strictly zero:

$$\begin{aligned} \min_{\hat{\boldsymbol{\theta}}, \lambda} \; & \nabla_{\mathbf{x}} L(\mathbf{x}) \\ \text{s.t.} \quad & \hat{\boldsymbol{\theta}} \geq 0 \end{aligned} \tag{2.10}$$

Since the KKT equations are linear with respect to the unknown variables $\hat{\boldsymbol{\theta}}$ and $\boldsymbol{\lambda}_{\text{eq}}$, Equation (2.10) can be written as a least square problem and solved computationally efficiently [155].

While [155] hand-selected a significant basis cost function to prevent the zero weight ($\boldsymbol{\theta} = 0$) trivial solution, [144] employed a basis function pivot to estimate the significant basis function, while [54] add a regularizing constraint to force $||\hat{\boldsymbol{\theta}}||_1 = 1$. Key improvements to the inverse KKT method include checks to ensure that the residual matrix is well formed [145], and factoring out $\hat{\boldsymbol{\lambda}}$ terms using the Hessian [54].

Other Optimality Conditions: Other optimality conditions that have been used in the literature include the Euler-Lagrange equation [5, 175], the Pontryagin's maximum principle [82, 130, 194], as well as the Hamilton-Jacobi-Bellman [112, 122, 136].

Controller-based IOC

A small number of papers approach the IOC problem by minimizing the error with respect to the gains. [154] and [123] employ a LQR framework. The optimal gain K_e is either known, or estimated using least squares from the observation data $y = A - B * K$, assuming known system dynamics matrix A and B. They then estimate the state Q and controller R matrix via gradient descent to minimize the Frobenius norm of $K - K_e$.

2.2.2 Inverse Reinforcement Learning

Given the parametrization of the objective function as a weighted sum of features (Equation (2.1)), early works in the IRL literature proposed to compare demonstrations and generated trajectories based on the *feature expectations*. With this approach, the expected cumulative discounted feature counts, or more succinctly the feature expectations for a policy π, are defined as

$$\begin{aligned} \boldsymbol{\mu}(\pi) &= \mathbb{E}_{\zeta\sim\pi}\Big[\boldsymbol{\mu}(\zeta)\Big] \\ &= \mathbb{E}_{\zeta\sim\pi}\Big[\sum_{t=1}^{T}\gamma^{t-1}\boldsymbol{\phi}(x_t, u_t)\Big], \end{aligned} \tag{2.11}$$

where T indicates the duration of the demonstration trajectory and $\boldsymbol{\mu}(\zeta) = \sum_{t=1}^{T}\gamma^{t-1}\boldsymbol{\phi}(x_t, u_t)$ corresponds to the feature expectations along any trajectory sampled from a policy π. Us-

ing this notation, the value function of a policy π can be rewritten as $V^{\pi}(x) = \boldsymbol{\theta}^T \boldsymbol{\mu}(\pi) \, \forall \, x \in \mathcal{X}$.

Given a set of expert demonstrations $\mathcal{D} = \{\zeta_1, \ldots, \zeta_N\}$, we denote the empirical estimate of the expert's feature expectations by

$$\bar{\boldsymbol{\mu}}(\pi^E) = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{\mu}(\zeta_i) \tag{2.12}$$

Apprenticeship Learning Algorithms

First introduced by [3], the apprenticeship learning (AL) algorithms aim at finding a policy $\tilde{\pi}$ whose performance (or expected value) is close to the expert's policy π^E maximizing the unknown reward function $R^{\boldsymbol{\theta}}(s,a) = \boldsymbol{\theta}^T \boldsymbol{f}(s,a)$. It is important to notice that for these linear reward function approximations, feature expectations completely determine the expected cumulative reward for any policy [3]. Hence, a match in feature expectations under 2 policies implies a match in the expected value of both policies. Formally, this relation is defined $\forall \, x \in \mathcal{X}$ as

$$\begin{aligned} |V^{\hat{\pi}}(x) - V^{\pi^E}(x)| &= |\boldsymbol{\theta}^T \boldsymbol{\mu}(\hat{\pi}) - \boldsymbol{\theta}^T \boldsymbol{\mu}(\pi^E)| \\ &\leq ||\boldsymbol{\theta}||_2 ||\boldsymbol{\mu}(\hat{\pi}) - \boldsymbol{\mu}(\pi^E)||_2 \\ &\leq 1 \cdot \epsilon = \epsilon \end{aligned} \tag{2.13}$$

where the first and second inequalities follow from the Cauchy-Schwarz inequality and $||\boldsymbol{\theta}^*||_2 \leq ||\boldsymbol{\theta}^*||_1 \leq 1$.

From Equation (2.13) it follows that a policy $\hat{\pi}$ that induces feature expectations $\boldsymbol{\mu}(\hat{\pi})$ close to $\boldsymbol{\mu}(\pi^E)$, i.e., $||\boldsymbol{\mu}(\tilde{\pi}) - \boldsymbol{\mu}(\pi^E)|| < \epsilon$ will result in a performance close to the expert's. To find this policy $\hat{\pi}$, AL algorithms solve the following optimization problem

$$\begin{aligned} \max_{p, \boldsymbol{\theta}} \quad & p \\ \text{s.t.} \quad & \boldsymbol{\theta}^T \boldsymbol{\mu}(\pi^E) \geq \boldsymbol{\theta}^T \boldsymbol{\mu}(\tilde{\pi}^{(j)}) + p, \; j = 0, \cdots, i-1 \\ & ||\boldsymbol{\theta}||_2 \leq 1. \end{aligned} \tag{2.14}$$

The solution to Equation (2.14) generates an objective function $J^{\boldsymbol{\theta}}(s,a) = \boldsymbol{\theta}^{(i)} \cdot \boldsymbol{\phi}(s,a)$ such that the expert's policy (as illustrated by the observed trajectories) does better, by a p margin, than any of the policies found so far. The solutions $\Pi = \{\hat{\pi}^{(0)}, \cdots, \hat{\pi}^{(j)}\}$

and $\Theta = \{\boldsymbol{\theta}^{(0)}, \cdots, \boldsymbol{\theta}^{(j)}\}$ are found through a three-step iterative process, as illustrated in Figure 2.1: (1) find an optimal policy $\tilde{\pi}^{(i)}$ under the current objective function estimate $\boldsymbol{\theta}^{(i)}$, (2) compute feature expectations $\boldsymbol{\mu}(\hat{\pi}^{(i)})$ of the optimal policy obtained in the previous step, and (3) update the objective function estimate so as to reduce the difference in feature expectations between the optimal and expert's policies until convergence. Notice that steps (1) and (2) require access to a RL method that computes an optimal policy from a given objective function.

Although AL algorithms estimate an objective function as part of the optimization process, they do not necessarily recover the expert's underlying objective function correctly. These algorithms are only guaranteed to find an objective function that matches feature expectations and results in a policy whose performance is bounded by the expert's observed performance [25].

Maximum Margin Planning

Although apprenticeship learning algorithms aim at finding an objective function that maximizes the similarity between the feature expectations underlying the optimal and expert's policies, this optimization criterion alone fails to provide a mechanism for explicitly matching the expert's behavior [167]. To address this issue, the Maximum Margin Planning (MMP) algorithm proposed in [159] learns an objective function for which a single deterministic and stationary policy with a guaranteed upper-bound (or margin) on the dissimilarity between the expert's and policy demonstrations can be obtained.

Starting from the same assumption of a linear objective function, MMP augments the optimization criterion based on the similarity between the expected value of the learned and expert policy with a loss function $l(\zeta, \hat{\zeta})$ that penalizes all state-action pairs for which the optimal path $\hat{\zeta}$ sampled from the learned policy $\hat{\pi}$ fails to match the observed expert's trajectory $\zeta \in \mathcal{D}$. Formally, the MMP algorithm aims to solve the following optimization problem

$$\begin{aligned} \min_{\boldsymbol{\theta}, \beta_i} \quad & \lambda ||\boldsymbol{\theta}||_2 + \sum_{i=1}^{N} \beta_i \\ \text{s.t.} \quad & \boldsymbol{\theta}^T \boldsymbol{\mu}(\zeta_i) \boldsymbol{\psi}(\zeta_i) + \beta_i \leq \max_{\hat{\zeta} \sim \hat{\pi}} \boldsymbol{\theta}^T \boldsymbol{\mu}(\hat{\zeta}) \boldsymbol{\psi}(\hat{\zeta}) + l(\zeta_i, \hat{\zeta}), \ \forall\, i = 1, \ldots, N \end{aligned} \tag{2.15}$$

where N is the total number of demonstrated trajectories, $\boldsymbol{\psi}(\zeta)$ are the state-action visitation counts along a trajectory ζ, β_i is a slack variable that accounts for the error in the

margin constraint for the $i-$th trajectory, and λ balances the trade-off between regularization and meeting the constraints. The loss function $l(\zeta, \hat{\zeta})$ is proportional to the empirical visitation frequencies of each state-action pair so as to make highly visited state-action pairs take on larger reward values. By doing so, a preference over learned policies that frequently visit these states and thus closely mimic the observed behavior is induced [159].

Maximum Entropy IRL

Based on the observation that a policy π can be also interpreted as a distribution over the entire class of possible trajectories or paths, [199] proposed to leverage the principle of maximum entropy (MaxEnt) so as to deal with the degeneracy of the IRL problem. Given that typically many different distributions of paths (i.e., policies) can match the empirical feature expectations obtained from the expert's observed trajectories $\mathcal{D} = \{\zeta_1, \ldots, \zeta_N\}$, the principle of maximum entropy resolves this ambiguity by choosing "the least committed" distribution, that is, the distribution (or policy) that does not exhibit any additional preferences beyond matching the expert's feature expectations. In addition to dealing with the inherent degeneracy of the IRL problem, the MaxEnt formulation also offers a principled way of accounting for potentially imperfect or sub-optimal behavior in the expert's demonstrations. Formally, the maximum entropy IRL algorithm aims at matching

$$\sum_{\zeta \sim \hat{\pi}} p(\zeta) \boldsymbol{\mu}(\zeta) = \bar{\boldsymbol{\mu}}(\pi^E), \tag{2.16}$$

where the empirical feature expectations $\bar{\boldsymbol{\mu}}(\pi^E)$ are computed according to Equation (2.12) and $p(\zeta)$ corresponds to the distribution over paths induced by the optimal policy $\hat{\pi}$ learned from the parameterized objective function $J^{\boldsymbol{\theta}}(x, u)$. Thus, $p(\zeta|\boldsymbol{\theta})$, the probability of observing a trajectory ζ given the weights $\boldsymbol{\theta}$, is defined as

$$p(\zeta|\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} q(\zeta) \exp\left(\boldsymbol{\theta}^T \boldsymbol{\mu}(\zeta)\right), \tag{2.17}$$

where $q(\zeta)$ is the (un-normalized) probability of any trajectory ζ to occur according to the system dynamics $\mathcal{T}$

$$q(\zeta) = p_0(x_1) \prod_{t=1}^{T} p(x_{t+1}|x_t, u_t), \tag{2.18}$$

and $Z(\boldsymbol{\theta}) = \sum_{\zeta' \sim \pi} q(\zeta') \exp\left(\boldsymbol{\theta}^T \boldsymbol{\mu}(\zeta')\right)$ is the normalization constant often referred to as the partition function. We note that according to Equation (2.17) equally rewarded trajectories

have equal probabilities, trajectories with higher rewards have the highest likelihood and the expert can still generate sub-optimal trajectories with a probability that decreases exponentially as the trajectories become less rewarded. In the case of deterministic dynamics, Equation (2.17) reduces to $p(\zeta|\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \exp\left(\boldsymbol{\theta}^T \boldsymbol{\mu}(\zeta)\right)$.

Learning "the least committed" distribution over paths can be formally defined as finding the objective function parameters $\boldsymbol{\theta}$ that maximize the casual entropy $\mathcal{H}(\cdot)$ of π, subject to the constraint of matching the observed feature expectations

$$\begin{aligned} \max_{\boldsymbol{\theta}} \quad & \mathcal{H}(\pi) \\ \text{s.t.} \quad & \boldsymbol{\theta}^T \boldsymbol{\mu}(\pi) = \boldsymbol{\theta}^T \bar{\boldsymbol{\mu}}(\pi^E), \\ & \sum_{\zeta \sim \pi} p(\zeta|\boldsymbol{\theta}) = 1, \\ & p(\zeta|\boldsymbol{\theta}) \geq 0 \; \forall \, \zeta \sim \pi. \end{aligned} \tag{2.19}$$

One of the main limitations of the MaxEnt IRL formulation initially proposed by [199] is the need for an exact computation of the partition function $Z(\boldsymbol{\theta})$. Although this can be easily done in discrete environments for which a full knowledge of the dynamics of the system is available, it becomes computationally unfeasible for large, continuous spaces for which the dynamics of the system are likely unknown. Several extensions to the MaxEnt formulation have been proposed to address this issue, they can be divided into two main groups: discretization and continuous approximations.

Bayesian IRL

Bayesian approaches estimate a probability distribution over the objective function given the observed demonstrations. [157] proposed the first Bayesian formulation for IRL.

Similar to the MaxEnt approach, they model the probability of observing a trajectory ζ given a particular objective function $J^{\boldsymbol{\theta}}$ via an exponential distribution:

$$p(\zeta|J^{\boldsymbol{\theta}}) = \frac{1}{Z} \exp \alpha \mathbb{E}(\zeta, J^{\boldsymbol{\theta}}) \tag{2.20}$$

where α is a parameter representing the confidence in the demonstrator's expertise; $\mathbb{E}(\zeta, J^{\boldsymbol{\theta}}) = \sum_i V(x_i, \boldsymbol{\theta})$, with $\hat{\pi}$ denotes the optimal policy with respect to the objective function $J^{\boldsymbol{\theta}}$ and Z is a normalizing constant. Given this model, the posterior probability of $J^{\boldsymbol{\theta}}$ can be

computed by applying Bayes theorem

$$p(J^{\boldsymbol{\theta}}|\zeta) = \frac{p(\zeta|J^{\boldsymbol{\theta}})p(J^{\boldsymbol{\theta}})}{p(\zeta)} = \frac{1}{Z'} \exp \alpha \mathbb{E}(\zeta, J^{\boldsymbol{\theta}}) p(J^{\boldsymbol{\theta}})$$

such that $p(J^{\boldsymbol{\theta}})$ captures any prior knowledge about the reward.

Given the posteriori distribution $p(J^{\boldsymbol{\theta}}|\zeta)$, different point estimates can be interpreted to optimize different similarity criteria. The posteriori mean minimizes the least squared loss function between the actual and estimated objective function, while the median minimizes the linear loss function. The optimal policy corresponding to the mean objective function is also shown to minimize the policy loss function (i.e., the difference between the optimal and estimated value functions). Therefore, depending on which posteriori point estimate is used, the Bayesian IRL (BIRL) framework either uses objective function or value function comparison for the similarity estimate.

Maximum Likelihood

In this class of approaches, given a model of the likelihood of observing a given trajectory given an objective function, the likelihood is directly optimized. For example, [85] use the PI^2 algorithm [176] as the basis for IRL. They assume that the state-dependent cost function is linearly parameterized (as in Equation 2.1), and the weight vector to be learned is the concatenation of the state-dependent basis function weights, the control cost scaling (assuming the shape of the quadratic control cost is known) and the terminal cost scaling. Given an exponential probability of observing a given trajectory conditioned on the reward (as in Equation (2.20)), the weights are found by minimizing the negative log of the probability of observing the demonstrated trajectories, with an added L-1 norm regularization term over the weights, using a quasi-Newton optimization approach. The proposed approach is demonstrated for learning inverse kinematics and optimal motion trajectories for reaching with a 7 DoF arm. [119] apply the path integral IRL algorithm [85] to recover the objective function of segmented human-human collaborative motions.

Controller-based IRL

Under the assumption that reward functions are paramterizations of a policy class, [139] propose a novel gradient algorithm that learns a reward function such that the resulting optimal policy matches closely an expert's observed trajectories. To do so, the authors

combine ideas from supervised learning and apprenticeship learning [3]. Specifically, the proposed algorithm seeks to minimize an optimization criterion that penalizes deviations from the expert's policy (*i.e.*, supervised learning). The policy against which the expert's policy is compared is obtained by tuning a reward function and learning the optimal policy with respect to this function (*i.e.*, AL). Formally, the proposed gradient algorithm aims to solve the following optimization problem

$$\begin{aligned} \min_{\boldsymbol{\theta}} \quad & \sum_{x \in \mathcal{X}, u \in \mathcal{U}} \bar{\psi}^E(x)(\hat{\pi}(x,u) - \bar{\pi}^E(x,u))^2 \\ \text{s.t.} \quad & \hat{\pi}(x,u) = G(Q(x,u,\boldsymbol{\theta})) \; \forall \, (x,u) \in \mathcal{X} \times \mathcal{U}, \end{aligned} \tag{2.21}$$

where $\bar{\psi}^E(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{T} \sum_{x_t \in \zeta_i} \mathbf{1}_{x_t = x}$ is the empirical occupation frequency of state x under the expert's policy, $\bar{\pi}^E(u|x) = \frac{\sum_{i=1}^{N} \sum_{(x_t,u_t) \in \zeta_i)} \mathbf{1}_{x_t = x, u_t = u}}{\sum_{i=1}^{N} \sum_{x_t \in \zeta_i} \mathbf{1}_{x_t = x}}$ corresponds to the empirical estimate of expert's policy, $Q(x,u,\boldsymbol{\theta})$ is the optimal action-value function of the optimal policy $\hat{\pi}$ learned from the parameterized objective function $J^{\boldsymbol{\theta}}(x,u)$, and G is a suitable smooth mapping that returns a greedy policy with respect to its argument. Since the mapping from the space of reward parameters $\boldsymbol{\theta}$ to action-value functions $Q(x,u,\boldsymbol{\theta})$ is non-smooth and the primary objective of this optimization problem to find the policy that is the closest to the expert's policy, the authors proposed to use sub-differentials and natural gradients when solving Equation (2.21). While the former allows to approximate the gradient of the non-differentiable reward to action-value function mapping, the latter determines the gradient direction in each step such that $\hat{\pi}$ moves in the steepest descent direction.

2.2.3 System and Environment Modeling

Most early approaches [3] assume that the system model is fully known, while more recent approaches move towards inferring the objective when the model is unknown.

Discrete vs. Continuous Space models

Early IRL methods [140] were mostly demonstrated on grid problems with discrete states and actions, exhibiting less than a hundred states and four actions. Having a finite state and action space is the easiest scenario [198] and even allows for online testing [79].

To deploy IRL approaches in continuous space, it requires either discretizing the space [31] or using an approximation function such as a neural network [58]. [4] and [86] extend

the MaxEnt IRL formulation to continuous-time stochastic systems with continuous state and action spaces by replacing feature counts in the objective function with a path integral formulation based on continuous states. Similarly, [98] applies MaxEnt IRL to learn the probability distribution over navigation trajectories of interacting pedestrians using a subset of their continuous space trajectories. A mixture distribution models both the discrete and continuous navigation decisions. [105] utilizes MaxEnt IRL in high dimensional continuous domains by using a local approximation to the reward function likelihood. [41] utilizes differential dynamic programming that approximately solves continuous state-space MDP. This is done by iteratively approximating it as an LQR control problem.

In most IOC works, the state and action spaces are continuous, either using continuous-time (e.g. [38, 82] or discrete-time (e.g. [31, 55, 130]) representations.

Human Body Models

For demonstrations of articulated body movement, most papers assume that there is some knowledge of the kinematics/dynamics of the limb or the whole body either in 2D or 3D. These models can be linear or nonlinear. Linear models for studying human motion are used to simplify the problem formulation and are generally formulated for specific tasks, such as reach to grasp behavior [52], seated balancing system [154], left shoulder flexion to study neuromuscular disorders [182], and gaze movements [53]. On the other hand, nonlinear models are used to represent the task with more details. Examples include squat motions [81, 114], human arm movement [8, 16, 32, 112, 122, 142, 147, 172], human locomotion [5, 37–39, 115, 131, 133], and human running [65, 115, 131, 134, 149, 155]. To simulate and analyze dynamic models of human movement, several software systems are also available. Some of open source ones are OpenSim [45], OpenSim Moco [46], and Mujoco [179].

[38] highlights this question: What is a good mechanical model that is able to reproduce the essential characteristics of the motions under investigation? [161] suggests that models should be chosen based on the experimental protocol and hypothesis under consideration. Common nonlinear models are : (1) torque-driven models (robot models), (2) musculoskeletal models (biomechanical inspired models). Torque driven models consider joint torques as control inputs, and joint angles and velocities as outputs. Examples include [8, 32, 81, 112, 114, 142, 147, 172]. Musculo-skeletal models consider muscle activation as control inputs. [16] models the musculoskeletal arm dynamics in the sagittal plane by adding the actuator dynamics (i.e., set as the acceleration of torques equals to the neural input to muscles) to the torque-driven model. [8] models the human arm by presenting the joint torques as a combination of torques generated by the muscle forces and the moment

arms, the torques resulting from passive properties of the human arm with joint damping and the torques induced on the arm by external forces and the Jacobian of the hand position. In this model the muscle behavior is considered as the second-order low-pass filter.

[16] observed that a more complex model (i.e., modeling agonist/antagonist muscles as second order low-pass filters) does not improve the prediction results for the cost functions drastically. Therefore, the choice of the model depends on the re-targetting objective. Also, there might me some properties that are difficult to estimate. For example, [142] models the musculoskeletal system for human arm reaching motion with torque-driven model. It is mentioned that viscous frictions and elastic properties of the tissues are difficult to estimate, so they are neglected in the dynamics.

Focusing on gait, [38] classifies walking models into two classes. First, template models, which represent some major characteristics of human gait, and second, full body models, which describe motion at the joint level, with kinematic and dynamic properties that are close to a real human body. The clear advantage of full body models lies in their anthropomorphic kinematics and dynamics. However, even though considering a template model does not give insight into human behavior at the individual joint level, those models can reveal characteristic behavior of human gait. Furthermore, template models can be used for human gait analysis and humanoid gait generation. [37] explains that making use of template models for the identification of optimality criteria is an interesting approach for robotic applications for the following reasons: (1) the same model can be used for different walking scenarios; (2) the same model with different parameters can be used for human gait analysis and humanoid gait generation; (3) a sequence of several steps can be considered; (4) computational results are directly usable for robot controllers if they are based on the same template model; (5) it has potential to be used for robot control in real time. The walking model in [161] is based on a ballistic walker that contains key aspects of gait, such as a heavy swinging leg, ground impacts, and torso balancing. This model omits many aspects of human locomotion, such as muscles, ligaments, and detailed anatomical joints. However, the goal is to capture the role of the major joints involved in walking, which are often analyzed in terms of overall rotational motion and simple torques. [149] considers a high-level kinematic model perspective for human path planning. So the walking human can be modeled with the unicycle kinematic model and the complex activities performed during walking by muscles and brain in commanding and coordinating many elementary motor acts can be neglected.

Unknown Dynamics

To relax the assumption of known dynamics, one approach is to use the demonstrated trajectories to both infer the objective and identify the system model [2]. A second approach is to simultaneously learn the expert's objective function and policy and thus learn the optimal policy directly.

Sample-based methods use trajectories sampled from the optimal policy [4] or a reference distribution [26, 58, 63] to solve the IRL problem when the model of the system dynamics is unknown. In [4], sampled trajectories are used to estimate the parameters of a close form maximum entropy distribution over trajectories as well as the reward function parameters. [26] uses model-free reinforcement learning methods and importance sampling to approximate both the partition function and log-likelihood gradient. Aware of how critical the choice of sampling distribution is when using sampled trajectories to estimate the function, [58] and [63] proposed to exploit deep policy optimization and generative adversarial networks to simultaneously learn the sampling distribution that best matches the maximum entropy trajectory distribution with respect to the current reward function parameters and the objective function parameters themselves. By doing so, the proposed methods can simultaneously learn the expert's objective function and policy.

[135] proposed a model-free apprenticeship learning for transferring human behavior to the robot, evaluated on a ball-in-a-cup scenario. They rely on the implicitly encoded dynamics information in the human demonstrations rather than the need for explicit dynamics model. Similarly, [3], [30], and [42] implicitly model the agent through expert demonstrations.

Stochastic vs. Deterministic Policy/Controller

In a situation where randomness is oblivious to the agent's intended actions, deterministic policies should be optimal in theory [140, 170]. However, in practice, it is almost always the case that the agent does not have access to a perfect model of the environment and necessarily has to approximate a policy or value function that aliases many different underlying environmental states. In this case, a deterministic policy may have a systematic bias. Adding some stochasticity to the policy allows the agent to eventually break out these situations.

While both deterministic [140, 162, 170] and stochastic [25, 94, 107] policies have been assumed in the literature, almost all of the IOC approaches applied on human motion analysis have been formulated for deterministic systems. An exception is the work done

in [112] in which both deterministic and stochastic systems are considered. The authors include noise as control dependent and additive term in the dynamics of the system. Then, the necessary and sufficient condition of the control signal to be optimal is defined based on the Hamilton-Jacobi-Bellman equation, and then the IOC approach is applied on the planar biological arm movement in simulation.

2.2.4 Validation Techniques

To validate the proposed approaches, most works consider one or more of the following strategies: validation with simulated data, human data, or noise-corrupted data.

Simulation Data: Objective learning methods aim to recover the underlying objective function from demonstration trajectories. As the ground truth objective functions are impossible to obtain from human demonstrators, simulations are commonly used to validate that the recovered objective function is accurate. This is typically accomplished by implementing a controller that generates optimal demonstration trajectories given a predefined objective function. The generated demonstration trajectories are then used in the proposed algorithms to recover the original objective function. A majority of these tasks are discrete-space gridworld type applications [13, 68, 139], but also have been applied to continuous-space models [4, 82] as well.

Human Data: For human data, methods are typically validated by assessing to what extent the optimal trajectory corresponding to the recovered objective function matches either the original demonstration trajectory, or some metric derived from the demonstration. This type of validation has been applied to gait [115] and locomotion [105, 107, 155] tasks.

A notable subset of human data validation are methods that use kinesthetic teaching to provide demonstration data and use the resultant objective function to regenerate the demonstrations. While they don't tend to verify the trajectory error, they replay the trajectory on a robot to verify that the task can be replicated. These tasks have been carried on object manipulation tasks on the Barrett WAM [85] and the PR-2 [58].

Non-optimal Assumptions: While a majority of the algorithms require the demonstration trajectory to be strictly optimal, in real-life applications, strictly optimal data is impossible to guarantee if the data was not simulated, due to suboptimal trajectories or noisy sensors [31, 78, 81, 82, 142, 152, 161, 193]. Papers tend to apply strong prefiltering [114, 147, 186], or minimize the degree of optimality violations [155].

To investigate the sensitivity to approximately optimal demonstration trajectories,

some researchers validate their results on noisy data [122,157,199], or manually corrupt observation data with injected control dependent [84,112,194,195], or state dependent [8,155] noise. Other researchers use multiple demonstrations to mitigate the impact of suboptimality affecting objective function accuracy [161].

2.3 Perturbation

Perturbation is an unanticipated disturbance in motion that increases the chance of a breakdown in the human movement system [129]. Perturbation can be used for physical therapy and sports medicine, as a rehabilitative modality, and as an assessment tool. Perturbation can also be used as assessment tool to determine a client's readiness for full-contact play and reintegration into a sport or activity. The application of perturbations has been clinically proven to improve balance, joint stability, postural control and longer-term success with return-to-activity programs [129].

In most studies, three types of perturbation are applied as external forces to human body: (1) mechanical only [7,14,24,97], (2) visual only [192], and (3) mechanical and visual [40,43,137]. All of these perturbations are repeatable, except the mechanical perturbation studied in [97] for push recovery.

Mechanical Perturbation

[24] studied human arm reaching motion in the horizontal plane for two tasks: static task and dynamic task. Subjects sat in a chair and moved a parallel-link direct drive air-magnet floating manipulandum in a series of forward reaching movements performed in the horizontal plane. Their shoulders were held against the back of the chair by means of a shoulder harness. Subjects were required to make 0.25 m long reaching movement in 600+/-100 ms in the forward direction. There were three force fields. Velocity-dependent force field and two different divergent force fields, which exerted forces on the hand. Visual feedback is provided on a screen behind the apparatus. The position of the cart is visualized by a dot and the workspace safety boundaries are visualized in the form of a boundary box. In both tasks, the visual feedback of the position of the cart is deactivated during the perturbations.

[7] studied human arm reaching motion to understand how the experience of error in the feedback signal improves the subsequent motor command. In this study, participants performed a center-out reaching task while holding the handle of a planar robotic arm. The forearm of each participant was supported by an arm set that moved freely with the arm. The arm was obscured from view by a horizontal screen, upon which a projector

displayed a cursor, serving as a proxy for hand position. The perturbations were standard velocity-dependent curl force fields that pushed the hand clockwise or counter-clockwise.

[97] addressed the problem of fall avoidance by analyzing push recovery during walking on level ground based on optimization techniques. The pushes were applied by a push stick at three different locations at the spine from the back (pelvis segment, middle trunk, and upper trunk) to avoid visual prediction of the perturbation.

[14] designed an experimental paradigm that exposes sensorimotor control mechanisms and the adaptations to danger of falling and injury. [14] studied an unconstrained whole body motion where the human subjects performed squat-to-stand movements that were methodologically subjected to non-trivial perturbations. The squat-to-stand motion in this study can be considered as a whole body equivalent to the well-studied arm-reaching motion with the same level of complexity, yet it inherently involves the danger of falling and injury. The experimental setup involved a 6 degrees-of-freedom Stewart platform on which human subjects stood and performed squat-to-stand movements. The vertical velocity of the participant's center of mass was acquired by the motion capture system in real time and was used to generate perturbations in the form of a linear motion of the platform in the posterior direction. The upward motion caused a posterior displacement of the platform whereas the downward motion caused no displacement of the platform. A visual feedback-loop in the form of a LCD was showing the subjects their current center-of-mass position (COM) in the sagittal plane together with the allowed circular areas of the fully squatted and fully extended COM positions.

Visual Perturbation

[192] analyzed human arm reaching motion in the horizontal plane by applying visual perturbation. In this study, a monitor-mirror system is used that both prevented the subjects seeing their own arm and allowed projecting images into the plane of movement. The position of the hand is displayed online as a red cursor. The visibility of the hand cursor depends on the hand velocity and it changes to affect the sensory noise.

Mechanical and Visual Perturbation

In three studies performed in upper limb motor tasks at the Queens University [40, 43, 137], both visual and mechanical perturbations have been applied which are repeatable. For mechanical perturbations, they happen exactly when elbow and shoulder are at some configuration and a step torque is applied on them. The data collected in [43] suggests that when dealing with unpredictable events, such as external disturbances, vision plays a secondary role to proprioceptive feedback.

2.4 Discussion

This chapter focused on approaches for analyzing forward and inverse problems in human motion analysis. Feedforward and feedback controllers are studied in the form of forward problems where the objective functions are assumed to be known. However, this assumption is not valid in this thesis, and we are interested in addressing trajectories with unknown objective functions. Therefore, we plan to use IOC to estimate the underlying objective functions.

It is not preferred to use IRL methods as they typically require a lot more data. In these methods, special form of models is not needed and, they permit more general policies, and more general reward structures. But the cost is that many more examples and demonstrations data are required for IRL methods. These approaches are black-box methods, that when a policy is learned through a neural network, there is limited interpretation.

One way of analyzing different controllers is to study perturbed and unperturbed motions. Given the studies done on human motion, we design simulation and experimental examples to test simple ideas on including perturbation on motion and understand the role of controllers. We plan to design an experiment with mechanical perturbation to understand how the human behaves before, and after perturbation. We would like to identify how the objective functions are weighted in different situations.

Chapter 3

Inverse Optimal Control for Feedforward-Feedback Controllers

[1] Inverse optimal control (IOC) is a useful tool for elucidating the control principles of human movements from the observed trajectory data [16], and applying them to imitation learning for robotics [54]. For example, IOC can be used to analyse human arm reaching movements, to understand what objectives the central nervous system is optimizing during various types of movement [16]. Typically, the controller is assumed to optimize a cost function consisting of a weighted sum of known features, and thus the objective of IOC is to estimate the weights of the cost function. Most IOC approaches infer the underlying cost function weights by assuming that a feedforward optimal controller generates an open-loop control input trajectory [16, 54, 81, 133]. On the other hand, for linear systems and quadratic cost functions, linear quadratic regulator (LQR) approaches present a closed-form solution which generates a feedback controller [18]. There is evidence that biological movements are generated via both feedback and feedforward control schemes [111], with feedback dominating particularly when a novel action is being performed. Therefore, in this paper, we aim to develop an IOC methodology for trajectories generated by controllers with a dominant feedback component.

In order to identify the underlying cost functions given trajectories generated by feedback controllers, the inverse LQR methods have provided closed-form solutions. For example, Menner et al. formulated a semidefinite program and a linear programming convex

[1] The content of this chapter is from the following conference paper: Mahsa Parsapour, and Dana Kulić, "Recovery-Matrix Inverse Optimal Control for Deterministic Feedforward-Feedback Controllers", American control conference (ACC), pp. 4765-4770, IEEE, 2021 [153].

optimization problem to infer the cost function matrices of a quadratic cost from both optimal and non-optimal closed-loop gains when the system is linear and time invariant [123]. The objective function is defined based on the deviation from the optimal solution of the algebraic Riccati equation (ARE). Zhang et al. presented an inverse optimal control approach for the discrete-time LQR over finite time-horizons [196]. They solved the problem for a linear system, assuming the cost weighting matrices are fixed and only the state-penalty matrix elements have to be found through IOC. The optimization problem is formulated as a feasibility problem with the constraints obtained from the conditions of Pontryagin's Maximum Principle. Priess et al. in [154] presented an approach based on the inverse LQR by describing a linear matrix inequality formulation. They also proposed a gradient-based least squares minimization method that can be applied when the linear matrix inequality is infeasible. These examples are limited to linear systems and quadratic cost functions. However, real-life systems are nonlinear in practice; in this work, we formulate an IOC approach that is able to deal with both linear and nonlinear systems.

IOC for nonlinear systems is well-studied for trajectories generated under feedforward control [16, 82]. Berret et al. [16] used a bilevel optimization approach where the forward optimal control problem is solved repeatedly in an inner loop while the cost function is updated in the outer loop. Other techniques such as [82] directly compute the cost weights by minimizing the violation of the first-order necessary conditions for optimality for the observed trajectory. A recent approach [81] recovers the cost weights and estimates the boundaries of multiphase trajectories by building a recovery matrix that captures the relationship between the observed trajectory and the cost function weighting matrices. Most of the IOC methods assume that the cost functions are stationary. However, this method deals with distinct cost functions in each phase of motion. Based on the successful implementation of this approach on human squat motion segmentation [81, 114], we would like to use this idea to formulate the IOC for trajectories generated by controllers with the feedback component as well.

In this chapter, we present an optimization approach for the inverse problem of both linear and nonlinear systems, assuming a quadratic cost. Quadratic cost functions are of particular interest, since most cost functions hypothesized to be relevant for human motion analysis are quadratic in both the states and the control signal e.g. [16, 81]. For linear systems, the common LQR can be employed to generate the trajectory; however, LQR cannot be used for nonlinear systems. For nonlinear systems, iterative LQR (iLQR) method introduced in [111] is an approximation to LQR that iteratively updates the control sequence by linearizing the system. We propose an IOC approach to estimate the cost function from an observed state trajectory of a nonlinear system using recovery matrix IOC [80] and then extend it to estimate the time-varying feedback gain for LQR problems.

3.1 Problem Statement

Consider a discrete time dynamical system described by

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k), \qquad \boldsymbol{x}_0 \in \mathbb{R}^n, \tag{3.1}$$

where $\boldsymbol{f}(.,.) : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^n$ is a given (possibly nonlinear) time-invariant function, $\boldsymbol{x}_k \in \mathbb{R}^n$ and $\boldsymbol{u}_k \in \mathbb{R}^m$ are the state and input vectors at time instance k, respectively. Let us define the following quadratic cost function:

$$\begin{aligned} J = &\frac{1}{2}(\boldsymbol{x}_N - \boldsymbol{x}_G)^T \boldsymbol{Q}_f (\boldsymbol{x}_N - \boldsymbol{x}_G) \\ &+ \frac{1}{2}\sum_{k=0}^{N-1} (\boldsymbol{x}_k^T \boldsymbol{Q} \boldsymbol{x}_k + \boldsymbol{u}_k^T \boldsymbol{R} \boldsymbol{u}_k), \end{aligned} \tag{3.2}$$

where N is the number of time steps, $\boldsymbol{x}_N$ is the final state, and $\boldsymbol{x}_G$ is the given goal state. $\boldsymbol{Q}$ and $\boldsymbol{Q}_f$ are symmetric positive semi-definite ($\mathbb{S}_+^n$) state penalty and terminal state penalty matrices, and $\boldsymbol{R}$ is a positive definite ($\mathbb{S}_{++}^m$) input-penalty matrix. Here, we assume that the weighting matrices are diagonal. The running cost (the second term of equation 3.2) can be represented as

$$\begin{aligned} L(\boldsymbol{x}_k, \boldsymbol{u}_k) &= \frac{1}{2}\sum_{k=0}^{N-1} (\boldsymbol{x}_k^T \boldsymbol{Q} \boldsymbol{x}_k + \boldsymbol{u}_k^T \boldsymbol{R} \boldsymbol{u}_k) \\ &= \sum_{k=0}^{N-1} \boldsymbol{w}^T \Phi(\boldsymbol{x}_k, \boldsymbol{u}_k), \end{aligned} \tag{3.3}$$

with $\boldsymbol{w} = \frac{1}{2}\begin{bmatrix} \boldsymbol{Q}_{11}, & ..., & \boldsymbol{Q}_{nn}, & \boldsymbol{R}_{11}, & ..., & \boldsymbol{R}_{mm} \end{bmatrix}^T$, and

$\Phi(\boldsymbol{x}_k, \boldsymbol{u}_k) = \begin{bmatrix} \boldsymbol{x}_{1,k}^2, & ..., & \boldsymbol{x}_{n,k}^2, & \boldsymbol{u}_{1,k}^2, & ..., & \boldsymbol{u}_{m,k}^2 \end{bmatrix}^T \in \mathbb{R}^r$.

Given the system dynamics in equation (3.1) and the cost function in equation (3.2), two optimization problems can be defined. The first is the direct optimal control (DOC) problem, which tries to minimize the cost function J subject to the constraints defined by the system dynamics in order to find the optimal trajectory $\boldsymbol{x}_{1:N}^*$ and the optimal control input $\boldsymbol{u}_{1:N-1}^*$. The second is the inverse optimal control (IOC) problem which aims at finding the weighting matrices ($\boldsymbol{Q}$, $\boldsymbol{R}$, $\boldsymbol{Q}_f$) given the optimal trajectory $\boldsymbol{x}_{1:N}^*$ and control input $\boldsymbol{u}_{1:N-1}^*$.

We assume that the controller is an optimal feedback controller. For the DOC problem, we generate the trajectory in the feedback form using variations of the LQR. We then

formulate the IOC problem by reformulating the recovery matrix IOC algorithm proposed by Jin et al. in [80] for feedback control.

3.2 Direct Optimal Control

The LQR algorithm is unfortunately limited to linear systems. For nonlinear systems, an approach for finding the optimal controller is Iterative LQR (iLQR) [111]. This algorithm is a special case of Differential Dynamic Programming (DDP) [74,177]. The DDP is solved by a linear approximation of the nonlinear dynamics model and a quadratic approximation of the cost function along the trajectory.

In iLQR, the LQR algorithm is used iteratively to estimate an optimal control signal sequence by using a linearised approximation of the nonlinear system along the trajectory. Each iteration starts with a nominal control sequence $\boldsymbol{u}_k$, and a corresponding nominal trajectory $\boldsymbol{x}_k$ obtained by applying $\boldsymbol{u}_k$ to the open-loop dynamical system.

Let us consider deviations from the nominal sequence $\boldsymbol{x}_k$, $\boldsymbol{u}_k$ to be $\delta\boldsymbol{x}_k$, $\delta\boldsymbol{u}_k$. The linearisation is,

$$\delta\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\delta\boldsymbol{x}_k + \boldsymbol{B}_k\delta\boldsymbol{u}_k, \tag{3.4}$$

where $\boldsymbol{A}_k = \frac{\partial \boldsymbol{f}}{\partial x}|_{\boldsymbol{x}_k,\boldsymbol{u}_k}$, and $\boldsymbol{B}_k = \frac{\partial \boldsymbol{f}}{\partial u}|_{\boldsymbol{x}_k,\boldsymbol{u}_k}$ are the Jacobians of $\boldsymbol{f}(.,.)$ with respect to the state and control signal and are evaluated along $\boldsymbol{x}_k$ and $\boldsymbol{u}_k$. Therefore, the linear approximation transforms the nonlinear system into a linear time-variant system. Li et al. obtained the formulations for the iLQR from the optimality conditions on the Hamiltonian function [111]

$$\begin{aligned}\mathcal{H}_k = &\frac{1}{2}(\boldsymbol{x}_k + \delta\boldsymbol{x}_k)^T\boldsymbol{Q}(\boldsymbol{x}_k + \delta\boldsymbol{x}_k)\\ &+ \frac{1}{2}(\boldsymbol{u}_k + \delta\boldsymbol{u}_k)^T\boldsymbol{R}(\boldsymbol{u}_k + \delta\boldsymbol{u}_k)\\ &+ \delta\boldsymbol{\lambda}_{k+1}^T(\boldsymbol{A}_k\delta\boldsymbol{x}_k + \boldsymbol{B}_k\delta\boldsymbol{u}_k),\end{aligned} \tag{3.5}$$

where $\delta\boldsymbol{\lambda}_k$ is the Lagrange multiplier. [111] assumed that for some unknown sequences $\boldsymbol{S}_k$ and $\boldsymbol{v}_k$ and based on the boundary condition in the Hamiltonian function, the following equations can be substituted to the state and costate equations

$$\delta\boldsymbol{\lambda}_k = \boldsymbol{S}_k\delta\boldsymbol{x}_k + \boldsymbol{v}_k, \tag{3.6}$$

With the boundary conditions as $\boldsymbol{S}_N = \boldsymbol{Q}_f$ and $\boldsymbol{v}_N = \boldsymbol{Q}_f(\boldsymbol{x}_N - \boldsymbol{x}_G)$, we can solve for the

optimal controller by the backward recursion

$$\delta \boldsymbol{u}_k = -\boldsymbol{K}_k^x \delta \boldsymbol{x}_k - \boldsymbol{K}_k^v \boldsymbol{v}_{k+1} - \boldsymbol{K}_k^u \boldsymbol{u}_k, \tag{3.7}$$

$$\boldsymbol{K}_k^x = (\boldsymbol{B}_k^T \boldsymbol{S}_{k+1} \boldsymbol{B}_k + \boldsymbol{R})^{-1} \boldsymbol{B}_k^T \boldsymbol{S}_{k+1} \boldsymbol{A}_k, \tag{3.8}$$

$$\boldsymbol{K}_k^v = (\boldsymbol{B}_k^T \boldsymbol{S}_{k+1} \boldsymbol{B}_k + \boldsymbol{R})^{-1} \boldsymbol{B}_k^T, \tag{3.9}$$

$$\boldsymbol{K}_k^u = (\boldsymbol{B}_k^T \boldsymbol{S}_{k+1} \boldsymbol{B}_k + \boldsymbol{R})^{-1} \boldsymbol{R}, \tag{3.10}$$

$$\boldsymbol{S}_k = \boldsymbol{A}_k^T \boldsymbol{S}_{k+1} (\boldsymbol{A}_k - \boldsymbol{B}_k \boldsymbol{K}_k^x) + \boldsymbol{Q}, \tag{3.11}$$

$$\boldsymbol{v}_k = (\boldsymbol{A}_k - \boldsymbol{B}_k \boldsymbol{K}_k^x)^T \boldsymbol{v}_{k+1} - \boldsymbol{K}_k^{xT} \boldsymbol{R} \boldsymbol{u}_k + \boldsymbol{Q} \boldsymbol{x}_k, \tag{3.12}$$

Once the modified LQR problem is solved, an improved nominal control sequence can be found: $\boldsymbol{u}_k^* = \boldsymbol{u}_k + \delta \boldsymbol{u}_k$. Algorithm 1 summarises the whole process for the forward optimal control.

Algorithm 1 Iterative Linear Quadratic Regulator

Input: Nominal state and control sequence $\boldsymbol{u}_k^0$ and $\boldsymbol{x}_k^0$

1: **for** $i = 1$ to maxIter **do**
2: Initialise $\boldsymbol{S}_N \leftarrow \boldsymbol{Q}_f$ and $\boldsymbol{v}_N \leftarrow \boldsymbol{Q}_f(\boldsymbol{x}_N - \boldsymbol{x}_G)$
3: Initialise $\delta \boldsymbol{x}_k \leftarrow 0$
Backward pass:
4: Form equations (3.8)-(3.12)
Forward pass:
5: Update $\delta \boldsymbol{u}_k$ in (3.7)
6: Obtain the state sequence $\delta \boldsymbol{x}_k$ in (3.4)
7: Update $\boldsymbol{u}_k \leftarrow \boldsymbol{u}_k + \delta \boldsymbol{u}_k$
8: Update $\boldsymbol{x}_k$ from (3.1)
9: **if** No change in the value of the cost function (3.2) **then**
10: Exit for loop
11: **end if**
12: **end for**
13: **return** $\hat{\boldsymbol{u}}_{1:N-1}^* = \boldsymbol{u}_k$ and $\hat{\boldsymbol{x}}_{1:N}^* = \boldsymbol{x}_k$

Algorithm 1 gives the locally-optimal trajectories $\hat{\boldsymbol{x}}_{1:N}$ and $\hat{\boldsymbol{u}}_{1:N-1}$ for the DOC problem. When there is no further change in the cost function (line 9 of Algorithm 1), we can assume that our estimate is approximately equal to the optimum. This means that $\delta \boldsymbol{u}_k$ is approximately zero, and equation (3.7) gives the control signal as

$$\boldsymbol{u}_k = -\boldsymbol{R}^{-1}\boldsymbol{B}_k^T\boldsymbol{S}_{k+1}\boldsymbol{A}_k\delta\boldsymbol{x}_k - \boldsymbol{R}^{-1}\boldsymbol{B}_k^T\boldsymbol{v}_{k+1}.$$

Here the control signal is a combination of a linear feedback plus an additional forcing term. The forcing term is acting as the feedforward component. Because of the forcing term, it is not possible to express the optimal control law in linear state feedback form similar to the classical LQR problem.

Remark: iLQR is a trajectory optimization for nonlinear systems. The optimal solution converges to the solution of LQR, if the system is linear. To understand the similarities, let us consider the time-invariant linear system as:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k + \boldsymbol{B}\boldsymbol{u}_k, \tag{3.13}$$

By assuming that ($\boldsymbol{A}$, $\boldsymbol{B}$) is stabilizable, $\boldsymbol{B}$ has full column rank, and ($\boldsymbol{Q}^{\frac{1}{2}}$, $\boldsymbol{A}$) is detectable [123], a unique optimal feedback controller can be found. This controller is the solution to the minimization problem of (3.2) subject to (3.1) with the following form

$$\boldsymbol{u}_k = -\boldsymbol{K}_{LQR_k}\boldsymbol{x}_k, \tag{3.14}$$

The time-varying LQR optimal feedback gain $\boldsymbol{K}_{LQR}$ is defined as

$$\boldsymbol{B}^T\boldsymbol{P}_{k+1}(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K}_{LQR_k}) + \boldsymbol{R}\boldsymbol{K}_{LQR_k} = \boldsymbol{0}, \tag{3.15}$$

where $\boldsymbol{P}$ is the positive definite solution of the discrete-time algebraic Riccati equation [18]:

$$\boldsymbol{P}_{k-1} = \boldsymbol{A}^T\boldsymbol{P}_k(\boldsymbol{A} - \boldsymbol{B}\boldsymbol{K}_{LQR_k}) + \boldsymbol{Q}, \tag{3.16}$$

This equation is exactly equation (3.11) with $\boldsymbol{P}$ acting as $\boldsymbol{S}$ and the feedback gain $\boldsymbol{K}_{LQR}$ as $\boldsymbol{K}^x$. As a result, by considering the linear system (3.1), iLQR converges to the LQR solution, and we can use algorithm 1 for generating optimal trajectories for both linear and nonlinear systems.

3.3 Recovery Matrix

The recovery matrix was proposed in [80], and was used to recover the weights of trajectories generated by an optimal feedforward controller for both linear and nonlinear systems. The recovery matrix $\boldsymbol{H}(k)$ captures the relationship between available observations and

the weights of the given basis functions. The update of this matrix is iterative, and the approach for building and updating the recovery matrix $\boldsymbol{H}$ is defined as

$$\boldsymbol{H}(k) = \begin{bmatrix} \boldsymbol{H}_1(k) & -\boldsymbol{H}_2(k) \end{bmatrix} \in \mathbb{R}^{m \times (r+n)} \tag{3.17}$$

where

$$\boldsymbol{H}_1(k) = \frac{\partial \boldsymbol{f}^T}{\partial \boldsymbol{u}_k} \frac{\partial \boldsymbol{\Phi}^T}{\partial \boldsymbol{x}_k} + \frac{\partial \boldsymbol{\Phi}^T}{\partial \boldsymbol{u}_k}$$
$$\boldsymbol{H}_2(k) = \frac{\partial \boldsymbol{f}^T}{\partial \boldsymbol{u}_k} \frac{\partial \boldsymbol{f}^T}{\partial \boldsymbol{x}_k}.$$

for a single observation at $(\boldsymbol{x}_k, \boldsymbol{u}_k)$. $\frac{\partial(.)}{\partial \boldsymbol{x}_k}$ is the Jacobian matrix with respect to $\boldsymbol{x}$ evaluated at $\boldsymbol{x}_k$. The recovery matrix is updated as

$$\boldsymbol{H}(k+1) = \begin{bmatrix} \boldsymbol{H}_1(k) & -\boldsymbol{H}_2(k) \\ \frac{\partial \boldsymbol{\Phi}^T}{\partial \boldsymbol{u}_{k+1}} & -\frac{\partial \boldsymbol{f}^T}{\partial \boldsymbol{u}_{k+1}} \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ -\frac{\partial \boldsymbol{\Phi}^T}{\partial \boldsymbol{x}_{k+1}} & \frac{\partial \boldsymbol{f}^T}{\partial \boldsymbol{x}_{k+1}} \end{bmatrix}, \tag{3.18}$$

This way, each new observation is integrated into the recovery matrix. The matrix is updated until the observed trajectory is sufficient to recover the weights. By examining the rank of the recovery matrix, it is possible to detect whether sufficient observations have been collected to enable cost weight recovery. For noise-free observations, [80] proved that if rank of $\boldsymbol{H}(k)$ is equal to $r + n - 1$, the corresponding window length is sufficient for successful weight recovery. Here n is the dimension of $\boldsymbol{x}_k$ and r is the dimension of the $\boldsymbol{\Phi}(\boldsymbol{x}_k, \boldsymbol{u}_k)$ vector (i.e., the number of cost function terms). However, with noisy observations, checking the rank of $\boldsymbol{H}$ directly can lead to numerical issues. Instead of directly checking the rank condition of the recovery matrix, a new metric was introduced by [80] as:

$$\frac{\sigma_2(\bar{\boldsymbol{H}}(k))}{\sigma_1(\bar{\boldsymbol{H}}(k))} \geq \gamma, \tag{3.19}$$

with γ as a pre-defined threshold and $\bar{\boldsymbol{H}}(k) = \frac{H(k)}{||H(k)||_F}$ with Frobenius norm. This metric is defined as the ratio of the second smallest singular value, $\sigma_2(\bar{\boldsymbol{H}}(k))$, to the smallest singular value, $\sigma_1(\bar{\boldsymbol{H}}(k))$, of the normalized recovery matrix. The recovery matrix is normalized to avoid having entries close to zero in solving the optimization problem. Note that as we are not going to use the minimal observation length in our formulation, we have removed the dependency of the $\boldsymbol{H}$ matrix on the observation length.

3.4 Inverse Optimal Control using Recovery Matrix

Given the above preliminaries, we now describe the optimization formulation for recovering the weights of a feedback controller. To formulate the IOC problem, we consider the recovery matrix in the objective function. The recovery matrix is formed based on the Karush-Kuhn-Tucker (KKT) optimality criteria [80]. If $col\{\hat{\boldsymbol{w}}_k, \hat{\boldsymbol{\nu}}_{k+1}\}$ is non-zero, the following property holds:

$$\bar{\boldsymbol{H}}(k) \begin{bmatrix} \hat{\boldsymbol{w}}_k \\ \hat{\boldsymbol{\nu}}_{k+1} \end{bmatrix} = \boldsymbol{0}, \tag{3.20}$$

where $\boldsymbol{\nu} \in \mathbb{R}^n$ is the Lagrange multiplier. This means that $col\{\hat{\boldsymbol{w}}_k, \hat{\boldsymbol{\nu}}_{k+1}\} \in kernel(\bar{\boldsymbol{H}}(k))$ and $\hat{\boldsymbol{w}}$ is a successful recovery of $\boldsymbol{w}$ [80]. The equality remains true as each observation $(\boldsymbol{x}_k, \boldsymbol{u}_k)$ is added to the recovery matrix, as in equation (3.18). We find the weight estimates which minimize the Euclidean norm of the recovery matrix. Given an observed trajectory $\boldsymbol{x}_{1:N}$ and $\boldsymbol{u}_{1:N-1}$, and if the pair $(\boldsymbol{A}_k,\ \boldsymbol{B}_k)$ is controllable, the weighting matrices can be estimated through the following inverse optimal control problem

$$\begin{aligned} &\min_{\boldsymbol{w},\boldsymbol{\nu}} || \bar{\boldsymbol{H}}(k) \begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{\nu} \end{bmatrix} ||_2 \\ &\text{s.t.} \quad \sum_{i=1}^{r} \boldsymbol{w}_i = 1, \boldsymbol{R} \in \mathbb{S}_{++}^m, \boldsymbol{Q} \in \mathbb{S}_{+}^n, \end{aligned} \tag{3.21}$$

Algorithm 2 shows the steps of estimating the cost function weights, solving the optimization problem in equation (3.21). Given the inputs to the algorithm, the algorithm starts by initializing the recovery matrix. The recovery matrix is updated and normalized after each new observation $(\boldsymbol{x}_k, \boldsymbol{u}_k)$ is added. After that, the optimization problem formulated in equation (3.21) is solved, and the weights $\boldsymbol{w}_k$ are estimated. The optimization problem is solved if the rank condition between the ratio of singular values meet the predefined threshold γ; otherwise, a constant value $\frac{1}{r}$ is returned, where r is the number of features. Then the steps are repeated for the next observation time step until the end of the horizon.

Estimating the controller gain: For linear systems, most IOC approaches such as [29,123,154,196] assume that the feedback gain $\boldsymbol{K}_{LQR}$ is known. Without this assumption, the optimization problem proposed for IOC fails to provide the estimation for weighting matrices. To provide the feedback gain for IOC, [154] formulated an unconstrained optimization problem. By minimizing the following least square minimization, they estimate the feedback gain as a constant value and ignore the transient behavior of the gain.

Algorithm 2 Feedback IOC

Input: $\boldsymbol{x}^*_{1:N}$, $\boldsymbol{u}^*_{1:N-1}$ output of Algorithm 1
Output: Estimated $\hat{\boldsymbol{w}}_k$

Initialise $\boldsymbol{H}(k=1)$ - (3.17)
for $k = 2$ to N **do**
  Add new observation $(\boldsymbol{x}_k, \boldsymbol{u}_k)$
  Update $\boldsymbol{H}(k)$ - (3.18)
  normalize recovery matrix to get $\bar{\boldsymbol{H}}(k)$
  if $\frac{\sigma_2(\bar{\boldsymbol{H}}(k))}{\sigma_1(\bar{\boldsymbol{H}}(k))} \geq \gamma$ **then**
    Solve (3.21) to estimate $\hat{\boldsymbol{w}}_k$
  else
    $\hat{\boldsymbol{w}}_{i,k} \leftarrow \frac{1}{r}$
  end if
end for
return $\hat{\boldsymbol{w}}_k$

$$\hat{\boldsymbol{K}}_{LQR} = \arg\min ||(\boldsymbol{x}_{estimated}(k) - \boldsymbol{x}_{trajectory}(k)||_2. \tag{3.22}$$

The weakness of the formulation of equation (3.22) is that it is not providing time-varying feedback gain, and assuming the gain stabilises to a constant value. In practice, having the whole trajectory might not be possible which results in the need of having the time-varying feedback gain. Algorithm 2 can contribute to solve this issue. This algorithm removes the assumption on knowing the feedback gain. The proposed approach estimates the weighting matrices first, then the time-varying feedback gain can be obtained by solving equations (3.15) and (3.16).

Our goal in developing such an optimization approach is to apply it for human motion analysis. In practice, model error and noisy observations can affect the performance. When analysing human datasets, it is common to apply filtering and/or interpolation before using this kind of method in order to reduce the impact of noise [5,114,146]. However, the errors cannot be completely removed. In the numerical examples, we evaluate the performance of the proposed algorithm in terms of observation noise in order to highlight the need to improve the method for stochastic systems.

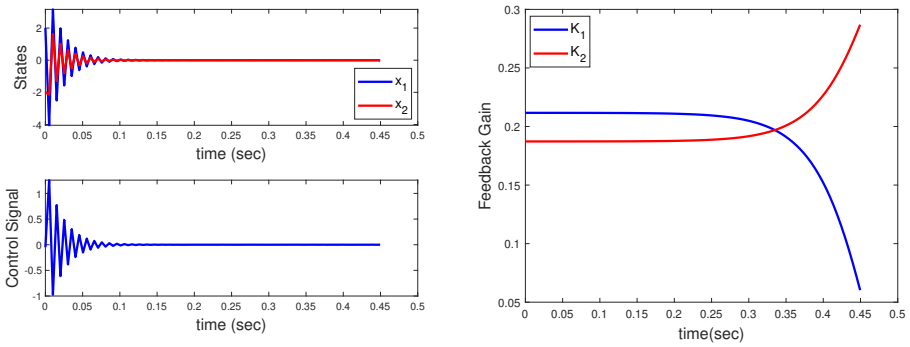
We study the performance of the IOC algorithm in two examples. For each example, weighting cost function matrices are estimated for noise-free and noisy cases.

Linear System Example

Let us consider the system

$$\boldsymbol{x}_{k+1} = \begin{bmatrix} -1 & 1 \\ 0 & 1 \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} 1 \\ 3 \end{bmatrix} u_k, \tag{3.23}$$

with the initial condition $\boldsymbol{x}_0 = [2, -2]^T$. The system is controlled by an optimal controller by minimizing the quadratic cost function in equation (3.3) with weighting matrices $\boldsymbol{Q} = \begin{bmatrix} q_1 & 0 \\ 0 & q_2 \end{bmatrix}$ and $R = r$ [80]. For this example, the true weights are $\boldsymbol{w} = [q_1, q_2, r] = [0.3333, 0.5556, 0.1111]$.



(a) (b)

Figure 3.1: Solution of the DOC problem (a) System state trajectory and control signal, (b) time-varying feedback gain.

We first generate the system trajectory using DOC (the solution is shown in Fig. 3.1), and then use it as input into the IOC algorithm described in Section IV. Also, to compare our method to the inverse LQR problem, we adapted the approach proposed in [123] for this study. The results are shown in Table 3.1. The errors shown in this table are focused on the recoverable part of the trajectory. As inverse LQR assume the feedback gain is known, the feedback controller gain $\hat{\boldsymbol{K}}_{LQR}$ is estimated through our approach in Section IV. For this example, we first consider the noise-free case, and then introduce zero mean Gaussian noise at the state output at each time step. As the noise variance increases, the accuracy of the weight estimates decreases for both the recovery matrix IOC (RM IOC) and the inverse LQR approach. Moreover, the estimate of $\hat{\boldsymbol{K}}_{LQR}$ deviates from the true

Table 3.1: Error comparison by changing the noise variance

	RM IOC	Inverse LQR	
σ_{noise}	$\|\|\boldsymbol{w} - \hat{\boldsymbol{w}}\|\|_2$	$\|\|\boldsymbol{w} - \hat{\boldsymbol{w}}\|\|_2$	$\|\|\boldsymbol{K}_{LQR} - \hat{\boldsymbol{K}}_{LQR}\|\|_2$
0.001	0.0042	0.0338	0.0012
0.01	0.0813	0.1901	0.0074
0.1	0.7433	1.2257	0.1886

gain. Overall, RM IOC is a more general approach that works for nonlinear systems as well, and its sensitivity to the increase of the noise variance is less than inverse LQR.

To understand the effect of time-varying feedback gain, the recovery error of both approaches is shown in Fig. 3.2 for different noise variances. Larger errors are observed for the inverse LQR than RM IOC at the beginning of the motion. Therefore, if we have a time-varying feedback gain, RM IOC outperforms inverse LQR.

Nonlinear System Example

We illustrate our feedback IOC approach for an inverted pendulum system. The dynamics of the system has the following form

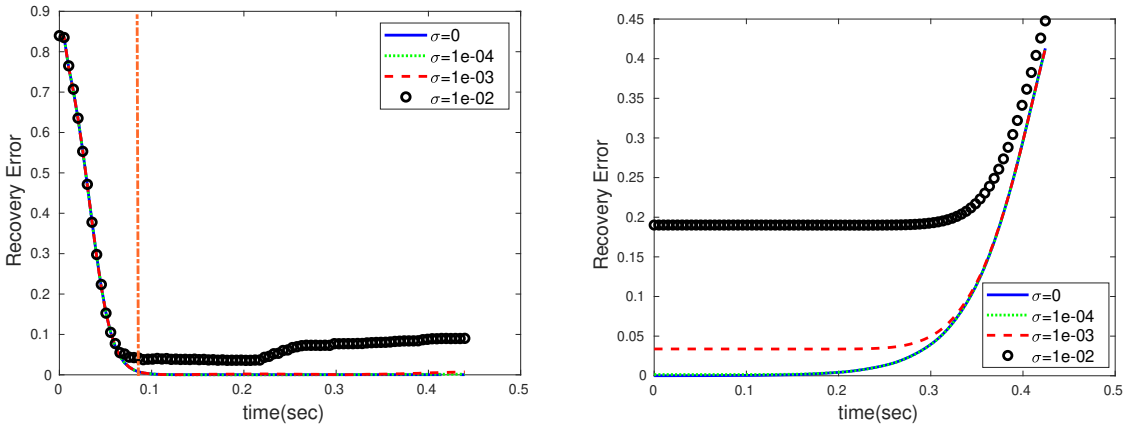
$$\ddot{\theta} = \frac{g}{l} sin\theta - \frac{\mu}{ml^2}\dot{\theta} + \frac{1}{ml^2}u, \tag{3.24}$$

where the pendulum mass $m = 1$kg, length $l = 1$m, $g = 9.8$, and friction coefficient $\mu = 0.01$ are specified. The state variables are $x_1 = \theta$ describing the angle between the vertical axis and the pendulum, and $x_2 = \dot{\theta}$ is the joint velocity. The input to the system is a torque at the pendulum base u. The system dynamics can be expressed in the state-space representation

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, u), \tag{3.25}$$

with the system state $\boldsymbol{x} = [x_1, x_2]^T$, and the control signal u. The dynamics can be discretised by Euler integration with sampling time $\Delta t = 0.001$ sec as

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta t f(\boldsymbol{x}_k, \boldsymbol{u}_k), \tag{3.26}$$



(a) (b)

Figure 3.2: Recovery error of (a) RM IOC method. The weights are recoverable when the condition (3.19) is met. The time that satisfies this condition is shown as an orange vertical dash-dot line. (b) Inverse LQR method. The behavior of the feedback gain shown in Fig. 3.1 affects the performance of the recovery error of inverse LQR at the end of the estimation. Considering the time-varying feedback gain shown in Fig. 3.1, the performance of the inverse LQR method degrades towards the end of the trajectory when the feedback gain changes magnitude, violating the constant gain assumption.

For this system, the goal is to find the control sequence $\boldsymbol{u}_k$ such that the pendulum swings up by minimizing the following objective function

$$J = \frac{1}{2}\boldsymbol{x}_N^T\boldsymbol{x}_N + \frac{1}{2}\sum_{k=0}^{N-1}(\rho_x \boldsymbol{x}_k^T\boldsymbol{Q}\boldsymbol{x}_k + \rho_u u_k^2), \tag{3.27}$$

with $\boldsymbol{Q} = \begin{bmatrix} q & 0 \\ 0 & 1-q \end{bmatrix}$, $0.1 \leq q \leq 0.9$, $\rho_x = 5e-5$, and $\rho_u = 1e-6$, respectively. For this example, we assume that ρ_u is known and the sum of the diagonal elements of the $\boldsymbol{Q}$ matrix is one. In the DOC, changing the parameter q changes the shape of the optimal state trajectory and the control signal. These trajectories, obtained from Algorithm 1, are shown in Fig. 3.3(a) and (b). Given these trajectories, the IOC problem in equation (3.21) successfully recovers the true weights as shown in Fig. 3.3(c) and (d). At the beginning of the estimation, the rank condition (3.19) is not satisfied and weights are set to 0.5.

Now, let us add zero mean Gaussian noise at the state output. Three noise levels with variance ($\sigma_1 = 1e-3$,$\sigma_2 = 1e-4$,$\sigma_3 = 1e-5$) were applied to the system. For each σ_i

the estimation was done 10 times. Fig. 3.4(a) shows the mean and standard deviation of the 2-norm of the final error between the true weights and the estimated weights. We can observe that, for the case $q = 0.4$, higher error is observed in the weight recovery compared to the case of $q = 0.9$. This is because, as the noise is increased, trajectories of the system generated with $q = 0.4$ deviate more from the nominal trajectory than those generated with $q = 0.9$. One example of estimation with $\sigma = 1e - 3$ is shown in Fig. 3.4(b) and (c). Based on the recommendation of [80], instead of considering one specific threshold level for the rank condition (3.19) in the implementation, a range for γ was set for different noise levels in Algorithm 2. As noise increases, more time steps are needed for the trajectory to satisfy the rank condition. For example, when $\sigma = 1e - 4$, the suitable range of γ is $[3, 38] \times 10^3$.

3.5 Summary

In this chapter, we formulated the inverse optimization problem for the trajectories generated by the feedforward-feedback controller for nonlinear systems and the feedback controller for linear systems. We employed the iterative LQR approach to generate the equations for a locally-optimal feedback controller for nonlinear systems, and showed that it converges to the optimal feedback controller LQR for linear systems. The norm of the recovery matrix, capturing the KKT conditions, was used as the objective function for the optimization problem. Our method contributed to providing a solution for estimating the feedback gain of linear systems where inverse LQR approaches are dependent on the given feedback gain assumption. The simulation results showed the performance of our approach for deterministic systems.

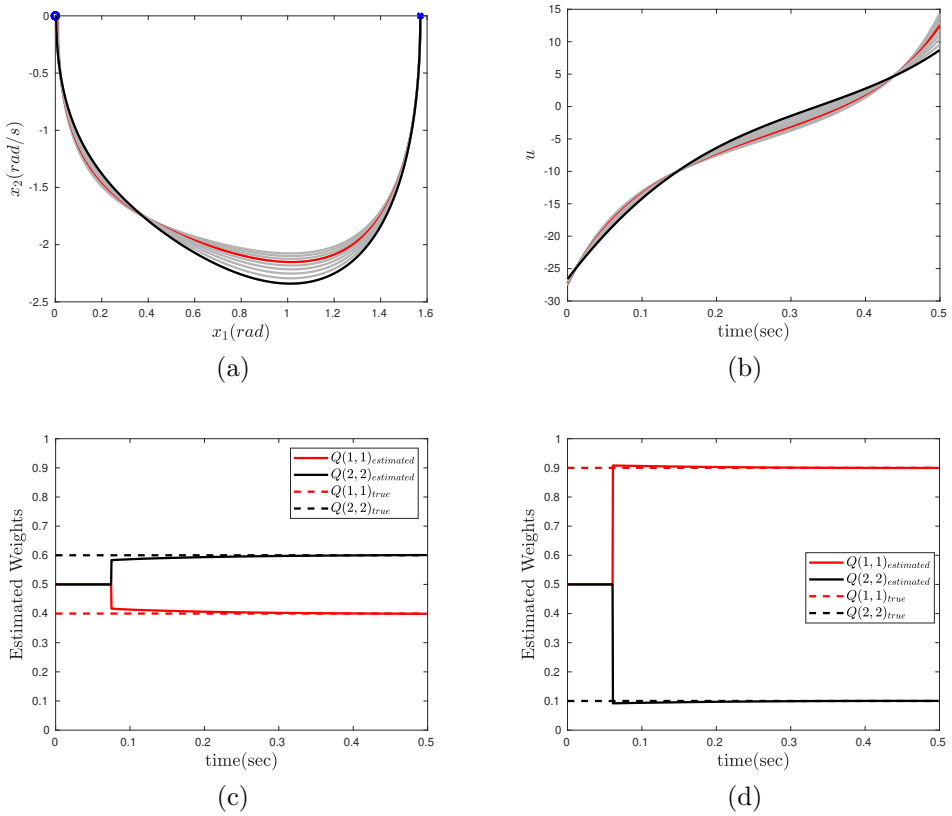
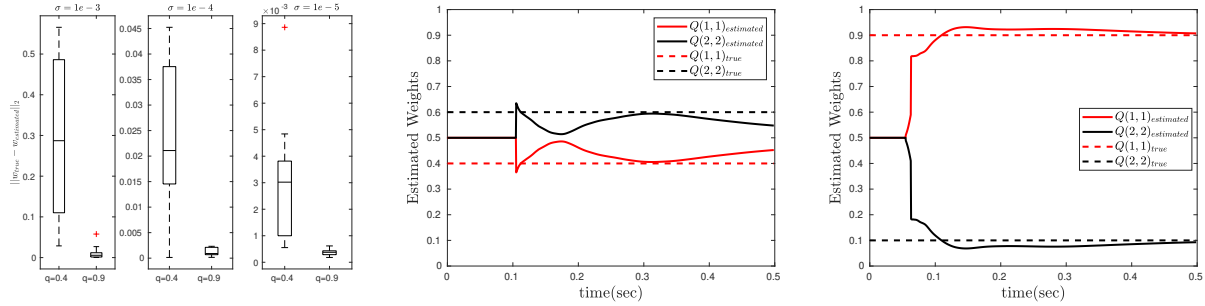


Figure 3.3: Optimal trajectories and recovered weights for the inverted pendulum system. (a) State optimal trajectories starting at position $\frac{\pi}{2}$ (rad) and zero velocity highlighted with the blue cross and converging to zero highlighted with the blue circle. All trajectories are shown in gray, the red ($q = 0.4$) and black ($q = 0.9$) trajectories are used as inputs for the IOC. (b) control signal with the same coloring as part a, (c) estimated weights when $q = 0.4$, (d) estimated weights when $q = 0.9$.



(a) (b) (c)

Figure 3.4: Estimation performance for inverted pendulum in the presence of noise. (a) Bars show the 2-norm on the final error between the true weights and the estimated ones under different noise levels. (b-c) Estimated weights with noise variance $\sigma = 1e-3$ when (a) $q = 0.4$ and (b) $q = 0.9$. The estimation error in (a) is more than (b) under this noise level. This happens as the trajectory deviates more from the noise-free trajectory when $q = 0.4$.

Chapter 4

Feedforward-Feedback Controller Decomposition

In Chapter 3, we formulated the inverse optimization problem for trajectories with both feedback and feedforward terms to learn the underlying control objective function. Learning the control objective functions of the observed behaviors enables us in understanding and predicting human motions. The results of the proposed algorithm in Chapter 3 helps in understanding rational behaviors. However, we need to develop other algorithms that can identify the controller structure and its components in order to extend the understanding of human motion analysis. In this chapter, we focus on using optimality conditions to extract information for the controller structure.

We address the following questions to analyze the human motion: What is the structure of the controller? Is the motion generated by a feedforward controller or the combination of the feedforward and feedback controllers? When was the perturbation applied? How are unperturbed and perturbed motions different? Given these questions, we present an algorithm based on value function to decide about the structure of the controller. We first describe the problem formulation and then explain the proposed algorithm. In this algorithm, the estimated control signal introduced above is going to be compared with the reference control policy to analyze the controller's structure. We define four structures to generate simulation trajectories where each includes an offline trajectory (feedforward controller). For the feedback controller, two cases of optimal and non-optimal feedback are implemented. For the non-optimal feedback, a proportional-derivative controller is used to reject perturbations, while for the optimal feedback we formulate an online optimization.

4.1 Problem Statement

Consider a discrete time nonlinear dynamical system described by

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k, \boldsymbol{d}_k), \qquad \boldsymbol{x}_1 \in \mathbb{R}^n, \tag{4.1}$$

where $\boldsymbol{f}(.,.,.) : \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is a time-invariant function, $\boldsymbol{x}_k \in \mathbb{R}^n$, $\boldsymbol{u}_k \in \mathbb{R}^m$ and $\boldsymbol{d}_k \in \mathbb{R}^m$ are the state, control input, and disturbance vectors at time instance k, respectively. The system may or may not be subject to a disturbance $\boldsymbol{d}_k$; if there is no disturbance then $\boldsymbol{d}_k$ is equal to zero. Given the control law in the general form

$$\boldsymbol{u}_k = \boldsymbol{u}_k^{FF} + \boldsymbol{u}_k^{FB}(\boldsymbol{x}_k), \tag{4.2}$$

with both feedforward and feedback components, the goal is to obtain an optimal state and control input trajectory by minimizing the following performance index

$$J(\boldsymbol{x}_1) = \boldsymbol{c}_t(\boldsymbol{x}_N) + \sum_{k=1}^{N-1} \boldsymbol{c}_r(\boldsymbol{x}_k, \boldsymbol{u}_k), \tag{4.3}$$

where N is the time horizon, $\boldsymbol{x}_N$ is the final state with $\boldsymbol{c}_t(\boldsymbol{x}_N)$ as the terminal cost function and $\boldsymbol{c}_r(\boldsymbol{x}_k, \boldsymbol{u}_k)$ as the running cost function. $V(\boldsymbol{x}_1)$ is the accumulated cost if the system is initialized at the first time step in state $\boldsymbol{x}_1$ with $\boldsymbol{u}_{1:N-1} = \{u_1, ..., u_{N-1}\}$.

Depending on the activation of the feedback component the direct optimal control problem can be formulated for two cases. The first case is having only the feedforward term $\boldsymbol{u}_k^{FF}$, where the optimization problem is solved offline with the general performance index form in equation (4.3). The second case is when the feedback component is active and the control signal is $\boldsymbol{u}_k^{FF} + \boldsymbol{u}_k^{FB}$. For this case, the optimization problem is solved online with the following specific index form

$$J(\boldsymbol{x}_k) = \sum_{i=k}^{k+N_o} k_p ||\boldsymbol{x}_i - \boldsymbol{x}_i^*||^2 + k_d ||\dot{\boldsymbol{x}}_i - \dot{\boldsymbol{x}}_i^*||^2, \tag{4.4}$$

where k_p and k_d are constants, N_o is the prediction time horizon, $\boldsymbol{x}^*$ is the desired state and $\dot{\boldsymbol{x}}^*$ is the derivative of the desired state. For the online optimization problem, a finite-horizon constrained optimal control with the performance index equation (4.4) is solved at each time step to find the control inputs on the prediction horizon. Then, only the first input is applied on the dynamics and the state obtained from the dynamics is set as the initial condition for the optimization; the horizon moves forward and the same procedure is repeated for the next time step.

4.2 Controller Structure

The main question in this section, we are interested in answering is "What is the controller structure in equation (4.2)?". To answer this question, the structures shown in Figure 4.1 could be assumed to be the forms under which the state trajectory was generated. If the motion is not perturbed, then the feedforward optimal control can describe the task best, Figure 4.1(a). However, if it is perturbed, then the objective cannot be achieved by the feedforward form Figure 4.1 (b), and a feedback must be provided to reach the desired configuration. Two possible structures could be assumed which is an optimal feedback loop or a non-optimal feedback structure. For the optimal feedback loop Figure 4.1(c), the online optimization with the performance index introduced in equation (4.4) can correct the motion online under disturbance. On the other hand, for the non-optimal feedback loop Figure 4.1(d), a simple proportional-derivative (PD) controller can reject the disturbance and achieve the desired configuration.

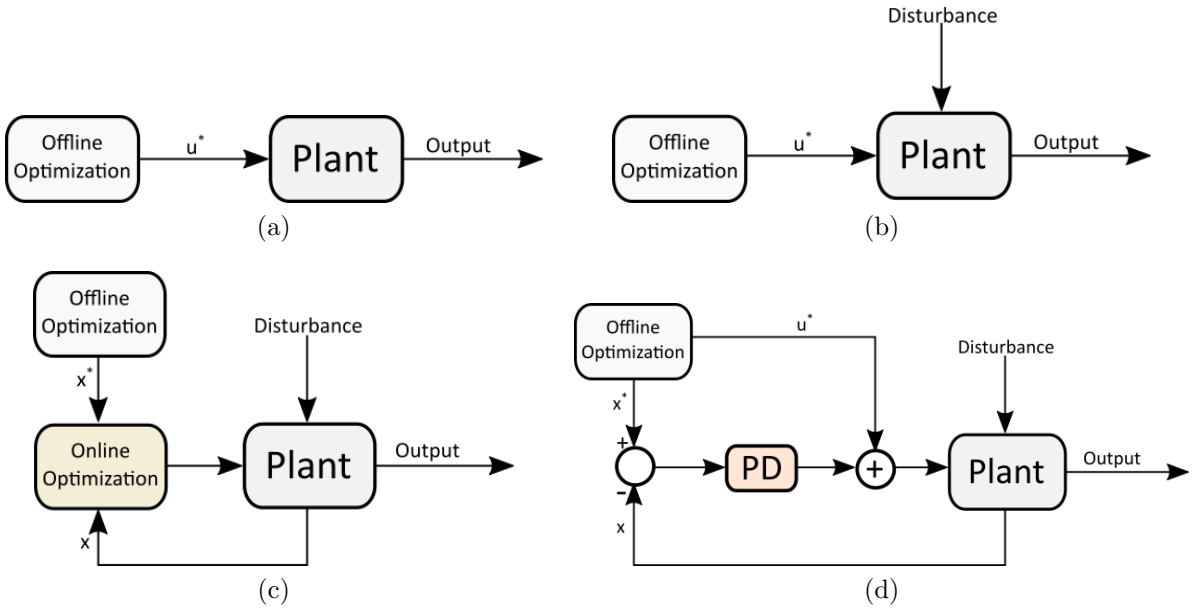


Figure 4.1: Reference trajectories (a) optimal control without disturbance, (b) optimal control with disturbance and no feedback, (c) optimal control with disturbance and optimal feedback, (d) optimal control with disturbance and non-optimal feedback.

One approach to identifying the controller structure is to look at the value function. The value function is the minimal total cost for completing the task starting from a given

state. This function captures the long-term cost for starting from a given state, and makes it possible to find optimal actions. The value function for the lth time step can be written as follows which is known as discrete-time Bellman equation

$$V_l(\boldsymbol{x}_l) = \boldsymbol{c}_r(\boldsymbol{x}_l, \boldsymbol{u}_l) + V_{l+1}(\boldsymbol{x}_{l+1}), \tag{4.5}$$

where at the Nth stage, we have $V_N(\boldsymbol{x}_N) = \boldsymbol{c}_t(\boldsymbol{x}_N)$. The Bellman optimality principle states that if a given state-action sequence is optimal, and we were to remove the first state and action, the remaining sequence is also optimal. In fact the choice of optimal actions in the future is independent of the past actions which led to the present state. Thus, optimal state-action sequences can be constructed by starting at the final state and extending backwards.

Given the state and control trajectory, and the known objective functions, the nonlinear Bellman optimality equation can be used to estimate the optimal control signal. In fact, the Bellman equation relates values of the function at different points while exploiting the fact that there exists a deterministic optimal policy that achieves the minimum value at each point. To do the estimation, we formulate equation (B.3) to recursively solve for the optimal control starting from $V_N(\boldsymbol{x}_N)$ backward in time. The optimal control law at kth stage can be estimated by minimizing the following objective function

$$\begin{aligned} \min_{\boldsymbol{u}_k} \; & \boldsymbol{c}_r(\boldsymbol{x}_k, \boldsymbol{u}_k) + V_{k+1}(\boldsymbol{x}_{k+1}) - V_k(\boldsymbol{x}_k) \\ \text{s.t.} \quad & \boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k), \end{aligned} \tag{4.6}$$

If we would like to extend the estimation in equation (4.6) for different state trajectories, the value function needs to be approximated first. Value function approximation aims at providing a scalable and effective approximation to an exact value function. There are a variety of architectures available for value-function approximation: perceptrons, neural networks, splines, polynomials, radial basis functions, support vector machines, decision trees, CMACs, wavelets, etc [102]. These architectures have diverse representational power and generalization abilities, and the most appropriate choice will heavily depend on the properties of the decision making problem at hand. In this work, we use neural networks for approximating the value function.

Algorithm 3 shows the steps to be taken to approximate the value function when there is no perturbation. First, for each initial condition in the entire state space $\boldsymbol{x}^o_1, ..., \boldsymbol{x}^o_{|\mathcal{D}|}$, we generate a set of optimal trajectories $\mathcal{D} = \{\boldsymbol{\zeta}_1, \boldsymbol{\zeta}_2, ..., \boldsymbol{\zeta}_{|\mathcal{D}|}\}$. Each optimal trajectory is made of the state and control sequence $\boldsymbol{\zeta}_i = \{\boldsymbol{x}_i, \boldsymbol{u}_i\}$ where $i = 1, 2, ...|\mathcal{D}|$. Then, a corresponding set of unperturbed optimal value function trajectories $\mathcal{V} = \{\boldsymbol{V}_1, \boldsymbol{V}_2, ..., \boldsymbol{V}_{|\mathcal{D}|}\}$

can be generated with $\boldsymbol{V}_i = [V_1, ..., V_N]^T$. Finally, the value function $\hat{V}(\boldsymbol{x}_k)$ is approximated using the radial basis function network [188] where the output of the network is the linear combination of radial basis functions as

$$\hat{V}(\boldsymbol{x}_k) = \sum_{i=1}^{m} w_i \boldsymbol{\phi}_i(\boldsymbol{x}_k), \tag{4.7}$$

where

$$\boldsymbol{\phi}_i(\boldsymbol{x}_k) = exp(-\frac{||\boldsymbol{x}_k - c_i||^2}{r_i^2}), \tag{4.8}$$

is the i^{th} Gaussian radial basis function centered at c_i and of width r_i. w_i is the associated weight of $\boldsymbol{\phi}_i(\boldsymbol{x}_k)$. The network is trained with error accuracy $1e-7$.

In Algorithm 3, we assumed that the underlying cost function is known for equation (4.3); however, when we are observing data (e.g. from a human demonstrator) we need to apply inverse optimal control first to identify the underlying cost function given the optimal state and control trajectories. Given the cost function, it is possible to approximate the value function for the entire state space.

Algorithm 3 Unperturbed Value Function Approximation

Input: Initial conditions of the entire state space $\boldsymbol{x}_1^o, ..., \boldsymbol{x}_{|\mathcal{D}|}^o$

1: **for** $i = 1$ to $|\mathcal{D}|$ **do**
2: Generate the state and control sequence $\boldsymbol{\zeta}_i = \{\boldsymbol{x}_i, \boldsymbol{u}_i\}$ by solving direct optimal control of the controller structure Fig. 4.1(a)
3: Generate the corresponding set of optimal value function trajectory $\boldsymbol{V}_i$ by (4.3)
4: $\mathcal{V} \leftarrow \boldsymbol{V}_i$
5: **end for**
6: Approximate the value function $\hat{V}(\boldsymbol{x}_k)$ (4.7)
7: **return** $\hat{V}(\boldsymbol{x}_k)$

Given the approximated value function, we propose Algorithm 4 to identify the controller structure in the presence of perturbations. In Algorithm 4, the input is the experimental data $\boldsymbol{x}_{1:N}^e$ where a disturbance might have been applied. We assume that the objective functions for all optimization problems are known. By computing the value function from Algorithm 3, we can solve the optimization problem in equation (4.6) to estimate the control signal $\hat{\boldsymbol{u}}_{1:N-1}$ and apply it to the system dynamics to get the corresponding state trajectory $\hat{\boldsymbol{x}}_{1:N}$ for the unperturbed case. Now, we can compare the experimental and the estimated state and control trajectory to output the controller structure.

The first step is to compare the experimental state trajectory and the estimated trajectory. If the difference is negligible, the controller is in the feedforward form and there is no disturbance; otherwise there is disturbance. The second step after the disturbance is identified, is to check the feedback part. As we have already assumed that the cost function is known, we can simply solve the optimization problem with equation (4.3) to have the feedforward control signal without disturbance $\hat{\boldsymbol{u}}_k^{FF}$. Then the feedback part can be computed by $\hat{\boldsymbol{u}}_k^{FB} = \boldsymbol{u}_k^e - \hat{\boldsymbol{u}}_k^{FF}$. Now, if the norm of the feedback part is really small, there is no feedback; otherwise, we need to solve the online optimization to get $\boldsymbol{u}_{1:N-1}^{Optimal}$ for the unperturbed case to identify whether the feedback is optimal or non-optimal. Depending on the systems to be analyzed, thresholds are assigned based on the 2-norm of the trajectories.

Algorithm 4 Controller Structure

Input: Experimental state $\boldsymbol{x}_{1:N}^e$

Compute the corresponding value function $\hat{V}(\boldsymbol{x}_k^e)$ by (4.3)
Estimate $\hat{\boldsymbol{u}}_{1:N-1}$ by (4.6) given $\boldsymbol{x}_{1:N}^e$ and $\hat{V}(\boldsymbol{x}_k^e)$
Compute $\hat{\boldsymbol{x}}_{1:N}$ given (4.1) and $\hat{\boldsymbol{u}}_{1:N-1}$
if $||\boldsymbol{x}_k^e - \hat{\boldsymbol{x}}_k|| < \epsilon_a$ **then**
    $msg \leftarrow$ No disturbance - Feedforward control
else
    There is disturbance
    Estimate the feedback part $\hat{\boldsymbol{u}}_k^{FB} = \boldsymbol{u}_k^e - \hat{\boldsymbol{u}}_k^{FF}$
    if $||\hat{\boldsymbol{u}}_k^{FB}|| < \epsilon_{FB}$ **then**
        $msg \leftarrow$ No feedback
    else
        There is feedback
        Solve (4.4) to obtain $\boldsymbol{u}_{1:N-1}^{Optimal}$
        if $||\boldsymbol{u}_k^e - \boldsymbol{u}_k^{Optimal}|| > \epsilon_{FB}^{OP}$ **then**
            $msg \leftarrow$ Non-optimal feedback
        else
            $msg \leftarrow$ Optimal feedback
        end if
    end if
end if
return msg

4.3 Computational Results

This section shows the results of Algorithm 3 and 4 on two nonlinear systems. We start with a two-states system as it permits full visualization of the state space and can give instruction about what problems to look out for with larger systems.

4.3.1 Inverted Pendulum

The dynamics of the system has the following form

$$\ddot{\theta} = \frac{g}{l} sin\theta - \frac{\mu}{ml^2}\dot{\theta} + \frac{1}{ml^2}u, \tag{4.9}$$

where the pendulum mass $m = 1$kg, length $l = 1$m, $g = 9.8$, and damping coefficient $\mu = 0.01$ are specified. The state variables are $x_1 = \theta$ describing the angle between the vertical axis and the pendulum, and $x_2 = \dot{\theta}$ is the joint velocity. The input to the system is a torque at the pendulum base u. The system dynamics can be expressed in the state-space representation

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, u), \tag{4.10}$$

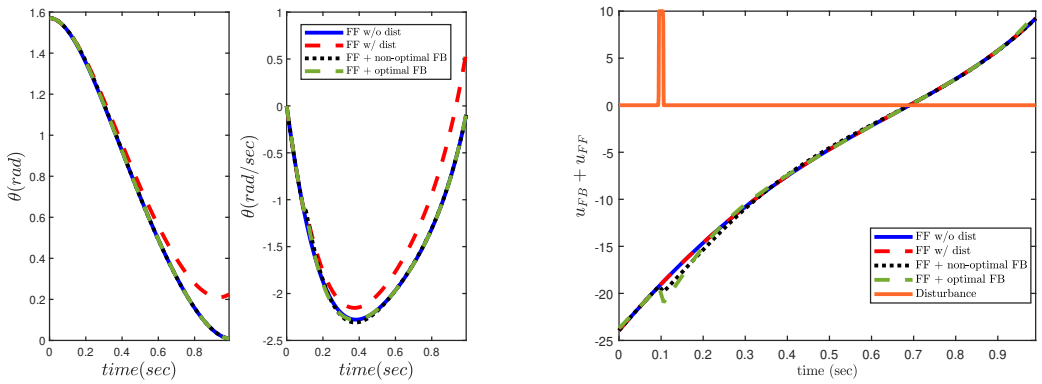
with the system state $\boldsymbol{x} = [x_1, x_2]^T$, and the control signal u. The dynamics can be discretized by Euler integration with sampling time $\Delta t = 0.001$ sec as

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta t f(\boldsymbol{x}_k, \boldsymbol{u}_k), \tag{4.11}$$

For this system, the goal is to find the control sequence $\boldsymbol{u}_k$ such that the state trajectory reaches zero by minimizing the following objective function

$$J(\boldsymbol{x}_0) = \frac{1}{2}\boldsymbol{x}_N^T \boldsymbol{x}_N + \frac{1}{2}\sum_{k=0}^{N-1}(\rho_x \boldsymbol{x}_k^T \boldsymbol{Q} \boldsymbol{x}_k + \rho_u u_k^2), \tag{4.12}$$

with $\boldsymbol{Q} = \boldsymbol{I}_{2\times 2}$, $\rho_x = 2e-7$, and $\rho_u = 8e-7$, respectively. The perturbation is added to the control signal with the magnitude of 10Nm at 0.1 sec. By, implementing the structures explained in Figure 4.1, the reference trajectories for the inverted pendulum are generated as Figure 4.2. When the perturbation is applied to the open-loop structure, the motion is diverged after 0.1 sec from the unperturbed motion, while with either non-optimal or optimal feedback there is a quick return to the offline optimal trajectory and the effect of the disturbance is rejected.



(a) (b)

Figure 4.2: Inverted pendulum (a) state trajectories, (b) control trajectories.

The value function is shown in Figure 4.3(a). Using Algorithm 3 the value function is approximated and the estimation results for the control signal are shown in Figure 4.2 (b) and (c). When there is no perturbation, the control signal is estimated to be exactly the same as the reference control trajectory. For other structures, in the estimation error we can observe that if the perturbation is active or not. Figure 4.2(c) is the estimated feedback component. For the optimal structure, after the disturbance is rejected, the signal is zero while for the non-optimal structure it has some value over the time. The output of Algorithm 4 is summarized in Figure 4.2(d). Before the perturbation is applied all signals are without disturbance which is the offline trajectory; however, after the perturbation is applied depending on if the control signal is returned back to the offline signal the output shows which condition happens.

4.3.2 Two-Link Leg

This example focuses on a two degrees of freedom system. The system is a leg model with two joints (ankle and knee), moving in the sagittal plane, Figure. 4.4. The task we are interested in studying is squatting. In this model, the phase from squatting to standing pose is going to be studied where the leg has to start at an initial position and move to a target position in a specified time interval. It has to stop at the target, and do the task with minimal energy consumption. The system dynamics is

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \boldsymbol{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \boldsymbol{g}(\boldsymbol{\theta}), \tag{4.13}$$

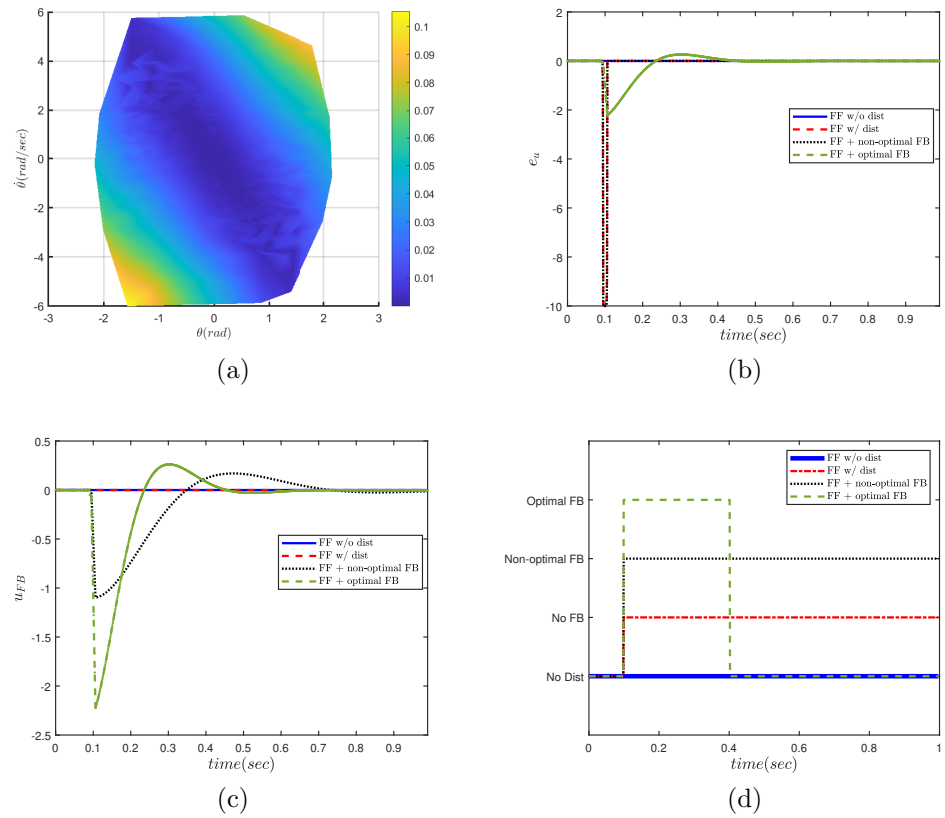


Figure 4.3: (a) True value function without disturbance over the state space for the offline optimization problem, (b) estimation error for different structures (c) feedback part of the estimation for different structures, (d) Output of Algorithm 4.

where $\boldsymbol{\theta} = [\theta_1, \theta_2]^T$ is the joint angle, $\boldsymbol{M}(\boldsymbol{\theta})$ is the positive definite inertia matrix, $\boldsymbol{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$ is the Coriolis matrix, and $\boldsymbol{g}(\boldsymbol{\theta})$ is the gravity vector. $\boldsymbol{u} = [u_1, u_2]^T$ are the torques applied to each joint. The mentioned matrices are defined as

$$\boldsymbol{M}(\boldsymbol{\theta}) = \begin{bmatrix} m_1 r_1^2 + m_2(l_1^2 + r_2^2) + a_1 & m_2 r_2^2 + a_2 \\ m_2 r_2^2 + a_2 & m_2 r_2^2 + I_2 \end{bmatrix}$$

$$\boldsymbol{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) = \begin{bmatrix} -m_2 l_1 r_2 \dot{\theta}_2 sin(\theta_2) & -m_2 l_1 r_2 (\dot{\theta}_1 + \dot{\theta}_2) sin(\theta_2) \\ m_2 l_1 r_2 \dot{\theta}_1 sin(\theta_2) & 0 \end{bmatrix}$$

$$\boldsymbol{g}(\boldsymbol{\theta}) = \begin{bmatrix} (l_1 m_2 + r_1 m_1) g cos(\theta_1) + r_2 m_2 g cos(\theta_1 + \theta_2) \\ r_2 m_2 g cos(\theta_1 + \theta_2) \end{bmatrix}$$

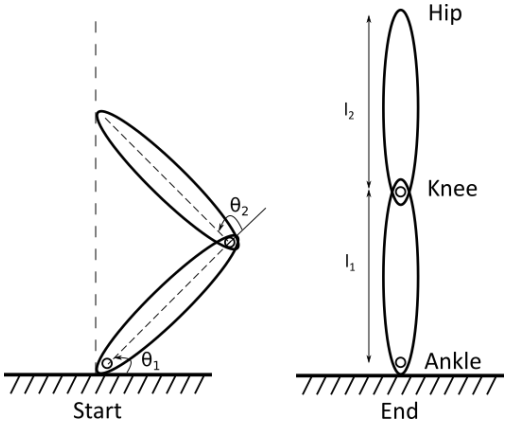


Figure 4.4: Starting and ending poses of 2-link leg model.

$$a_1 = I_1 + I_2 + 2m_2 l_1 r_2 cos(\theta_2)$$
$$a_2 = I_2 + m_2 l_1 r_2 cos(\theta_2)$$

where the mass of link i is ($m_1 = 3.84$kg, $m_2 = 6.3$kg), the length of link i is ($l_1 = 0.45$m, $l_2 = 0.38$m), the distance from the joint center to the center of mass for link i is ($r_1 = 0.2712$m, $r_2 = 0.1666$m), and the moment of inertia with respect to the corresponding center of mass for link i ($I_1 = 0.1332 kgm^2$, $I_2 = 0.0972 kgm^2$). The system dynamics can be expressed in the state-space representation

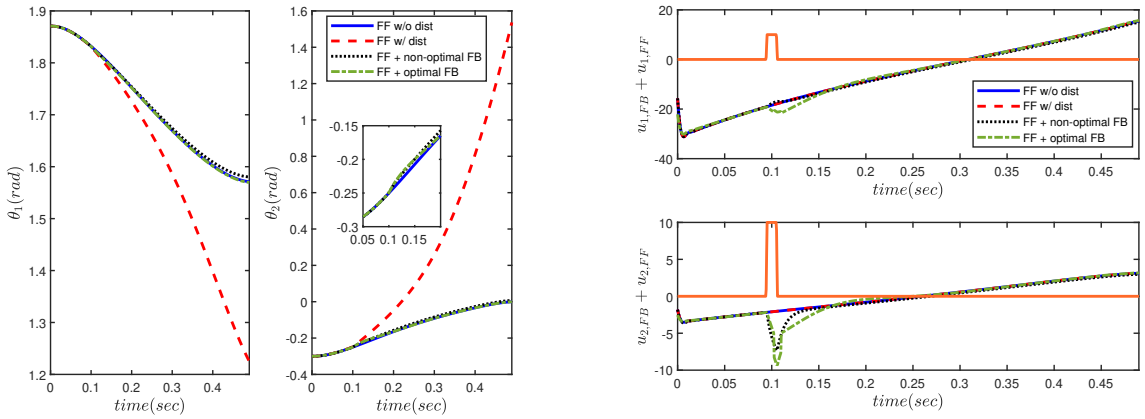
$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}), \tag{4.14}$$

with the configuration space $\boldsymbol{x} = [\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2]^T$ and torque input $\boldsymbol{u} = [u_1, u_2]^T$. The dynamics can be discretized by Euler integration with sampling time $\Delta t = 0.001$ sec as equation (4.11). The state and control trajectories are generated by solving the following optimal control using direct collocation method [90]:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{u}} \quad & \int_0^{0.5} w_1.\boldsymbol{u}_1^2 + w_2.\boldsymbol{u}_2^2 \\ \text{s.t.} \quad & \dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \\ & \boldsymbol{x}^L \leq \boldsymbol{x} \leq \boldsymbol{x}^U, \\ & \boldsymbol{u}^L \leq \boldsymbol{u} \leq \boldsymbol{u}^U, \\ & \boldsymbol{x}(0) = \boldsymbol{x}_I, \boldsymbol{x}(0.5) = \boldsymbol{x}_F. \end{aligned} \tag{4.15}$$

where the weightings in the objective function are $w_1 = 0.3$ and $w_2 = 0.7$. The goal is to start at $\boldsymbol{x}_I = [\pi/2 + 0.3, -0.3, 0, 0]^T$ and reach $\boldsymbol{x}_F = [\pi/2, 0, 0, 0]^T$. The perturbation is

added on joints through the control signal. For this example the perturbation is equally applied to both joints at 0.1 sec with the magnitude of 10Nm. The four different trajectories based on Figure 4.1 is shown in Figure 4.5. The task from squatting to standing is achieved for all controllers except the open loop controller where the joint angles are diverged from the offline optimal control trajectory. The same behavior as the previous example on inverted pendulum is observed for this model for other controllers. After the perturbation is applied, there is a quick return to the offline optimal trajectory. The magnitude of the optimal feedback control signal is higher than the non-optimal feedback control signal.



(a) (b)

Figure 4.5: Leg Model (a) state trajectories, (b) control trajectories.

For this model, the estimation error on the control signal is shown in Figure 4.6(a). In these plots, we can observe that at 0.1 sec the disturbance is activated and we can use this information to confirm that the motion was perturbed. The estimated feedback part is shown in Figure 4.6(b). The magnitude of the control signal for the optimal feedback is higher than the non-optimal feedback and it takes more time to become zero, while for the non-optimal feedback the reaction is faster but the signal does not become zero after the perturbation is rejected. From this information, we conclude that which signal is optimal. The output of Algorithm 4 is summarized in Figure 4.6(c). Before the perturbation is applied all signals are without disturbance which is the offline trajectory; however, after the perturbation is applied depending on if the control signal is returned back to the offline signal the output shows which condition happens.

To investigate the sensitivity of the algorithm to changes of parameters, we focus on the parameters that are likely to be estimated inaccurately in practice. The limb length is

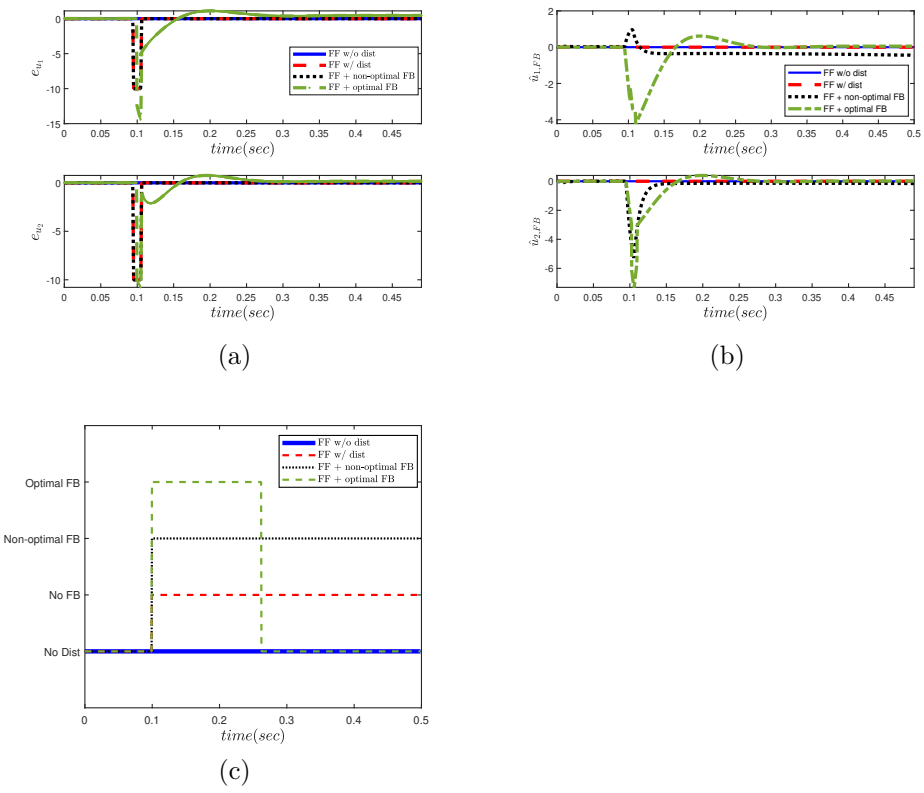


Figure 4.6: (a) Estimated control signal for unperturbed condition with feedforward control, (b) Estimated control signal for perturbed with feedforward control (c) Output of Algorithm 4 for the leg model.

likely to be accurate, since we have motion capture data. However, the estimate of limb masses and inertia distributions may be inaccurate, because the participant's whole body weight is measured and limb masses are estimated through tables that correspond to the average population [187].

To evaluate sensitivity to inertial parameters, we keep the mass of leg constant and redistribute the mass of the thigh and the shank. The redistribution of the mass is by increasing the mass of the thigh and decreasing the mass of the shank. Given these changes in the model, the algorithm reports the correct structures by two percent change of the whole leg mass. When the mass is changed more than two percent, the difference between the true state trajectory and the state trajectory with new mass is significant for the feedforward structure without disturbance, and it is classified as a perturbed trajectory. For

the feedforward structure with disturbance, the algorithm always classifies the corresponding trajectory as disturbed because the difference is larger than other specified thresholds in the algorithm and by increasing the change in the mass the difference becomes even larger. Structures with feedback information cause the output of the algorithm to be with disturbed status even before the perturbation is applied.

4.4 Discussion and Summary

This chapter looked at analyzing the underlying controller from a new perspective by looking at value function and Bellman's optimality condition. A constrained optimization problem was formulated to estimate the control signal and this optimization problem was used in an algorithm to decide about the structure of controller that generated the motion for nonlinear systems. Four structures were introduced as a reference to simulate motions and hopefully these structures can be used in the future for human motion analysis. The feedforward part of the controller was the same for all structures while the feedback parts were different. We used a proportional-derivative controller for non-optimal feedback and an online optimization for the optimal feedback.

The main component of the approach was based on value function and we represented the value function by approximating it with a neural network for the entire state space. Generating the value function for each model may take some time, but the advantage is that it is possible to save the function and use it for many different trajectories to be analyzed. If we do not know about cost functions, we input the state trajectory to the approximated value function. The value function helped us in understanding the effect of perturbation. The approximated function is valid if the disturbance is not really huge. Otherwise, the states of the system are pushed out into an entirely new state space. As such, we need to assume that the disturbance is going to stay in some neighborhood, as long as we have trajectories in that neighborhood.

This research is an introduction to answer questions on the human motion controllers and how the CNS controls the body. In this work, we assumed the structures for both the offline and online controllers are known. The next step is to see which of these structures are valid for human motions. We are going to apply an inverse optimal control on the optimal trajectories from human motions to estimate the underlying cost functions and decide about the experimental controller structures using the algorithm presented here.

Chapter 5

Human Motion Controller Components Analysis

In daily life, we often need to adapt our motion to the environment to cope with perturbations to accomplish the task successfully. To detect the perturbations in the environment, humans use their senses and mainly rely on vision, the vestibular system and the somatosensory sensor (proprioception) [97]. These sensory systems provide feedback to the central nervous system (CNS) about the orientation and position of the body. For weak perturbations, only a small reaction may be needed to recover from the perturbation. Strong perturbations may result in failure to accomplish a task, such as fall. To avoid falling due to a perturbation, three strategies can be taken: ankle strategy (i.e. movement of the ankles to apply torque to the ground) [92], hip strategy (i.e. movement of hips and arms to apply horizontal ground forces) [72], and foot placement (i.e. movement of hips and knees to take a step or squat) [117].

In this chapter, we investigate the difference between perturbed and unperturbed human motions to extract information about feedforward and feedback controller components. For this study, we modify the algorithm developed in Chapter 4 to analyze human motion in the presence of perturbations. First of all, we select the motion of interest, and describe the task. Then, we explain the dataset collected for the analysis and present the dynamic equations of motion. After that, we formulate the optimal control to describe the part of motion we will focus on. The main part of the analysis is on the inverse optimal control and how we recover the weights of the features in the objective function. Finally, we discuss the controller structure by inverse optimal control results and presenting the new algorithm.

5.1 Motion Selection

The aim of this chapter is to identify the human controller structure. To test the method proposed in Chapter 4, we need to find the suitable motion first to see how the feedback part is acting to reject perturbation. This motion has to fulfill several criteria:

- The recorded motion should be analyzable in a multi-body simulation. The trajectories generated in simulation can be compared with the data for the analysis and validation.
- The motion has to be simple enough such that the model can be represented with few segments. This helps simplifying the analysis.
- The perturbation has to be measurable. The information from this measurement can help modeling and analyzing the problem better.
- The perturbation has to be small enough to avoid injuries to participants.
- The task has to have clear objectives. If the objectives are not well-defined, the demonstrator can interpret the objectives in different ways leading to unexplained variability.

Also we could have two task specifications for generating a suitable dataset:

- **Approach one**: We could specify to the participant that the motion objective is trajectory tracking. If a disturbance occurs, they should aim to return the motion to the nominal (feedforward) trajectory. This can be handled by the approach presented in Chapter 4.
- **Approach two**: We could specify to the participant that the motion objective is goal reaching and we can look at the difference between the unperturbed and perturbed motion to see what kind of decision has been made about the new trajectory.

The first approach does not study all types of motions, while the second approach can be applied to any kind of natural motions human do everyday. For the second approach, one example that involves whole body motion is the squat exercise. The squat exercise involves multiple joints in a single motion and is considered effective for improving lower-limb muscle function [23]. Furthermore, the ability to perform this task is a prerequisite for more complex skills of daily living such as picking up an item, descending the stairs, or rising from a chair. Given that, in this chapter we will apply and modify the algorithm proposed in Chapter 4 to the squat motion.

5.2 Squat Motion

In biomechanics, squatting has been described using a variety of measured and estimated quantities [99]. Kritz et. al [99] reviewed the literature on major segment and joint during the upward and downward phase of a bilateral squat. They showed that the squat is a fundamental movement pattern that requires mobility at the ankle, hip, and thoracic spine and stability at the foot, knee, and lumbar spine [99]. Table 5.1 summarizes this review and explains the kinematic region engaged during the motion. The main regions considered are head, thoracic spine, lumbar spine, hip joints, knees, and feet/ankles. The motion of head and lumbar spine are neutral. Thoracic spine is slightly extended or neutral and held stable. Hip joints are stable and no mediolateral movement and no dropping of the hips, should stay aligned with knees. Knees are stable and they are aligned with the hips and feet. There is no excessive movement inside/out, forward/back. Feet are flat and stable, heels are in contact with the ground at all times.

Based on this pattern, we will consider three degrees of freedom in our model consisting ankle, knee, and hip. These are the most relevant joint angles to describe the motion in the sagittal plane. It has been proposed that the ability to perform a bodyweight squat at or below 90^o of knee flexion with proper symmetry and coordination is a good indicator of overall movement quality [99]. The optimal movement may be described as movement that occurs without pain or discomfort and involves proper joint alignment, muscle coordination, and posture.

The majority of previous studies for human squat motion analysis have been also focused on three DOF models with relevant joint angles (ankle, knee, and hip) in the sagittal plane. For example, Matsui et. al [120] formulated the optimal control problem for simulating the squat motion. The model is characterized by sequentially minimizing two functions for crouching–down and rising up. Lin et. al [114] formulated the inverse optimal control problem for this motion to recover the underlying cost functions. This paper showed that power and Cartesian acceleration are the important basis functions in this motion.

5.2.1 Task Description

The task instructions are to do a squat motion with bare feet or comfortable shoes and focus on keeping the feet always on the ground. The squat motion task is performed without time limitation. The task starts and ends with the standing pose. The depth of the squat is determined through an elastic band in front of the participant. The person is asked to hold their arms in front of the body as shown in Fig. 5.1. The height is adjusted

Table 5.1: Kinematic consideration for the bilateral squat motion from [99]

Anatomical Region	**Kinematic Motion**	**Optimal Pattern**
Head	Neutral	Held straight inline with the shoulders, gaze straight or slightly up
Thoracic Spine	Slightly extended	Scapulae adducted, slightly extended or neutral and held stable
Lumbar Spine	Neutral	Neutral, stable throughout movement
Hip Joints	Flexed and aligned	Stable, no mediolateral movement and no dropping of the hips, should stay aligned with knees
Knees	Aligned with feet	Aligned with the hips and feet, stable, no excessive movement inside/out, forward/back
Feet/ankles	Flat not rolling in/lifting up	Feet flat and stable, heels in contact with the ground at all times

based on the participant's preference for the depth in order not to make them exhausted after a few sets of 10 repetitions. After each set there is a break to minimize fatigue.

For the perturbation sets, another person with a push stick pushes the participant from behind on the upper trunk. This push is done from the back to avoid visual prediction of the perturbation Fig. 5.1(c). The participant is asked to continue the motion after being perturbed.

5.2.2 Data Collection

One participant is recorded for the squat motion. The following sets are done by the participant:

- **Set one**: 10 repetitions without perturbation.
- **Set two**: 10 repetitions with pushes at each repetition.
- **Set three**: 10 repetitions without perturbation.

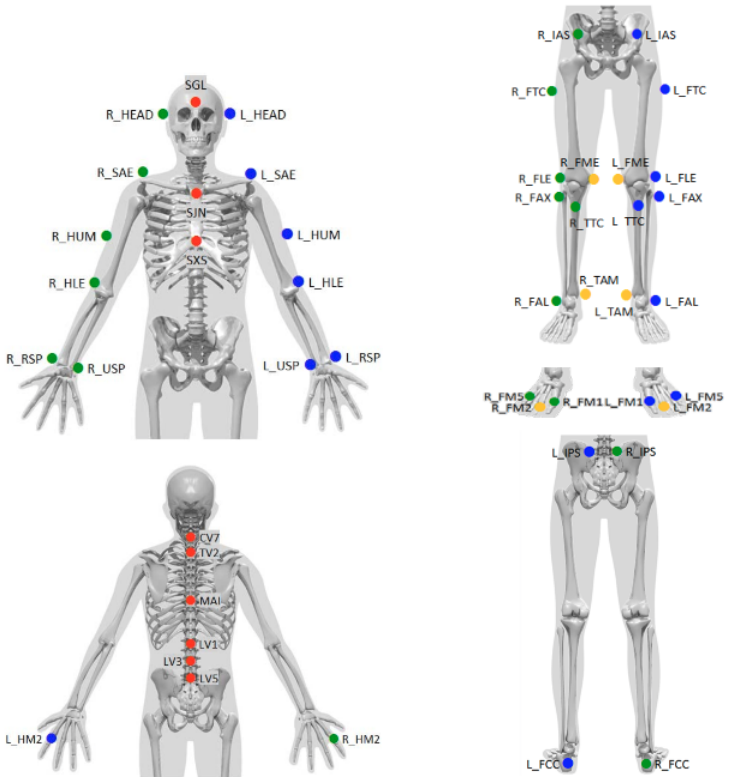
(a) (b) (c)

Figure 5.1: (a) standing phase (b) squatting phase (c) how the perturbation is applied through a push stick.

- **Set four**: 10 repetitions with pushes at random repetitions.
- **Set five**: 10 repetitions without perturbation.

The motion is recorded through a Vicon motion capture system consisting of ten infrared cameras, which record whole-body motion kinematics by tracking the spatial positions of markers with a high spatial resolution. The markers are attached with double sided adhesive tape to tight clothing worn by the participant. Markers are placed on the locations specified by Istituti Ortopedici Rizzoli (IOR) marker set [103] shown in Fig. 5.2. The position of these points in a global coordinate system is saved as data files. The motion is recorded with a rate of 100 Hz.

The force of the perturbation is measured with the 3D force sensor OMD - 50SA - 1800N from OptoForce. It is a three-axis force sensor that measures slippage and shear forces using infrared light and different kinds of optical grade elastomers to detect the smallest deformation in the shape of the outer surface, Fig. 5.3. Deforming surfaces are physically separated from the sensing element. For our experiments, the sensor is attached to a stick [97] to be able to push the subjects, Fig. 5.4. The push force is recorded with a rate of 100Hz.



(a) (b)

Figure 5.2: (a) upper body marker set, (b) lower body marker set.

To synchronize the data from the OptoForce and the Vicon, the participant is standing on two force plates. Before the experiment, the push stick is dropped down vertically on the force plate and a push is applied. In both the OptoForce data and the force plate, this point in time can be determined by the first significant increase in the force. In the same setup, also pushes during squatting are recorded and experiments in which the subject is told not to take a step in a defined area. The data for both sensors is shown in Fig. 5.5.

To get joint angles from marker positions, Biorbd library is used [126] for inverse kinematics where the model is specified in a .biomod form as shown in Fig. 5.11. The zero configuration is defined when the participant is at the upright posture. The output of Biorbd is shown in Fig. 5.6 for ankle and knee joint angles. To segment the data, we looked at the joint velocity in the knee and considered the zero joint velocity as the criterion for segmentation, as shown in Fig. 5.7. The vertical red line on the joint angle and the

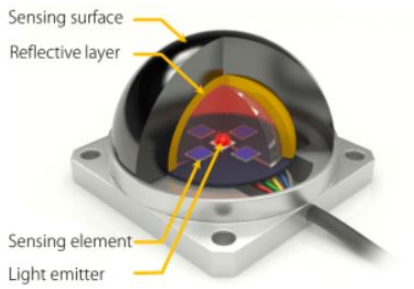


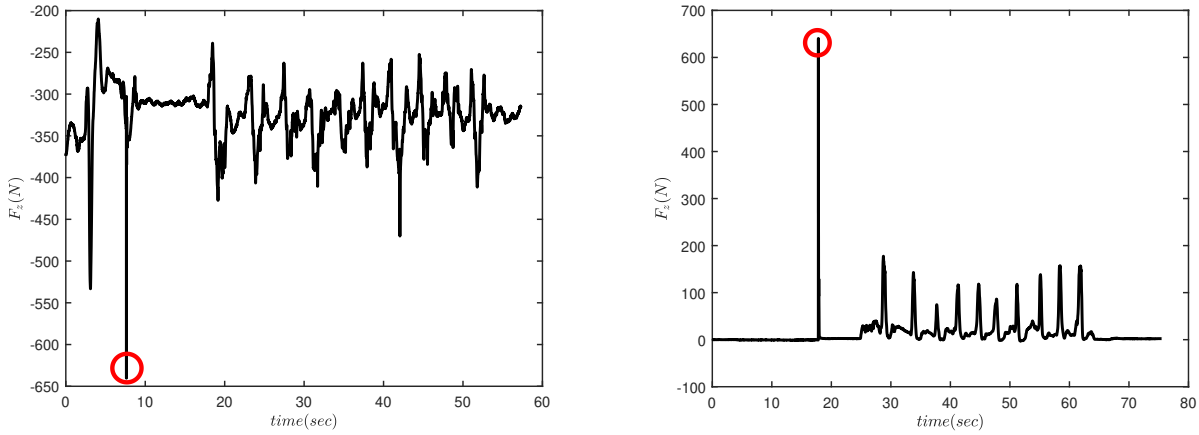
Figure 5.3: Function of the OptoForce sensor.

Figure 5.4: Picture of the push stick with markers

joint velocity for each repetition shows the peak of force of the push stick that is applied. For all repetitions, the external force is applied when the participant was squatting.

Fig. 5.8 shows the selected trials and repetitions for unperturbed and perturbed motions. Ten unperturbed motions are chosen as follows: five trials are selected from "set one" which are before the perturbation set "set two", and five trials are chosen after the perturbation set from "set three". Also, five repetitions from the perturbed motions are extracted in "set two". In this analysis our focus is on how the motion is affected after perturbation is applied, so we extract the motion after the peak of perturbation (red line) in Fig. 5.7(b). This will be the part from squatting to standing. The first observation for the comparison between trials in Fig. 5.8 is on the variation of joint angles. Before and after perturbation, the trials are consistent and the consistency is higher before the perturbation. However, when the motion is perturbed the initial joint angles are significantly impacted which is more obvious in Fig. 5.6. Also, Fig. 5.7(b) shows the planning for the ankle is not a straight path from the initial point to the final point and there is a bump in the signal. This is less observed for the knee and the hip joints.

The force plate also records center of pressure (CoP) data. CoP is a point where the ground reaction force is balanced. This data can gives information if the CoP is within the based of support (BoS) to keep the balance. BoS is the region of ground surface which



(a) (b)

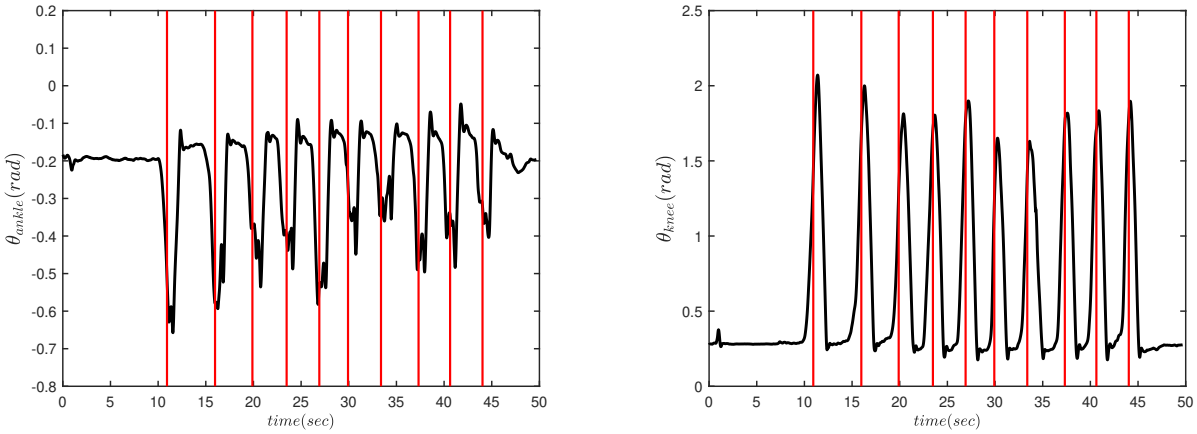
Figure 5.5: Synchronization data for the push stick and the Vicon through (a) force plate data, (b) push stick data. The red circle shows the peak force used for synchronization.

the body contacts with. Fig. 5.9 shows CoP along the position of markers from the IOR model Fig. 5.2 in the feet. To highlight the BoS, FCC is approximately the position of the heel, and Toe is the estimated position of the tip of toe. CoP reports that the motion is balanced during the motion. For unperturbed motion, the markers positions are almost flat, but they oscillate for perturbed motions and return back to the original location.

5.3 Modelling and Analysis of Human Squat Motion

5.3.1 Fixed-Base Mechanical Model of Human Squat Motion

The human sagittal model with three degrees of freedom is shown in Fig. 5.10. The base of the model at the feet is assumed to be fixed on the ground. It consists of ankle, knee and hip joints. The link angle θ_i $(i = 1, 2, 3)$ is the angle between link i and the vertical line. The joint angle $q_i = \theta_i - \theta_{i-1}$ (with $\theta_0 = 0$) is the angle between link $i-1$ and i. The first joint angle, q_1, is the same as the first link angle θ_1. For link i, m_i is its mass, l_i is the length, l_{ci} is the distance from its center of mass (COM) to joint i, and I_i is the moment of inertia around its COM. These parameters are calculated according to [44] and the calculated parameters are summarized in Appendix A.



(a) (b)

Figure 5.6: Inverse kinematic results from Biorbd for perturbed motion (a) ankle joint angle, and (d) knee joint angle; the vertical red line shows the time at which the peak of force from push stick is applied on the participant.

We assume that the model is frictionless. The equation of motion follows the dynamics presented in [191] and is

$$\boldsymbol{B\tau} = \boldsymbol{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \boldsymbol{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \boldsymbol{G}(\boldsymbol{\theta}), \tag{5.1}$$

where $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]^T$, $\boldsymbol{M}(\boldsymbol{\theta})$ is the positive definite inertia matrix, $\boldsymbol{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$ is the Coriolis matrix, and $\boldsymbol{g}(\boldsymbol{\theta})$ is the gravity vector. $\boldsymbol{\tau} = [\tau_1, \tau_2, \tau_3]^T$ are the torques applied to each joint. The mentioned matrices are defined as

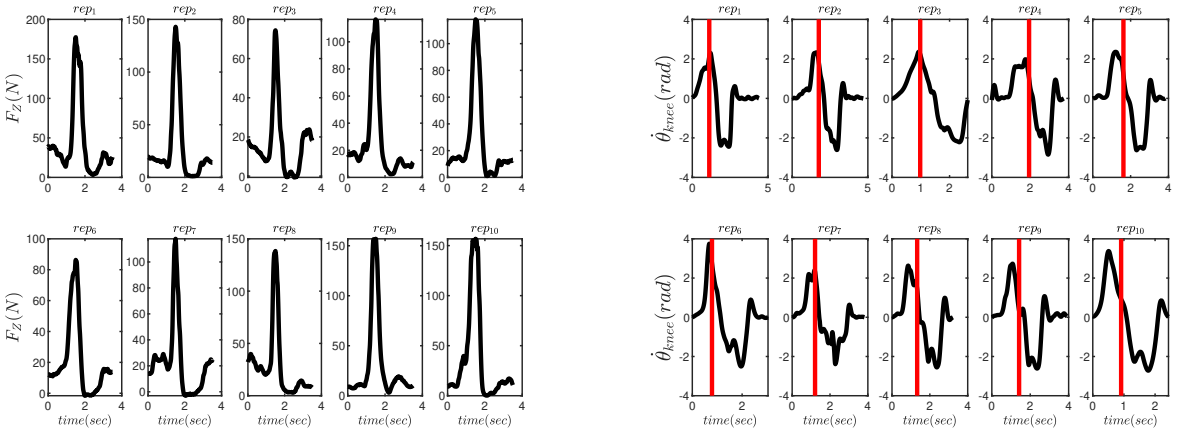
$$\boldsymbol{M}(\boldsymbol{\theta}) = \left[\alpha_{ij} cos(\theta_i - \theta_j)\right]$$

$$\boldsymbol{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) = \left[\alpha_{ij}\dot{\boldsymbol{\theta}}_j sin(\theta_i - \theta_j)\right]$$

$$\boldsymbol{G}(\boldsymbol{\theta}) = g\left[-\beta_i sin(\theta_i)\right]$$

in these formulas,

$$\alpha_{ii} = I_i + m_i l_{ci}^2 + l_i^2 \sum_{k=i+1}^{3} m_k \quad 1 \leq i \leq 3$$



(a) (b)

Figure 5.7: Segmentation on the perturbed motion (a) push stick data, and (b) knee joint velocity; the vertical red line shows the time at which the peak of force from push stick is applied on the participant.

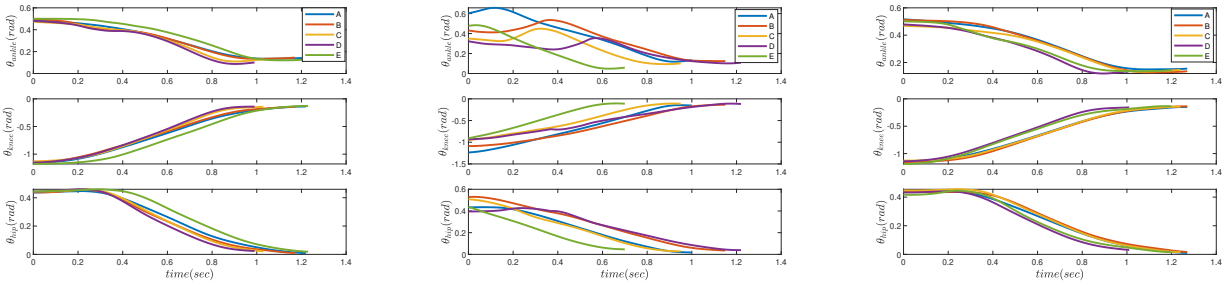
$$\alpha_{ij} = \alpha_{ji} = m_j l_i l_{cj} + l_i l_j \sum_{k=i+1}^{3} m_k \quad 1 \leq i \leq j \leq 3$$

$$\beta_i = m_i l_{ci} + l_i \sum_{k=i+1}^{3} m_k \quad 1 \leq i \leq 3$$

g is the acceleration gravity. The input matrix $\boldsymbol{B} \in \mathcal{R}^{n \times p}$

$$\boldsymbol{B} = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix}$$

The inverse kinematics in the Biorbd library gives $\boldsymbol{q} = [q_1, q_2, q_3]^T$ in Fig. 5.10. The model used in Biorbd has zero configuration when the participant is at the upright position



(a) (b) (c)

Figure 5.8: Selected trials/repetitions for analysis (a) unperturbed motions before perturbation, (b) perturbed motions, and (c) unperturbed motions after perturbation.

Fig. 5.11. Given that, $\boldsymbol{\theta}$ is obtained as follows:

$$\begin{aligned} \theta_1 &= q_1 \\ \theta_2 &= q_2 + \theta_1 \\ \theta_3 &= q_3 + \theta_2 \end{aligned} \tag{5.2}$$

5.3.2 Optimal Control Problem Formulation for Squatting to Standing Motion

We formulate the problem as a one-phase optimal control problem for squatting to standing part of motion. A general optimal control problem can be formulated as:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{u}} & \int_0^T \boldsymbol{\psi}(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \\ \text{s.t.} \quad & \dot{\boldsymbol{x}} = f(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \\ & r^{eq}(\boldsymbol{x}(0), ..., \boldsymbol{x}(T)) = 0, \\ & r^{ineq}(\boldsymbol{x}(0), ..., \boldsymbol{x}(T)) \geq 0. \end{aligned} \tag{5.3}$$

In the optimal control formulation, an objective function of Lagrangian type $\boldsymbol{\psi}(.)$ is minimized with respect to the states $\boldsymbol{x}(t)$ and the controls $\boldsymbol{u}(t)$.

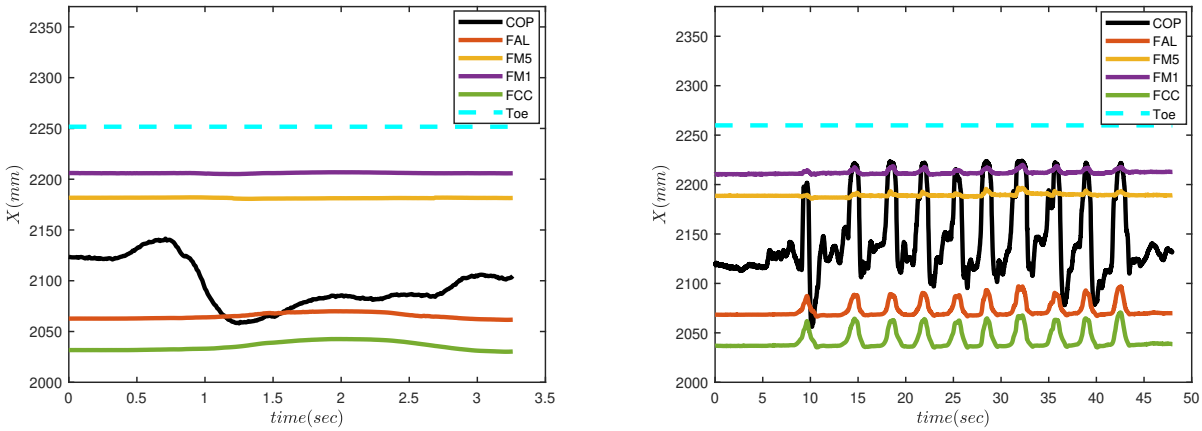


Figure 5.9: Center of pressure data from force plate (a) unperturbed motion for trial B after perturbation, (b) perturbed motion with ten repetitions. The solid lines are from measurements, and the dashed line is the estimated position. In the legend, COP shows the center of pressure data, FAL is the marker on the outside of ankle, FM1 is the marker on first knuckle of the foot, FM5 is the marker on the fifth knuckle of the foot, FCC is the marker on the back of the heel, and Toe is the estimated position of the tip of foot.

States and Control: The state vector is $\boldsymbol{x} = \begin{bmatrix}\boldsymbol{\theta}, & \dot{\boldsymbol{\theta}}\end{bmatrix}^T$ where $\boldsymbol{\theta} = \begin{bmatrix}\theta_{ankle}, & \theta_{knee}, & \theta_{hip}\end{bmatrix}^T$, and $\dot{\boldsymbol{\theta}} = \begin{bmatrix}\dot{\theta}_{ankle}, & \dot{\theta}_{knee}, & \dot{\theta}_{hip}\end{bmatrix}^T$. The control vector is $\boldsymbol{u}(t) = \boldsymbol{\tau}(t) = \begin{bmatrix}\tau_{ankle}, & \tau_{knee}, & \tau_{hip}\end{bmatrix}^T$. The system dynamics can be expressed in the state-space representation

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}), \tag{5.4}$$

The right hand side of the $\boldsymbol{f}(.)$ is represented by the forward dynamics formulation, with the equation of motions and actuation torques described in the previous section. The dynamics can be discretized by Euler integration with sampling time $\Delta t = 0.01$ sec as equation (4.11).

Objective function: The objective function is a weighted sum of features

$$\psi = \sum_{i=1}^{r} w_i \boldsymbol{\Phi}_i \tag{5.5}$$

The component features are derived from the human motion analysis literature. It has been shown that a single feature cost function can not describe a task fully, and a combination

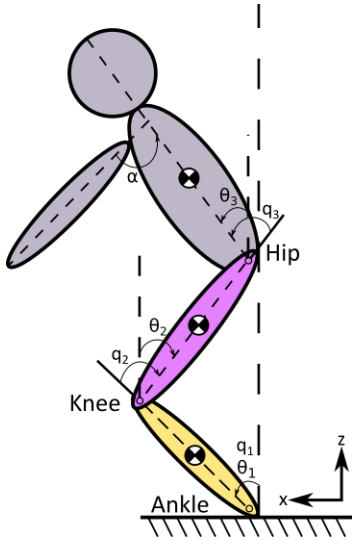


Figure 5.10: Three degrees of freedom human model for squat motion.

of features explain the human behavior better [16]. The following component features of the cost function are considered:

- Minimization of joint torques squared, which had shown good motion optimization results in dynamic motions [185]. This term produces smooth solutions with quite low energy consumption [96].

$$\Phi_1(k, \boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{u}_k^T \boldsymbol{u}_k \tag{5.6}$$

- Minimization of power [132]

$$\Phi_2(k, \boldsymbol{x}_k, \boldsymbol{u}_k) = |\boldsymbol{u}_k^T \dot{\boldsymbol{q}}_k| \tag{5.7}$$

- Balance maintenance, formulated as the distance between the foot center and the center of pressure (COP). The ground reaction force vector represents the sum of all forces acting between a physical object and its supporting surface. Balance is ensured when the sum of total momentum is equal to zero [143, 144],

$$COP = \frac{bF_{gx} - \tau_1 + cm_f g}{F_{gz}} \tag{5.8}$$

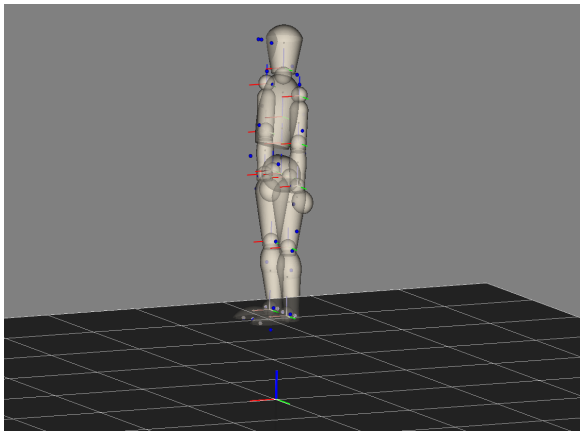


Figure 5.11: Biorbd model with x axis (red), green y axis (green) and z axis (blue) in the upright posture at the zero configuration.

where τ_1 is the ankle torque, F_{gx} and F_{gz} are horizontal and vertical components of the ground reaction force; F_{ax} and F_{az} are the resultant joint force components. m_f, l_f, a, b, c, and COP are the feet mass, the length of the base of support, the distance between the ankle and the heel, ankle height, the distance between the center of the foot and the ankle, and the distance between the COP and the toe. Fig. 5.12 shows these parameters on the foot.

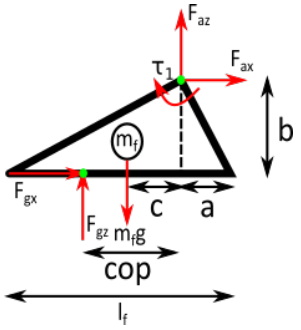


Figure 5.12: Foot model with free body diagram.

To define the vertical and horizontal ground reaction forces, we need to obtain the acceleration of center of mass for each segment in the model from the position vector $\boldsymbol{r}_x^c = \frac{\sum_{i=1}^{3} \boldsymbol{r}_{i,x} m_i}{\sum_{i=1}^{3} m_i}$ and $\boldsymbol{r}_z^c = \frac{\sum_{i=1}^{3} \boldsymbol{r}_{i,z} m_i}{\sum_{i=1}^{3} m_i}$:

$$
\begin{aligned}
\boldsymbol{r}_{1,x} &= l_{c1} sin(q_1) \\
\boldsymbol{r}_{2,x} &= l_1 sin(q_1) + l_{c2} sin(q_1 + q_2) \\
\boldsymbol{r}_{3,x} &= l_1 sin(q_1) + l_2 sin(q_1 + q_2) + l_{c3} sin(q_1 + q_2 + q_3) \\
\boldsymbol{r}_{1,z} &= l_{c1} cos(q_1) \\
\boldsymbol{r}_{2,z} &= l_1 cos(q_1) + l_{c2} cos(q_1 + q_2) \\
\boldsymbol{r}_{3,z} &= l_1 cos(q_1) + l_2 cos(q_1 + q_2) + l_{c3} cos(q_1 + q_2 + q_3)
\end{aligned}
$$

$$
\begin{aligned}
F_{gx} &= \sum_{i=1}^{3} m_i \ddot{\boldsymbol{r}}_{i,x} \\
F_{gz} &= -m_T g + \sum_{i=1}^{3} m_i \ddot{\boldsymbol{r}}_{i,z}
\end{aligned}
\tag{5.9}
$$

The CoP feature in the objective function is included as follows with x^C_{foot} the position of the center of foot

$$
\Phi_3(k, \boldsymbol{x}_k, \boldsymbol{u}_k) = (COP(\boldsymbol{x}_k, \boldsymbol{u}_k) - x^C_{foot})^2 \tag{5.10}
$$

- Angular momentum, which has been identified as a factor in motion control [169]. This feature is formulated as [69]:

$$
\Phi_4(k, \boldsymbol{x}_k) = |L(\boldsymbol{x}_k)| \tag{5.11}
$$

where L is defined as the sum of individual segment angular momenta about the foot center.

$$
L = \sum_{i=1}^{3} (\boldsymbol{r}_i - \boldsymbol{r}^C_{foot}) \times (m_i \dot{\boldsymbol{r}}_i) + \boldsymbol{I}_i \boldsymbol{\omega}_i \tag{5.12}
$$

This formulation has two terms. The first term represents the transnational movement of each segment by defining the vector $\boldsymbol{r}_i$ which is the position of center of mass of each segment with respect to ankle. $\boldsymbol{r}^c_{foot}$ is the position of the foot center. The second term is the rotational movement of each segment which is expressed by an angular velocity $\boldsymbol{\omega}_i$ multiplied by a moment of inertia $\boldsymbol{I}_i$ around the center of mass for each segment.

As each objective feature has different units, we require normalize each term to enable comparison. Each objective term Φ_i is normalized by multiplying it by constant β_i which is defined as

$$\beta_i = \frac{1}{\sum_{k_1}^{T} \Phi_i(k, \boldsymbol{x}_k, \boldsymbol{u}_k)}$$

Constraints: The set of constraints included in the optimal control are defined as follows:

- System dynamics based on equation (5.1).
- *Initial and end positions*: We specify the initial and final conditions based on the values of the joint positions and velocities at which the segment starts in the squatting and ends at the standing posture.
- *Box constraints*: For the joint angles, the ranges are defined based on the range obtained from the dataset.

$$\begin{aligned} 0^o \leq q_{ankle} \leq 40^o \\ -139^o \leq q_{knee} \leq 0^o \\ 0^o \leq q_{hip} \leq 180^o \\ -10 \leq \dot{\theta}_j \leq 10 \;\; j = 1, 2, 3. \\ -200 \leq \tau_i \leq 200 \;\; i = 1, 2, 3. \end{aligned} \tag{5.13}$$

Software tool: we solve the optimal control problem with the OptimTraj library [90]. The solution of the optimal control problem is carried out using the direct collocation method.

In summary, the optimal control problem is defined as follows:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{u}} \quad & \sum_{k=1}^{T} w_1\beta_1\boldsymbol{u}_{1,k}^2 + w_2\beta_2\boldsymbol{u}_{2,k}^2 + w_3\beta_3\boldsymbol{u}_{3,k}^2 + w_4\beta_4|\boldsymbol{u}_{1,k}\dot{\boldsymbol{q}}_{1,k}| + w_5\beta_5|\boldsymbol{u}_{2,k}\dot{\boldsymbol{q}}_{2,k}| + w_6\beta_6|\boldsymbol{u}_{3,k}\dot{\boldsymbol{q}}_{3,k}| \\ & + w_7\beta_7(COP_k - x_{foot}^C)^2 + w_8\beta_8|L_k| \\ \text{s.t.} \quad & \boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k), \\ & \boldsymbol{x}^L \leq \boldsymbol{x} \leq \boldsymbol{x}^U, \\ & \boldsymbol{u}^L \leq \boldsymbol{u} \leq \boldsymbol{u}^U, \\ & \boldsymbol{x}_1 = \boldsymbol{x}_I, \boldsymbol{x}_T = \boldsymbol{x}_F. \end{aligned} \tag{5.14}$$

where the weightings w_i in the objective function are estimated by inverse optimal control first and the starting point $\boldsymbol{x}_I$ and ending point $\boldsymbol{x}_F$ are the same as the data.

5.3.3 Inverse Optimal Control

Here we use the recovery matrix inverse optimal control method by Jin in [77] to estimate the weights of the objective function. As the objective function does not change for the upright motion, we assume that the weights are not time-varying and build the recovery matrix for the whole window size. Therefore, the recovery matrix is defined as

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{H}_1 & -\boldsymbol{H}_2 \end{bmatrix} \tag{5.15}$$

with

$$\boldsymbol{H}_1 = \boldsymbol{F}_u \boldsymbol{F}_x^{-1} \boldsymbol{\Phi}_x + \boldsymbol{\Phi}_u$$

$$\boldsymbol{H}_2 = \boldsymbol{F}_u \boldsymbol{F}_x^{-1} \boldsymbol{Z}$$

Here, F_x, F_u, Φ_x, Φ_u and $\boldsymbol{Z}$ are defined as:

$$F_x(t) = \begin{bmatrix} I & -\frac{\partial \boldsymbol{f}^T}{\partial \boldsymbol{x}_2^*} & & \\ & I & \ddots & \\ & & \ddots & \frac{\partial \boldsymbol{f}^T}{\partial \boldsymbol{x}_{T-1}^*} \\ & & & I \end{bmatrix} \tag{5.16}$$

$$F_u(t) = \begin{bmatrix} \frac{\partial \boldsymbol{f}^T}{\partial \boldsymbol{u}_2^*} & & \\ & \ddots & \\ & & \frac{\partial \boldsymbol{f}^T}{\partial \boldsymbol{u}_{T-1}^*} \end{bmatrix} \tag{5.17}$$

$$\Phi_x = \begin{bmatrix} \frac{\partial \Phi}{\partial x_2^*} & \frac{\partial \Phi}{\partial x_3^*} & \cdots & \frac{\partial \Phi}{\partial x_{T-1}^*} \end{bmatrix}^T \tag{5.18}$$

$$\Phi_u = \begin{bmatrix} \frac{\partial \Phi}{\partial u_2^*} & \frac{\partial \Phi}{\partial u_3^*} & \cdots & \frac{\partial \Phi}{\partial u_{T-1}^*} \end{bmatrix}^T \tag{5.19}$$

$$\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{0} & \frac{\partial \Phi}{\partial x^*_{T-1}} \end{bmatrix}^T \tag{5.20}$$

with $\frac{\partial f}{\partial x^*_T} = I$. The optimal trajectory is governed by the KKT optimality criteria with the following Lagrange function

$$L = \sum_{k=1}^{T} \boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \sum_{k=1}^{T} \boldsymbol{\lambda}_k^T (\boldsymbol{x}_k - \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k)) \tag{5.21}$$

where $\boldsymbol{\lambda}_k \in \mathbb{R}^n$ is the Lagrange multiplier. The optimal solution is given by

$$\frac{\partial L}{\partial \boldsymbol{x}^*_{1:T}} = 0$$
$$\frac{\partial L}{\partial \boldsymbol{u}^*_{1:T}} = 0$$

Based on this condition, the optimization problem is defined as follows

$$\begin{aligned} \min_{\boldsymbol{w}, \boldsymbol{\lambda}} \quad & \boldsymbol{H}(t) \begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{\lambda} \end{bmatrix} \\ \text{s.t.} \quad & \sum_{i=1}^{r} \boldsymbol{w}_i = 1, \quad \boldsymbol{w} \geq \boldsymbol{0}. \end{aligned} \tag{5.22}$$

5.3.4 Controller Structure Analysis

To analyze the controller structure for the squat motion dataset, we present Algorithm 5 for the current selected squat motion trial to output the status of the controller. In Algorithm 5, the input is the experimental state trajectory $\boldsymbol{x}^e_{1:N}$ where a disturbance might have been applied. We first compute the control signal by inverse dynamics to obtain $\boldsymbol{u}^e_{1:N-1}$. Then, an inverse optimal control is applied on the state and control trajectory to recover the weights. Using the identified weights of the active objective features, the optimal control problem is solved to estimate the state and control trajectory $\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}}$. The experimental and estimated state trajectories are compared. If the difference is negligible, the controller is in the feedforward form and there is no disturbance; otherwise there is disturbance. Next, the experimental and estimated control signals are compared to determine if the controller has only a feedforward term or not. The threshold in this algorithm is set by choosing two of trials as the train data.

Algorithm 5 Human Controller Structure

Input: Experimental state trajectory $\boldsymbol{x}^e_{1:N}$

Compute control signal $\boldsymbol{u}^e_{1:N-1}$ from inverse dynamics
$\hat{\boldsymbol{w}} \leftarrow$ Apply IOC on the state and control trajectory
$\hat{\boldsymbol{x}}, \hat{\boldsymbol{u}} \leftarrow$ Apply DOC
if $||\boldsymbol{x}^e_k - \hat{\boldsymbol{x}}_k|| < \epsilon_d$ **then**
  $msg \leftarrow$ No disturbance - Feedforward control
else
  $msg \leftarrow$ There is disturbance
  if $||\boldsymbol{u}^e_k - \hat{\boldsymbol{u}}_k|| < \epsilon_{FF}$ **then**
    $msg \leftarrow$ The controller has only a feedforward term.
  else
    $msg \leftarrow$ The controller includes more terms than a feedforward term.
  end if
end if
return msg

Table 5.2: Confusion matrix for classifying trajectories as disturbed (D) or not disturbed (ND).

		Actual		
		D	ND	Total
Detected	D	5	2	7
	ND	0	8	8
	Total	5	10	15

Table 5.2 shows the output of the algorithm for classifying the trajectories as disturbed or not disturbed. Out of 15 trials, 13 trials are classified correctly. However, 2 "not disturbed" trials are in the category of disturbed as the error between the direct optimal control and data was more than the threshold.

To better understand each part of the algorithm, we analyze the control strategy estimated by the optimization problem given by (5.22). Estimated weights for all trials are summarized in Table 5.3. In this table, the recovered weights correspond to the torque squared at ankle, knee and hip, power at ankle, knee and hip, center of pressure around the center of foot, and the angular momentum about the center of foot, respectively.

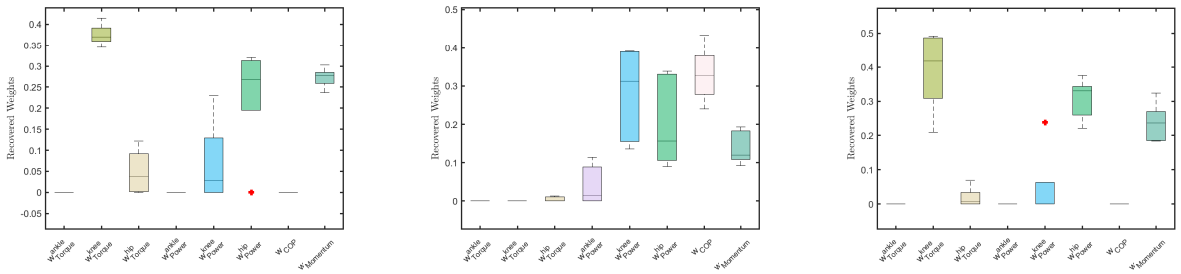
The recovered weights are also shown as bar graphs in Fig. 5.13 to better compare the effect of each weight before perturbation, perturbed, and after perturbation. The

Table 5.3: Inverse optimal control results. In each trial, the superscripts "bp", "p", and "ap" stand for before perturbation, perturbed, and after perturbation, respectively. For each result, the root mean-square error (RMSE) of trajectories is also computed.

Trials	Recovered Weights $[w_{Torque}^{ankle}, w_{Torque}^{knee}, w_{Torque}^{hip}, w_{Power}^{ankle}, w_{Power}^{knee}, w_{Power}^{hip}, w_{COP}, w_{Momentum}]$	RMSE
T_A^{bp}	[0.0000 0.3653 0.0000 0.0000 0.0966 0.2608 0.0000 0.2773]	0.0153
T_B^{bp}	[0.0000 0.3838 0.0021 0.0000 0.0276 0.3209 0.0000 0.2657]	0.0181
T_C^{bp}	[0.0000 0.3704 0.0817 0.0000 0.0000 0.3109 0.0000 0.2370]	0.0177
T_D^{bp}	[0.0000 0.4141 0.0384 0.0000 0.0002 0.2682 0.0000 0.2791]	0.0170
T_E^{bp}	[0.0000 0.3452 0.1221 0.0000 0.2296 0.0000 0.0000 0.3031]	0.0178
T_A^{p}	[0.0000 0.0000 0.0000 0.0796 0.1612 0.3392 0.2399 0.1801]	0.0318
T_B^{p}	[0.0000 0.0000 0.0000 0.0143 0.3913 0.1118 0.3628 0.1198]	0.0330
T_C^{p}	[0.0000 0.0000 0.0133 0.0000 0.3899 0.1564 0.3275 0.1129]	0.0466
T_D^{p}	[0.0000 0.0000 0.0000 0.1138 0.3122 0.0899 0.2914 0.1927]	0.0537
T_E^{p}	[0.0000 0.0000 0.0093 0.0000 0.1359 0.3276 0.4347 0.0926]	0.0905
T_A^{ap}	[0.0000 0.4175 0.0214 0.0000 0.0000 0.3751 0.0000 0.1860]	0.0105
T_B^{ap}	[0.0000 0.4853 0.0000 0.0000 0.0000 0.3305 0.0000 0.1842]	0.0129
T_C^{ap}	[0.0000 0.4914 0.0000 0.0000 0.0000 0.2725 0.0000 0.2361]	0.0130
T_D^{ap}	[0.0000 0.3412 0.0690 0.0000 0.0045 0.3329 0.0000 0.2525]	0.0208
T_E^{ap}	[0.0000 0.2099 0.0069 0.0000 0.2383 0.2207 0.0000 0.3242]	0.0223

results suggest that the participant minimized joint torques on knee and hip before and after perturbation while the effect of this term is negligible when the motion is perturbed. The weight on the power on the hip is greater than the power on knee before and after perturbation. This makes sense as more body weights is in the third segment of the dynamical model, and more energy is needed for the hip. However, the weight on the power on ankle is increased when the motion is perturbed Fig. 5.13(b). The weights on the ankle and knee power might suggest that the participant used the ankle strategy to maintain balance and reject the perturbation to continue the task. Also, when the participant is perturbed the term on the center of pressure appears. This term appears as a feedback term in the objective function to balance forces around the center of foot. Also, the center of pressure includes a term on the ankle torque which confirms the ankle strategy is being used to balance. The IOC also showed the effect of angular momentum in all trials. It is slightly higher when the motion is not perturbed. For unperturbed trials,

the angular momentum is quite low because the person is just going up and down, and higher weight is observed. Whereas the participant uses angular momentum in perturbed trials and this term increases to minimize the rotational movements in the body. The participant lowers the weight on the angular momentum to allow themselves to rock its body to recover from the disturbance.



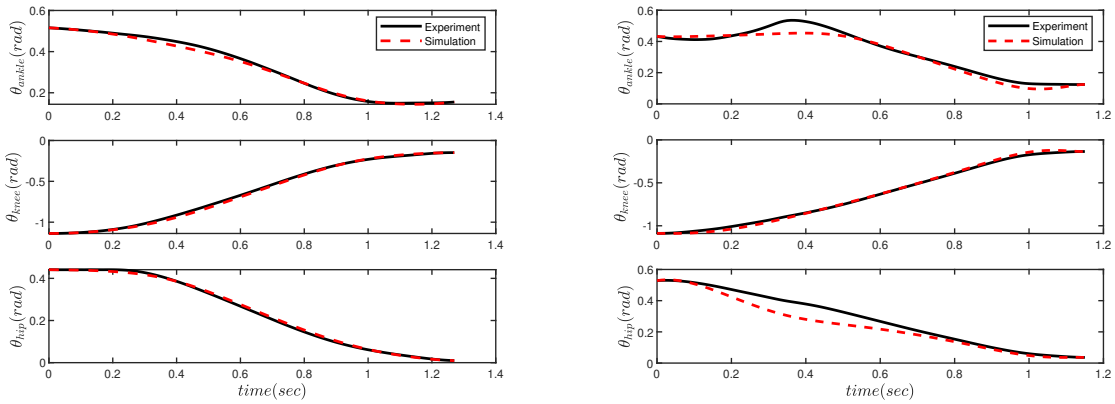
(a) (b) (c)

Figure 5.13: Recovered weights from inverse optimal control for (a) unperturbed motion before perturbation, (b) perturbed motion, and (c) unperturbed motion after perturbation.

To validate the recovery results, we simulate the trajectory for each condition by solving the optimal control problem based on the recovered cost functions. Fig. 5.14 shows the simulated trajectory for one unperturbed trial and one perturbed trial. For the unperturbed trial, the simulated trajectory using the recovered cost functions fits well the real data, indicating the validity of the IOC. For the perturbed trial, there are remaining differences for the ankle and hip trajectories, which suggests that additional terms in the cost function might be needed to regenerate the motion.

The root mean square error (RMSE) of trajectories for each trial is summarized in Table 5.3 and Fig. 5.15 presents the RMSE for each joint trajectory separately. When the participant is perturbed, more error is observed on all joint angles. Errors on unperturbed motion is relatively smaller with a lower variation. The errors on each joint for each condition is almost in the same range; however, the error on the ankle joint is more than other joints when the participant is perturbed.

To understand the controller structure better, we can also look at the control signal we get from IOC and the inverse dynamics. Fig. 5.16 shows the control signals for one unperturbed and one perturbed trial. The pattern of the signal for the unperturbed trial in Fig. 5.16 (a) is similar between the inverse dynamics and the feedforward optimal control. However, the differences for the perturbed trial in Fig. 5.16(b) shows the feedback terms



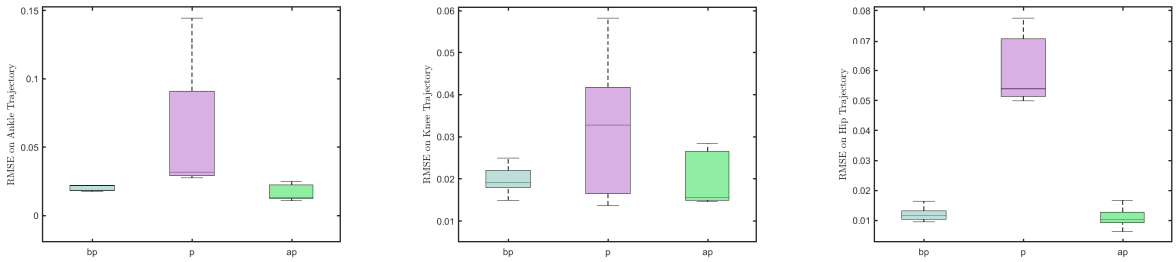
(a) (b)

Figure 5.14: Simulated motion using the recovered cost functions. Solid blue lines are from the experimental data, and dashed red lines are the simulated trajectories for (a) unperturbed motion trial A after perturbation and, (b) perturbed motion repetition B.

regenerated the command with a similar pattern; however, there is some error that might be improved by including more terms in the feedback components.

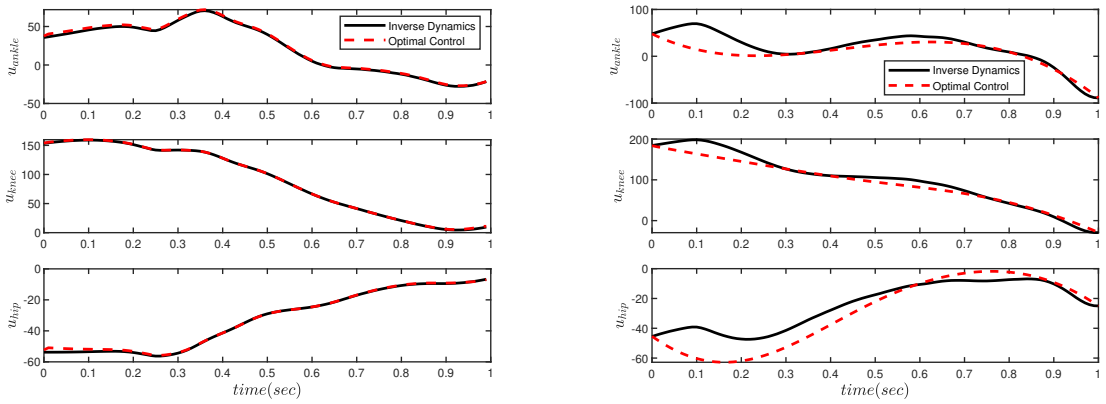
5.4 Summary

In this chapter, we chose and collected data to study the effect of perturbation on an squat motion. An analytical 3 degrees of freedom model with joints on ankle, knee and hip was formulated to analyze the data. We also formulated the direct optimal control to simulate the trajectories and compare the state trajectories with the dataset. To infer the objective functions, a recovery-based inverse optimal control was applied and the results suggest that the participant used the ankle strategy to balance and reject the perturbation to continue the task. The proposed approach can differentiate between perturbed and unperturbed trajectories with an accuracy of 87%.



(a) (b) (c)

Figure 5.15: RMSE of simulated and data trajectories for (a) ankle joint, (b) knee joint, and (c) hip joint.



(a) (b)

Figure 5.16: Control signals from inverse dynamics solution and inverse optimal control solution for (a) unperturbed trial D before perturbation and (b) perturbed trial A.

Chapter 6

Conclusions and Future Work

6.1 Summary

In this thesis, we approached the study of human feedforward and feedback controllers using optimal control theory. For this study, we focused on understating how the optimality criteria can be used to see what controller is used under different conditions. For this analysis, we first proposed two main algorithms in simulation, and at the end tested and modified the algorithm on real data.

Recovering Objective Functions for Feedforward-Feedback Controllers

We formulated the inverse optimization problem for the trajectories generated by iterative linear quadratic regulators. The controller in these trajectories have both feedforward and feedback components. The feedback controllers in these trajectories are locally optimal for nonlinear systems. Through this formulation we showed that the controller converges to the optimal feedback controller LQR for linear systems. Our method provided a solution for estimating the feedback gain of linear systems where inverse linear quadratic regulators are dependent on the given feedback gain assumption. This formulation only estimates the underlying objective function, and does not decompose the feedforward and feedback components.

To detect feedback components through IOC, we also formulated IOC on Hamiltonian functions and summarized the results in Appendix B. The formulation represents the estimation of cost function and value function at once; however, the final matrix always become

undetermined (i.e., the number of rows is less than number of columns). Therefore, it is not possible to estimate the cost function. To solve the issue, we proposed to use an iterative algorithm instead. This approach estimated the cost function only in the quadratic form. This approach is not preferred at the end, because it only estimates quadratic objective terms, and is not efficient in comparison to our proposed method in chapter 3.

Decomposing Feedforward-Feedback Components

We decomposed controllers components by assuming specific structures for controllers in simulation for tracking problems. In the implementation, we assumed the same feedforward controller for all structures while the feedback parts were different. We used a proportional-derivative controller for non-optimal feedback and an online optimization for the optimal feedback. To decompose the controller components, we proposed an algorithm that is based on value function. We formulated a constrained optimization problem to estimate the control signal and this optimization problem was used in an algorithm to decide about the structure of controller that generated the motion for nonlinear systems.

Perturbed Human Motion Dataset

As part of this thesis, motion capture data of unperturbed and perturbed for squat motion are recorded. The motion is perturbed by pushes from behind the participant in the upper trunk with random timing and different strength.

Squat Motion Feedforward-Feedback Controller Components Analysis

We studied human squat motion controller by classifying the recorded motions as perturbed and not-perturbed motions. This classification was done based on describing the task by an optimal control problem and regenerating the motion. To regenerate the motion, the objective functions were inferred by applying recovery-based inverse optimal control on the whole window of each trial at once. Then the controller feedforward and feedback terms were extracted through inverse optimal control. The results showed that the participant used the ankle strategy to balance by minimizing center of pressure distance. Also, minimization of torques and power were shown as feedforward components in the motion.

6.2 Future Work

Extension of the Database

- In this study, we only analyzed the motion for one participant. To generalize the results of this work, it is reasonable to investigate for more perturbed motions of more participants. It would be valuable to have data from different age groups in order to compare the behavior of controllers.

Extension of the Human Model

- In this study, we simplified the model to three degrees of freedom. As what is suggested in the literature, a four degrees of freedom with the additional degree in the middle trunk could be more useful to study this motion when the perturbation is included in the model. As we have the information from the push stick, it is possible to model the perturbation as the external force and include it in the dynamical model. Also, to understand the human behavior the model can be extended to include muscle like behavior by muscle torque generators.

- For human squat motion, the deterministic part of the motion is analyzed. One limitation in the implementation of optimal control was on the OptimTraj library which is in MATLAB, and there is no option to connect other dynamical models in C++ to this library. Other libraries for solving optimal control problems such as MUSCOD-II and Bioptim are alternatives to continue the analysis. MUSCOD-II is not an open source library, and Bioptim is. We have summarized the results we got from these libraries in the Appendix C. One challenge we faced with Bioptim is the definition of unilateral constraints to fix feet on the ground. If this issue is fixed, the library is a useful tool for human motion analysis. However, inverse optimal control methods such as the one we used as recovery-matrix IOC limits the use of dynamical models, which has to be analytical to be able estimate objective functions.

Extension of Optimization-based Approaches

- For the human squat dataset, we classified the motion as disturbed or not, and then analyzed the feedforward and feedback components through inverse optimal control. To improve the results for perturbed trials, we suggest to implement nonlinear model predictive control to see whether the same motion can be generated.

- If the human model becomes stochastic and includes more degrees of freedom, it would be interesting to see what kind of optimal control problem can regenerate the motion. The inverse optimal control method has to be adapted for this analysis, respectively.

- If the control policy is detected as the linear combination of feedforward and feedback components, one possible way of generalizing the controller for other conditions is by approximating the controller through neural networks. Methods used on value function approximation can be used to generalize the policy to other initial and final conditions.

- We also formulated state feedback control estimation using Bellman optimality condition and the method is explained in the Appendix D. In order to use this method in practice, we need to extend the work such that non-quadratic terms such as center of pressure and angular momentum could be included in the Hamiltonian function. This might result in a non-closed form solution for the control signal. However, we might be able to extract the feedback part in the control signal automatically.

- The recovery matrix IOC has the advantage of fast analyzing data, analyzing incomplete observations and multiphase motions. This method only includes objective terms which depends only on states and control signal. If the time is a free variable in the task, it cannot be identified. To better identify the underlying objective terms, inclusion of terminal cost function and other objective terms needs to be explored as well.

Potential Applications

- Techniques to analyze the human's controller structures could be useful to analyze the way a person responds to disturbance and then to identify whether somebody is at the falls risk before they actually fall. The results can aid in rehabilitation, sports and mobility assessment. Also this study can support the development of training equipment and assistive devices that work in synchronous manner with human. For example, one specific application could be for falls risk for elderly. For elderly people falling is really bad because they usually break their hip, and it is really hard for them to heal again especially when they have dementia. This is really serious as there is going to be a significant decline in health and increase the death after the fall. Having better tools for being able to understand what people are doing in response to a disturbance could be useful. When elderly people become more frail,

the risk of falls for them is increased. Everyone falls when something unexpected happens. For example, you try to get off from your chair there is something under foot that you do not expect it or the cat walks by or something. For young people when there is that kind of disturbance, they are able to quickly respond and reject the disturbance but for older people that ability to handle disturbances becomes challenging.

References

[1] Nematollah Ab Azar, Aref Shahmansoorian, and Mohsen Davoudi. From Inverse Optimal Control to Inverse Reinforcement Learning: A Historical Review. *Annual Reviews in Control*, 2020.

[2] Pieter Abbeel, Adam Coates, and Andrew Y. Ng. Autonomous Helicopter Aerobatics through Apprenticeship Learning. *International Journal of Robotics Research*, 29(13):1608–1639, 2010.

[3] Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *International Conference on Machine Learning*, ICML '04, pages 1–8, 2004.

[4] Navid Aghasadeghi and Timothy Bretl. Maximum entropy inverse reinforcement learning in continuous state spaces with path integrals. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1561–1566, 2011.

[5] Navid Aghasadeghi and Timothy Bretl. Inverse optimal control for differentially flat systems with application to locomotion modeling. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6018–6025. IEEE, 2014.

[6] Asma Al-Tamimi, Frank L Lewis, and Murad Abu-Khalaf. Discrete-time nonlinear hjb solution using approximate dynamic programming: Convergence proof. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38(4):943–949, 2008.

[7] Scott T Albert and Reza Shadmehr. The neural feedback response to error as a teaching signal for the motor learning system. *Journal of Neuroscience*, 36(17):4832–4845, 2016.

[8] Sebastian Albrecht, Marion Leibold, and Michael Ulbrich. A bilevel optimization approach to obtain optimal cost functions for human arm movements. *Numerical Algebra, Control & Optimization*, 2(1):105, 2012.

[9] Sebastian Albrecht, Karinne Ramirez-Amaro, Federico Ruiz-Ugalde, David Weikersdorfer, Marion Leibold, Michael Ulbrich, and Michael Beetz. Imitating human reaching motions using physically inspired optimization principles. In *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pages 602–607. IEEE, 2011.

[10] Brenna D. Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009.

[11] Saurabh Arora and Prashant Doshi. A Survey of Inverse Reinforcement Learning: Challenges, Methods and Progress. *arXiv:1806.06877 [cs.LG]*, 2018.

[12] Julien Audiffren, Michal Valko, Alessandro Lazaric, and Mohammad Ghavamzadeh. Maximum entropy semi-supervised inverse reinforcement learning. In *International Joint Conference on Artificial Intelligence*, 2015.

[13] Monica Babes-Vroman, Vukosi Marivate, Kaushik Subramanian, and Michael Littman. Apprenticeship learning about multiple intentions. In *International Conference on Machine Learning*, pages 897–904, 2011.

[14] Jan Babič, Erhan Oztop, and Mitsuo Kawato. Human motor adaptation in whole body motion. *Scientific reports*, 6(1):1–12, 2016.

[15] Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

[16] Bastien Berret, Enrico Chiovetto, Francesco Nori, and Thierry Pozzo. Evidence for composite cost functions in arm movement planning: an inverse optimal control approach. *PLoS computational biology*, 7(10):e1002183, 2011.

[17] Bastien Berret and Frédéric Jean. Why don't we move slower? the value of time in the neural control of action. *Journal of neuroscience*, 36(4):1056–1070, 2016.

[18] Dimitri P Bertsekas. Dynamic programming and optimal control 3rd edition, volume ii. *Belmont, MA: Athena Scientific*, 2011.

[19] John T. Betts. Survey of Numerical Methods for Trajectory Optimization. *Journal of Guidance, Control, and Dynamics*, 21(2):193–207, 1998.

[20] Aude Billard, Sylvain Calinon, Rüdiger Dillmann, and Stefan Schaal. Robot Programming by Demonstration. In *Springer Handbook of Robotics*, pages 1371–1394. 2008.

[21] Hans Georg Bock and Karl-Josef Plitt. A multiple shooting algorithm for direct solution of optimal control problems. *IFAC Proceedings Volumes*, 17(2):1603–1608, 1984.

[22] Kenneth Bogert, Jonathan Feng-Shun Lin, Prashant Doshi, and Dana Kulic. Expectation-maximization for inverse reinforcement learning with hidden data. In *International Conference on Autonomous Agents & Multiagent Systems*, pages 1034–1042, 2016.

[23] Vincent Bonnet, Claudia Mazza, Philippe Fraisse, and Aurelio Cappozzo. A least-squares identification algorithm for estimating squat exercise mechanics using a single inertial measurement unit. *Journal of biomechanics*, 45(8):1472–1477, 2012.

[24] Hendrik Börner, Satoshi Endo, and Sandra Hirche. Estimation of involuntary components of human arm impedance in multi-joint movements via feedback jerk isolation. *Frontiers in Neuroscience*, 14:459, 2020.

[25] Abdeslam Boularias and Brahim Chaib-draa. Bootstrapping Apprenticeship Learning. In *Advances in Neural Information Processing Systems*, volume 23, pages 289–297. 2010.

[26] Abdeslam Boularias, Jens Kober, and Jan Peters. Relative Entropy Inverse Reinforcement Learning. In *JMLR Workshop and Conference*, volume 15, pages 182–189. 2011.

[27] Abdeslam Boularias, Oliver Krömer, and Jan Peters. Structured Apprenticeship Learning. In *Machine Learning and Knowledge Discovery in Databases*, Lecture Notes in Computer Science, pages 227–242, 2012.

[28] Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. 2004.

[29] Stephen Boyd et al. *Linear matrix inequalities in system and control theory*, volume 15. Siam, 1994.

[30] Daniel S. Brown and Scott Niekum. Machine Teaching for Inverse Reinforcement Learning: Algorithms and Applications. In *AAAI Conference on Artificial Intelligence*, pages 7749–7758, 2019.

[31] Arunkumar Byravan, Mathew Monfort, Brian Ziebart, Byron Boots, and Dieter Fox. Graph-based inverse optimal control for robot manipulation. In *International Joint Conference on Artificial Intelligence*, pages 1874–1890, 2015.

[32] Pamela Carreno-Medrano, Tatsuki Harada, Jonathan Feng-Shun Lin, Dana Kulic, and Gentiane Venture. Analysis of Affective Human Motion During Functional Task Performance: An Inverse Optimal Control Approach. In *IEEE/RAS International Conference on Humanoid Robots*, pages 461–468, 2019.

[33] Jaedeug Choi and Kee-Eung Kim. Inverse Reinforcement Learning in Partially Observable Environments. *Journal of Machine Learning Research*, 12:691–730, 2011.

[34] Jaedeug Choi and Kee-eung Kim. MAP Inference for Bayesian Inverse Reinforcement Learning. In *Advances in Neural Information Processing Systems*, pages 1989–1997. 2011.

[35] Jaedeug Choi and Kee-eung Kim. Nonparametric Bayesian Inverse Reinforcement Learning for Multiple Reward Functions. In *Advances in Neural Information Processing Systems*, pages 305–313. 2012.

[36] Jaedeug Choi and Kee-Eung Kim. Bayesian nonparametric feature construction for inverse reinforcement learning. In *International Joint Conference on Artificial Intelligence*, pages 1287–1293, 2013.

[37] Debora Clever, Yue Hu, and Katja Mombaur. Humanoid gait generation in complex environments based on template models and optimality principles learned from human beings. *International Journal of Robotics Research*, 37(10):1184–1204, 2018.

[38] Debora Clever and Katja Mombaur. On the Relevance of Common Humanoid Gait Generation Strategies in Human Locomotion: An Inverse Optimal Control Approach. In *Modeling, Simulation and Optimization of Complex Processes HPSC*, pages 27–40, 2017.

[39] Debora Clever, R. Malin Schemschat, Martin L. Felis, and Katja Mombaur. Inverse optimal control based identification of optimality criteria in whole-body human walking on level ground. In *IEEE International Conference on Biomedical Robotics and Biomechatronics*, pages 1192–1199, 2016.

[40] Tyler Cluff and Stephen H Scott. Apparent and actual trajectory control depend on the behavioral context in upper limb motor tasks. *Journal of Neuroscience*, 35(36):12465–12476, 2015.

[41] Adam Coates, Pieter Abbeel, and Andrew Y. Ng. Apprenticeship learning for helicopter control. *Communications of the ACM*, 52(7):97–105, 2009.

[42] Matthew Cockcroft, Shahil Mawjee, Steven James, and Pravesh Ranchod. Learning Options from Demonstration using Skill Segmentation. In *International SAUPEC/RobMech/PRASA Conference*, pages 1–6, 2020.

[43] Frédéric Crevecoeur, Douglas P Munoz, and Stephen H Scott. Dynamic multisensory integration: somatosensory speed trumps visual accuracy during feedback control. *Journal of Neuroscience*, 36(33):8598–8611, 2016.

[44] Paolo De Leva. Adjustments to zatsiorsky-seluyanov's segment inertia parameters. *Journal of biomechanics*, 29(9):1223–1230, 1996.

[45] Scott L Delp, Frank C Anderson, Allison S Arnold, Peter Loan, Ayman Habib, Chand T John, Eran Guendelman, and Darryl G Thelen. Opensim: open-source software to create and analyze dynamic simulations of movement. *IEEE transactions on biomedical engineering*, 54(11):1940–1950, 2007.

[46] Christopher L Dembia, Nicholas A Bianco, Antoine Falisse, Jennifer L Hicks, and Scott L Delp. Opensim moco: Musculoskeletal optimal control. *PLOS Computational Biology*, 16(12):e1008493, 2020.

[47] Michel Desmurget and Scott Grafton. Forward modeling allows feedback control for fast reaching movements. *Trends in cognitive sciences*, 4(11):423–431, 2000.

[48] Jörn Diedrichsen, Reza Shadmehr, and Richard B Ivry. The coordination of movement: optimal feedback control and beyond. *Trends in cognitive sciences*, 14(1):31–39, 2010.

[49] Christos Dimitrakakis and Constantin A. Rothkopf. Bayesian Multitask Inverse Reinforcement Learning. In *Recent Advances in Reinforcement Learning*, volume 7188 of *Lecture Notes in Computer Science*, pages 273–284, 2011.

[50] Andreas Doerr, Nathan Ratliff, Jeannette Bohg, Marc Toussaint, and Stefan Schaal. Direct Loss Minimization Inverse Optimal Control. In *Robotics: Science and Systems*, 2015.

[51] Krishnamurthy Dvijotham and Emanuel Todorov. Inverse Optimal Control with Linearly-Solvable MDPs. In *International Conference on Machine Learning*, pages 335–342, 2010.

[52] Haitham El-Hussieny, A. A. Abouelsoud, Samy FM Assal, and Said M. Megahed. Adaptive learning of human motor behaviors: An evolving inverse optimal control approach. *Engineering Applications of Artificial Intelligence*, 50:115–124, 2016.

[53] Haitham El-Hussieny and Jee-Hwan Ryu. Inverse discounted-based LQR algorithm for learning human movement behaviors. *Applied Intelligence*, 49(4):1489–1501, 2019.

[54] Peter Englert and Marc Toussaint. Inverse kkt–learning cost functions of manipulation tasks from demonstrations. In *Robotics Research*, pages 57–72. Springer, 2018.

[55] Peter Englert and Marc Toussaint. Learning manipulation skills from a single demonstration. *International Journal of Robotics Research*, 37(1):137–154, 2018.

[56] Bin Fang, Shidong Jia, Di Guo, Muhua Xu, Shuhuan Wen, and Fuchun Sun. Survey of imitation learning for robotic manipulation. *International Journal of Intelligent Robotics and Applications*, 3(4):362–369, 2019.

[57] Martin L. Felis. Rbdl: an efficient rigid-body dynamics library using recursive algorithms. *Autonomous Robots*, pages 1–17, 2016.

[58] Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided Cost Learning: Deep Inverse Optimal Control via Policy Optimization. In *International Conference on Machine Learning*, pages 49–58, 2016.

[59] Tamar Flash and Neville Hogan. The coordination of arm movements: an experimentally confirmed mathematical model. *Journal of neuroscience*, 5(7):1688–1703, 1985.

[60] David W Franklin, Etienne Burdet, Keng Peng Tee, Rieko Osu, Chee-Meng Chew, Theodore E Milner, and Mitsuo Kawato. Cns learns stable, accurate, and efficient movements using a simple algorithm. *Journal of neuroscience*, 28(44):11165–11173, 2008.

[61] V. Freire da Silva, A.H. Reali Costa, and Pedro Lima. Inverse reinforcement learning with evaluation. In *IEEE International Conference on Robotics and Automation*, pages 4246–4251, 2006.

[62] Karl Friston. What is optimal about motor control? *Neuron*, 72(3):488–498, 2011.

[63] Justin Fu, Katie Luo, and Sergey Levine. Learning Robust Rewards with Adversarial Inverse Reinforcement Learning. *arXiv:1710.11248 [cs.LG]*, 2017.

[64] Daniel H Grollman and Aude Billard. Donut as I do: Learning from failed demonstrations. In *IEEE International Conference on Robotics and Automation*, pages 3804–3809, 2011.

[65] Kathrin Hatz. *Efficient Numerical Methods for Hierarchical Dynamic Optimization with Application to Cerebral Palsy Gait Modeling*. Dissertation, Ruprecht Karl University of Heidelberg, 2014.

[66] Kathrin Hatz, Johannes P. Schlöder, and Hans Georg Bock. Estimating Parameters in Optimal Control Problems. *SIAM Journal on Scientific Computing*, 34(3):A1707–A1728, 2012.

[67] Peter Henry, Christian Vollmer, Brian Ferris, and Dieter Fox. Learning to navigate through crowded environments. In *IEEE International Conference on Robotics and Automation*, pages 981–986, 2010.

[68] Michael Herman, Tobias Gindele, Jorg Wagner, Felix Schmitt, and Wolfram Burgard. Inverse Reinforcement Learning with Simultaneous Estimation of Rewards and Dynamics. In *International Conference on Artificial Intelligence and Statistics*, volume 51, pages 102–110, 2016.

[69] Hugh Herr and Marko Popovic. Angular momentum in human walking. *Journal of experimental biology*, 211(4):467–481, 2008.

[70] Jonathan Ho and Stefano Ermon. Generative Adversarial Imitation Learning. In *Advances in Neural Information Processing Systems*, volume 29, pages 4565–4573. 2016.

[71] Neville Hogan. The mechanics of multi-joint posture and movement control. *Biological cybernetics*, 52(5):315–331, 1985.

[72] Fay B Horak. Clinical measurement of postural control in adults. *Physical therapy*, 67(12):1881–1885, 1987.

[73] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys*, 50(2):1–35, 2017.

[74] David H Jacobson and David Q Mayne. Differential dynamic programming. 1970.

[75] Jun Jin, Laura Petrich, Masood Dehghan, and Martin Jagersand. A Geometric Perspective on Visual Imitation Learning. *arXiv:2003.02768 [cs.RO]*, 2020.

[76] Jun Jin, Laura Petrich, Masood Dehghan, Zichen Zhang, and Martin Jagersand. Robot eye-hand coordination learning by watching human demonstrations: A task function approximation approach. In *IEEE International Conference on Robotics and Automation*, pages 6624–6630, 2019.

[77] Wanxin Jin, Dana Kulić, Shaoshuai Mou, and Sandra Hirche. Inverse optimal control from incomplete trajectory observations. *The International Journal of Robotics Research*, 40(6-7):848–865, 2021.

[78] Wanxin Jin, Zhaoran Wang, Zhuoran Yang, and Shaoshuai Mou. Pontryagin Differentiable Programming: An End-to-End Learning and Control Framework. *arXiv:1912.12970 [cs.LG]*, 2019.

[79] Zhuo-jun Jin, Hui Qian, Shen-yi Chen, and Miao-liang Zhu. Convergence analysis of an incremental approach to online inverse reinforcement learning. *Journal of Zhejiang University SCIENCE C*, 12(1):17–24, 2011.

[80] Wanxin Jin et al. Inverse optimal control with incomplete observations. *arXiv preprint arXiv:1803.07696*, 2018.

[81] Wanxin Jin et al. Inverse optimal control for multiphase cost functions. *IEEE Transactions on Robotics*, 35(6):1387–1398, 2019.

[82] Miles Johnson, Navid Aghasadeghi, and Timothy Bretl. Inverse optimal control for deterministic continuous-time nonlinear systems. In *52nd IEEE Conference on Decision and Control*, pages 2906–2913. IEEE, 2013.

[83] Vladimir Joukov and Dana Kulic. Gaussian process based model predictive controller for imitation learning. In *IEEE/RAS International Conference on Humanoid Robotics*, pages 850–855, 2017.

[84] Mrinal Kalakrishnan. *Learning Objective Functions for Autonomous Motion Generation*. 2014.

[85] Mrinal Kalakrishnan, Peter Pastor, Ludovic Righetti, and Stefan Schaal. Learning objective functions for manipulation. In *IEEE International Conference on Robotics and Automation*, pages 1331–1336, 2013.

[86] Mrinal Kalakrishnan, Evangelos Theodorou, and Stefan Schaal. *Inverse Reinforcement Learning with PI2*. 2010.

[87] Rudolf Emil Kalman. When is a linear control system optimal? *Journal of Basic Engineering*, 86(1):51–60, 1964.

[88] Shoko Kasuga, Sebastian Telgen, Junichi Ushiba, Daichi Nozaki, and Jörn Diedrichsen. Learning feedback and feedforward control in a mirror-reversed visual environment. *Journal of neurophysiology*, 114(4):2187–2193, 2015.

[89] Hiroaki Kawamoto and Yoshiyuki Sankai. Function analysis method of human's motion control system. In *Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics.'cybernetics evolving to systems, humans, organizations, and their complex interactions'(cat. no. 0*, volume 5, pages 3877–3882. IEEE, 2000.

[90] Matthew Kelly. An introduction to trajectory optimization: How to do your own direct collocation. *SIAM Review*, 59(4):849–904, 2017.

[91] Arezou Keshavarz, Yang Wang, and Stephen Boyd. Imputing a convex objective function. In *IEEE International Symposium on Intelligent Control*, pages 613–619, 2011.

[92] Myunghee Kim and Steven H Collins. Once-per-step control of ankle-foot prosthesis push-off work reduces effort associated with balance during walking. *Journal of neuroengineering and rehabilitation*, 12(1):1–13, 2015.

[93] Kris M. Kitani, Brian D. Ziebart, James Andrew Bagnell, and Martial Hebert. Activity Forecasting. In *European Conference on Computer Vision*, Lecture Notes in Computer Science, pages 201–214, 2012.

[94] Edouard Klein, Bilal Piot, Matthieu Geist, and Olivier Pietquin. A Cascaded Supervised Learning Approach to Inverse Reinforcement Learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Lecture Notes in Computer Science, pages 1–16, 2013.

[95] Jens Kober, J. Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[96] Kai Henning Koch, Katja Mombaur, and Philippe Soueres. Optimization-based walking generation for humanoid robot. *IFAC Proceedings Volumes*, 45(22):498–504, 2012.

[97] Ruth Malin Kopitzsch. *Analysis of Human Push Recovery Motions Based on Optimization*. PhD thesis, 2020.

[98] Henrik Kretzschmar, Markus Spies, Christoph Sprunk, and Wolfram Burgard. Socially compliant mobile robot navigation via inverse reinforcement learning. *International Journal of Robotics Research*, 35(11):1289–1307, 2016.

[99] Matthew Kritz, John Cronin, and Patria Hume. The bodyweight squat: A movement screen for the squat pattern. *Strength & Conditioning Journal*, 31(1):76–85, 2009.

[100] Oliver Kroemer, Scott Niekum, and George Konidaris. A review of robot learning for manipulation: Challenges, representations, and algorithms. *arXiv:1907.03146 [cs.RO]*, 2019.

[101] Dana Kulić, Gentiane Venture, Katsu Yamane, Emel Demircan, Ikuo Mizuuchi, and Katja Mombaur. Anthropomorphic movement analysis and synthesis: A survey of methods and applications. *IEEE Transactions on Robotics*, 32(4):776–795, 2016.

[102] Michail G. Lagoudakis. *Value Function Approximation*, pages 1311–1323. Springer US, Boston, MA, 2017.

[103] Alberto Leardini, Zimi Sawacha, Gabriele Paolini, Stefania Ingrosso, Roberto Nativo, and Maria Grazia Benedetti. A new anatomically based protocol for gait analysis in children. *Gait & posture*, 26(4):560–571, 2007.

[104] Daniel B Leineweber, Irene Bauer, Hans Georg Bock, and Johannes P Schlöder. An efficient multiple shooting based reduced sqp strategy for large-scale dynamic process optimization. part 1: theoretical aspects. *Computers & Chemical Engineering*, 27(2):157–166, 2003.

[105] Sergey Levine and Vladlen Koltun. Continuous inverse optimal control with locally optimal examples. In *International Coference on International Conference on Machine Learning*, ICML'12, pages 475–482, 2012.

[106] Sergey Levine, Zoran Popovic, and Vladlen Koltun. Feature Construction for Inverse Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 23, pages 1342–1350. 2010.

[107] Sergey Levine, Zoran Popovic, and Vladlen Koltun. Nonlinear Inverse Reinforcement Learning with Gaussian Processes. In *Advances in Neural Information Processing Systems*, volume 24, pages 19–27. 2011.

[108] Frank L Lewis, Draguna Vrabie, and Vassilis L Syrmos. *Optimal control*. John Wiley & Sons, 2012.

[109] Kun Li and Joel W. Burdick. Inverse Reinforcement Learning in Large State Spaces via Function Approximation. *arXiv:1707.09394*, 2017.

[110] Kun Li and Joel W. Burdick. Meta Inverse Reinforcement Learning via Maximum Reward Sharing for Human Motion Analysis. *arXiv:1710.03592 [cs.AI]*, 2017.

[111] Weiwei Li and Emanuel Todorov. Iterative linear quadratic regulator design for nonlinear biological movement systems. In *ICINCO (1)*, pages 222–229, 2004.

[112] Weiwei Li, Emanuel Todorov, and Dan Liu. Inverse optimality design for biological movement systems. *IFAC Proceedings Volumes*, 44(1):9662–9667, 2011.

[113] Jonathan Feng-Shun Lin, Pamela Carreno-Medrano, Mahsa Parsapour, Maram Sakr, and Dana Kulić. Objective learning from human demonstrations. *Annual Reviews in Control*, 51:111–129, 2021.

[114] Jonathan Feng-Shun Lin et al. Human motion segmentation using cost weights recovered from inverse optimal control. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 1107–1113. IEEE, 2016.

[115] C. Karen Liu, Aaron Hertzmann, and Zoran Popović. Learning Physics-Based Motion Style with Nonlinear Inverse Optimization. *ACM Transactions on Graphics*, 24(3):1071–1081, 2005.

[116] Yueyue Liu, Zhijun Li, Huaping Liu, and Zhen Kan. Skill transfer learning for autonomous robots and human-robot cooperation: A survey. *Robotics and Autonomous Systems*, pages 103515:1–11, 2020.

[117] Colum D MacKinnon and David A Winter. Control of whole body balance in the frontal plane during human walking. *Journal of biomechanics*, 26(6):633–644, 1993.

[118] Rodrigo S Maeda, Tyler Cluff, Paul L Gribble, and J Andrew Pruszynski. Feedforward and feedback control share an internal model of the arm's dynamics. *Journal of Neuroscience*, 38(49):10505–10514, 2018.

[119] Jim Mainprice and Dmitry Berenson. Learning cost functions for motion planning of human-robot collaborative manipulation tasks from human-human demonstration. In *AAAI Fall Symposium Series*, pages 107–109, 2014.

[120] Toshikazu Matsui, Masayoshi Motegi, and Natsuki Tani. Mathematical model for simulating human squat movements based on sequential optimization. *Mechanical Engineering Journal*, 3(2):15–00377, 2016.

[121] Naser Mehrabi, Reza Sharif Razavian, Borna Ghannadi, and John McPhee. Predictive simulation of reaching moving targets using nonlinear model predictive control. *Frontiers in computational neuroscience*, 10:143, 2017.

[122] Marcel Menner, Peter Worsnop, and Melanie N. Zeilinger. Constrained Inverse Optimal Control With Application to a Human Manipulation Task. *IEEE Transactions on Control Systems Technology*, page (in print), 2019.

[123] Marcel Menner et al. Convex formulations and algebraic solutions for linear quadratic inverse optimal control problems. In *2018 European Control Conference (ECC)*, pages 2107–2112. IEEE, 2018.

[124] Benjamin Michaud, François Bailly, Eve Charbonneau, Amedeo Ceglia, Léa Sanchez, and Mickael Begon. Bioptim, a python framework for musculoskeletal optimal control in biomechanics. *bioRxiv*, 2021.

[125] Benjamin Michaud, François Bailly, Eve Charbonneau, Amedeo Ceglia, L'ea Sanchez, and Mickael Begon. Bioptim, a python framework for musculoskeletal optimal control in biomechanics. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2022.

[126] Benjamin Michaud and Mickaël Begon. Biorbd: A C++, Python and MATLAB library to analyze and simulate the human body biomechanics. 6(57):2562.

[127] Bernard Michini, Mark Cutler, and Jonathan P. How. Scalable reward learning from demonstration. In *IEEE International Conference on Robotics and Automation*, pages 303–308, 2013.

[128] Bernard Michini and Jonathan P. How. Bayesian Nonparametric Inverse Reinforcement Learning. In *Machine Learning and Knowledge Discovery in Databases*, Lecture Notes in Computer Science, pages 148–163, 2012.

[129] Kenneth Miller. Push, pull and shift: Learn the power of perturbations. 2018.

[130] Timothy L. Molloy, Jason J. Ford, and Tristan Perez. Finite-horizon inverse optimal control for discrete-time nonlinear systems. *Automatica*, 87:442–446, 2018.

[131] Katja Mombaur and Debora Clever. Inverse optimal control as a tool to understand human movement. In *Geometric and Numerical Foundations of Movements*, pages 163–186. 2017.

[132] Katja Mombaur and Khai-Long Ho Hoang. How to best support sit to stand transfers of geriatric patients: Motion optimization under external forces for the design of physical assistive devices. *Journal of biomechanics*, 58:131–138, 2017.

[133] Katja Mombaur, Anh Truong, and Jean-Paul Laumond. From human to humanoid locomotion—an inverse optimal control approach. *Autonomous robots*, 28(3):369–383, 2010.

[134] Katja D. Mombaur, Anne-Hélène Olivier, and Armel Crétual. Forward and Inverse Optimal Control of Bipedal Running. In *Modeling, Simulation and Optimization of Bipedal Walking*, pages 165–179, 2013.

[135] Takeshi Mori, Matthew Howard, and Sethu Vijayakumar. Model-free apprenticeship learning for transfer of human impedance behaviour. In *IEEE/RAS International Conference on Humanoid Robots*, pages 239–246, 2011.

[136] P. Moylan and B. Anderson. Nonlinear regulator theory and an inverse optimal control problem. *IEEE Transactions on Automatic Control*, 18(5):460–465, 1973.

[137] Joseph Y Nashed, Frédéric Crevecoeur, and Stephen H Scott. Rapid online selection between multiple motor plans. *Journal of Neuroscience*, 34(5):1769–1780, 2014.

[138] Gergely Neu and Csaba Szepesvári. Training parsers by inverse reinforcement learning. *Machine Learning*, 77(2-3):303–337, 2009.

[139] Gergely Neu and Csaba Szepesvari. Apprenticeship Learning using Inverse Reinforcement Learning and Gradient Methods. *arXiv:1206.5264 [cs.LG]*, 2012.

[140] Andrew Y. Ng and Stuart J. Russell. Algorithms for Inverse Reinforcement Learning. In *International Conference on Machine Learning*, ICML '00, pages 663–670, 2000.

[141] Francesco Nori and Ruggero Frezza. Linear optimal control problems and quadratic cost functions estimation. In *Mediterranean Conference on Control and Automation*, page 1099. Citeseer, 2004.

[142] Ozgur S Oguz, Zhehua Zhou, Stefan Glasauer, and Dirk Wollherr. An inverse optimal control approach to explain human arm reaching control based on multiple internal models. *Scientific reports*, 8(1):1–17, 2018.

[143] Yi-Chung Pai and James Patton. Center of mass velocity-position predictions for balance control. *Journal of biomechanics*, 30(4):347–354, 1997.

[144] Adina M. Panchea. *Inverse Optimal Control for Redundant Systems of Biological Motion*. PhD thesis, Orléans University, 2015.

[145] Adina M. Panchea, Sylvain Miossec, Olivier Buttelli, Philippe Fraisse, Angèle Van Hamme, Marie-Laure Welter, and Nacim Ramdani. Gait analysis using optimality criteria imputed from human data. In *IFAC World Congress*, volume 50, pages 13510–13515, 2017.

[146] Adina M Panchea and Nacim Ramdani. Towards solving inverse optimal control in a bounded-error framework. In *2015 American Control Conference (ACC)*, pages 4910–4915. IEEE, 2015.

[147] Adina M. Panchea, Nacim Ramdani, Vincent Bonnet, and Philippe Fraisse. Human Arm Motion Analysis Based on the Inverse Optimization Approach. In *IEEE International Conference on Biomedical Robotics and Biomechatronics*, pages 1005–1010, 2018.

[148] Marcus G Pandy, Felix E Zajac, Eunsup Sim, and William S Levine. An optimal control model for maximum-height human jumping. *Journal of biomechanics*, 23(12):1185–1198, 1990.

[149] Alessandro Vittorio Papadopoulos, Luca Bascetta, and Gianni Ferretti. Generation of human walking paths. *Autonomous Robots*, 40(1):59–75, 2016.

[150] Daehyung Park, Michael Noseworthy, Rohan Paul, Subhro Roy, and Nicholas Roy. Inferring Task Goals and Constraints using Bayesian Nonparametric Inverse Reinforcement Learning. In *Conference on Robot Learning*, volume 100, pages 1005–1014, 2020.

[151] Jaebum Park, Vladimir M. Zatsiorsky, and Mark L. Latash. Finger Coordination Under Artificial Changes in Finger Strength Feedback: A Study Using Analytical Inverse Optimization. *Journal of Motor Behavior*, 2011.

[152] Taesung Park and Sergey Levine. Inverse optimal control for humanoid locomotion. In *Robotics science and systems workshop on inverse optimal control and robotic learning from demonstration*, pages 4887–4892, 2013.

[153] Mahsa Parsapour and Dana Kulić. Recovery-matrix inverse optimal control for deterministic feedforward-feedback controllers. In *2021 American control conference (ACC)*, pages 4765–4770. IEEE, 2021.

[154] M Cody Priess et al. Solutions to the inverse lqr problem with application to biological systems analysis. *IEEE Transactions on control systems technology*, 23(2):770–777, 2014.

[155] Anne-Sophie Puydupin-Jamin, Miles Johnson, and Timothy Bretl. A convex approach to inverse optimal control and its application to modeling human locomotion. In *IEEE International Conference on Robotics and Automation*, pages 531–536, 2012.

[156] Qifeng Qiao and Peter A. Beling. Inverse reinforcement learning with Gaussian process. In *American Control Conference*, pages 113–118, 2011.

[157] Deepak Ramachandran. Bayesian Inverse Reinforcement Learning. In *International Joint Conference on Artificial Intelligence*, pages 2586–2591, 2007.

[158] Nathan Ratliff, David Bradley, J. Andrew Bagnell, and Joel Chestnutt. Boosting structured prediction for imitation learning. In *International Conference on Neural Information Processing Systems*, NIPS'06, pages 1153–1160, 2006.

[159] Nathan D. Ratliff, J. Andrew Bagnell, and Martin A. Zinkevich. Maximum margin planning. In *International Conference on Machine Learning*, ICML '06, pages 729–736, 2006.

[160] Harish Ravichandar, Athanasios S. Polydoros, Sonia Chernova, and Aude Billard. Recent advances in robot learning from demonstration. *Annual Review of Control, Robotics, and Autonomous Systems*, 3, 2020.

[161] John R. Rebula, Stefan Schaal, James Finley, and Ludovic Righetti. A Robustness Analysis of Inverse Optimal Control of Bipedal Walking. *IEEE Robotics and Automation Letters*, 4(4):4531–4538, 2019.

[162] Tummalapalli Sudhamsh Reddy, Vamsikrishna Gopikrishna, Gergely Zaruba, and Manfred Huber. Inverse reinforcement learning for decentralized non-cooperative multiagent systems. In *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 1930–1935, 2012.

[163] Constantin A. Rothkopf and Christos Dimitrakakis. Preference Elicitation and Inverse Reinforcement Learning. In *Machine Learning and Knowledge Discovery in Databases*, Lecture Notes in Computer Science, pages 34–48, 2011.

[164] Stefan Schaal. Learning from demonstration. In *Advances in Neural Information Processing Systems*, volume 9, pages 1040–1046, 1997.

[165] Pierre Sermanet, Kelvin Xu, and Sergey Levine. Unsupervised perceptual rewards for imitation learning. *arXiv:1612.06699 [cs.CV]*, 2016.

[166] K. Shiarlis, J. Messias, and S. A. Whiteson. *Inverse Reinforcement Learning from Failure*. 2016.

[167] David Silver, J. Andrew Bagnell, and Anthony Stentz. Learning from Demonstration for Autonomous Navigation in Complex Unstructured Terrain. *International Journal of Robotics Research*, 29(12):1565–1592, 2010.

[168] Lizeth H Sloot, Matthew Millard, Christian Werner, and Katja Mombaur. Slow but steady: Similar sit-to-stand balance at seat-off in older vs. younger adults. *Frontiers in sports and active living*, 2, 2020.

[169] Kevin Stein and Katja Mombaur. A quantitative comparison of slackline balancing capabilities of experts and beginners. *Frontiers in Sports and Active Living*, 4, 2022.

[170] Umar Syed, Michael Bowling, and Robert E. Schapire. Apprenticeship learning using linear programming. In *ACM International Conference on Machine Learning*, ICML '08, pages 1032–1039, 2008.

[171] Umar Syed and Robert E. Schapire. A game-theoretic approach to apprenticeship learning. In *ACM International Conference on Neural Information Processing Systems*, NIPS'07, pages 1449–1456, 2007.

[172] N. Sylla, V. Bonnet, G. Venture, N. Armande, and P. Fraisse. Human arm optimal motion analysis in industrial screwing task. In *IEEE EMBS/RAS International Conference on Biomedical Robotics and Biomechatronics*, pages 964–969, 2014.

[173] Yuval Tassa, Tom Erez, and Emanuel Todorov. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4906–4913. IEEE, 2012.

[174] Alexander V. Terekhov, Yakov B. Pesin, Xun Niu, Mark L. Latash, and Vladimir M. Zatsiorsky. An analytical approach to the problem of inverse optimization with additive objective functions: An application to human prehension. *Journal of Mathematical Biology*, 61(3):423–453, 2010.

[175] Alexander V. Terekhov and Vladimir M. Zatsiorsky. Analytical and numerical analysis of inverse optimization problems: Conditions of uniqueness and computational methods. *Biological Cybernetics*, 104(1-2):75–93, 2011.

[176] Evangelos Theodorou, Jonas Buchli, and Stefan Schaal. A Generalized Path Integral Control Approach to Reinforcement Learning. *Journal of Machine Learning Research*, 11:3137–3181, 2010.

[177] Evangelos Theodorou et al. Stochastic differential dynamic programming. In *Proceedings of the 2010 American Control Conference*, pages 1125–1132. IEEE, 2010.

[178] Emanuel Todorov. Optimality principles in sensorimotor control. *Nature neuroscience*, 7(9):907–915, 2004.

[179] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[180] Emanuel Todorov and Michael I Jordan. Optimal feedback control as a theory of motor coordination. *Nature neuroscience*, 5(11):1226–1235, 2002.

[181] Emanuel Todorov and Weiwei Li. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 300–306. IEEE, 2005.

[182] Midhun P Unni, Aniruddha Sinha, Kingshuk Chakravarty, Debatri Chatterjee, and Abhijit Das. Neuromechanical cost functionals governing motor control for early screening of motor disorders. *Frontiers in bioengineering and biotechnology*, 5:78, 2017.

[183] Yoji Uno, Mitsuo Kawato, and Rika Suzuki. Formation and control of optimal trajectory in human multijoint arm movement. *Biological cybernetics*, 61(2):89–101, 1989.

[184] Kyriakos G Vamvoudakis and Frank L Lewis. Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem. *Automatica*, 46(5):878–888, 2010.

[185] Kazunori WADA and Toshikazu MATSUI. Optimal control model for reproducing human sitting movements on a chair and its effectiveness. *Journal of Biomechanical Science and engineering*, 8(2):164–179, 2013.

[186] Kevin Westermann, Jonathan Feng-Shun Lin, and Dana Kulić. Inverse optimal control with time-varying objectives: Application to human jumping movement analysis. *Scientific Reports*, 10(1):11174, 2020.

[187] David A Winter. *Biomechanics and motor control of human movement*. John Wiley & Sons, 2009.

[188] Yue Wu, Hui Wang, Biaobiao Zhang, and K-L Du. Using radial basis function networks for function approximation and classification. *International Scholarly Research Notices*, 2012, 2012.

[189] Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. Maximum Entropy Deep Inverse Reinforcement Learning. *arXiv:1507.04888 [cs]*, 2015.

[190] Chen Xia and Abdelkader El Kamel. Neural inverse reinforcement learning in autonomous navigation. *Robotics and Autonomous Systems*, 84:1–14, 2016.

[191] Xin Xin, Kanjian Zhang, and Haikun Wei. Linear strong structural controllability for an n-link inverted pendulum in a cart. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 1204–1209. IEEE, 2018.

[192] Sang-Hoon Yeo, David W Franklin, and Daniel M Wolpert. When optimal feedback control is not enough: Feedforward strategies are required for optimal control with active sensing. *PLoS computational biology*, 12(12), 2016.

[193] Hang Yin, Patrícia Alves-Oliveira, Francisco S. Melo, Aude Billard, and Ana Paiva. Synthesizing robotic handwriting motion by learning from human demonstrations. In *International Joint Conference on Artificial Intelligence*, number CONF, pages 3530–3537, 2016.

[194] Han Zhang, Yibei Li, and Xiaoming Hu. Inverse Optimal Control for Finite-Horizon Discrete-time Linear Quadratic Regulator Under Noisy Output. In *IEEE Conference on Decision and Control*, pages 6663–6668, 2019.

[195] Han Zhang, Jack Umenberger, and Xiaoming Hu. Inverse Quadratic Optimal Control for Discrete-Time Linear Systems. *arXiv:1810.12590 [math]*, 2018.

[196] Han Zhang et al. Inverse optimal control for discrete-time finite-horizon linear quadratic regulators. *Automatica*, 110:108593, 2019.

[197] Jiangchuan Zheng, Siyuan Liu, and Lionel M Ni. Robust Bayesian Inverse Reinforcement Learning with Sparse Behavior Noise. *AAAI Conference on Artificial Intelligence*, pages 2198–2205, 2014.

[198] Shao Zhifei and Meng Joo Er. A survey of inverse reinforcement learning techniques. *International Journal of Intelligent Computing and Cybernetics*, 5(3):293–311, 2012.

[199] Brian D Ziebart, Andrew Maas, J Andrew Bagnell, and Anind K Dey. Maximum Entropy Inverse Reinforcement Learning. In *National Conference on Artificial Intelligence*, page 6, 2008.

APPENDICES

Appendix A

Human Model Parameters

The participant is with the total mass $m_T = 63$ kg, and total height of $l_T = 1.57$ m. The summary of each segment's properties is shown in Table A.1. For link i, m_i is its mass, l_i is the length, l_{ci} is the distance from its center of mass (COM) to joint i, and I_i is the moment of inertia around its COM.

Table A.1: Parameters for the model. m_T is the total mass, and l_T is the total height of the participant.

Segment Name	Shank $i = 1$	Thigh $i = 2$	Upper Body (Trunk + Head+ Arms) $i = 3$
$l_i[m]$	$0.2492 l_T$	$0.27 l_T$	$0.4205 l_T$
$m_i[kg]$	$0.0962 m_T$	$0.2956 m_T$	$0.5823 m_T$
$l_{ci}[m]$	$0.093 l_1$	$0.162 l_2$	(A.2)
$I_i[kg.m^2]$	$0.0962 m_T (0.093 l_1)^2$	$0.2956 m_T (0.162 l_2)^2$	(A.3)
CoM $[m]$	$0.4416 l_1$	$0.3612 l_2$	(A.1)

$$
\begin{aligned}
CoM^{new}_{head} &= l_{trunk} + l_{head} - CoM_{head} \\
CoM^{new}_{trunk} &= l_{trunk} - CoM_{trunk} \\
CoM^{new}_{upperArm} &= l_{trunk} - CoM_{upperArm} cos(\alpha) \\
CoM^{new}_{forearm} &= l_{trunk} - (l_{upperArm} + CoM_{forearm}) cos(\alpha) \\
CoM^{new}_{hand} &= l_{trunk} - (l_{upperArm} + l_{forearm} + CoM_{hand}) cos(\alpha)
\end{aligned}
\tag{A.1}
$$

$$
l_{c3} = \frac{M_{head} CoM^{new}_{head} + M_{trunk} CoM^{new}_{trunk} + 2(M_{upperArm} CoM^{new}_{upperArm} + M_{forearm} CoM^{new}_{forearm} + M_{hand} CoM^{new}_{hand})}{M_{head} + M_{trunk} + 2(M_{upperArm} + M_{forearm} + M_{hand})} \tag{A.2}
$$

$$
\begin{aligned}
I^{new}_{trunk} &= I_{trunk} + M_{trunk} d^2_{trunk} \\
I^{new}_{head} &= I_{head} + M_{head} d^2_{head} \\
I^{new}_{upperArm} &= I_{upperArm} + M_{upperArm} d^2_{upperArm} \\
I^{new}_{forearm} &= I_{forearm} + M_{forearm} d^2_{forearm} \\
I^{new}_{hand} &= I_{hand} + M_{hand} d^2_{hand} \\
I_3 &= I^{new}_{trunk} + I^{new}_{head} + 2(I^{new}_{upperArm} + I^{new}_{forearm} + I^{new}_{hand})
\end{aligned}
\tag{A.3}
$$

Appendix B

Inverse Optimal Control and Hamiltonian Function

This appendix summarizes the methods we develop on inverse optimal control to detect the feedback component using the Hamiltonian function.

B.1 Inverse Optimal Control using Hamiltonian Function

Consider a class of discrete-time systems described by deterministic nonlinear dynamics in the affine state space difference equation form

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k) + \boldsymbol{G}(\boldsymbol{x}_k)\boldsymbol{u}_k, \tag{B.1}$$

where $\boldsymbol{f}(\boldsymbol{x}_k)$ and $\boldsymbol{G}(\boldsymbol{x}_k)$ are known nonlinearities with $\boldsymbol{x}_k \in \mathbb{R}^{n_x}$ and $\boldsymbol{u}_k \in \mathbb{R}^{m_u}$ as the state and input vectors at time instance $k \in \mathbb{Z}$, respectively. The performance index to be minimized is

$$J(\boldsymbol{x}_0) = \sum_{k=0}^{N-1} \boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}_k, \boldsymbol{u}_k), \tag{B.2}$$

where N is the number of time steps, and $\boldsymbol{w}^T\boldsymbol{\Phi}(.,.)$ is the running cost function. The cost function is represented as the linear combination of cost terms.

$V(\boldsymbol{x}_0)$ is the accumulated cost if the system is initialized at time step zero in state $\boldsymbol{x}_0$ with $\boldsymbol{u}_{0:N-1} = \{u_0, ..., u_{N-1}\}$. The value function (cost-to-go) for the kth stage can be rewritten as

$$V_k(\boldsymbol{x}_k) = \boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}_k, \boldsymbol{u}_k) + \sum_{n=k+1}^{N-1} \boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}_n, \boldsymbol{u}_n)$$

which is known as discrete-time Bellman equation:

$$V_k(\boldsymbol{x}_k) = \boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}_k, \boldsymbol{u}_k) + V_{k+1}(\boldsymbol{x}_{k+1}) \tag{B.3}$$

where at the Nth stage, we have $V_N(\boldsymbol{x}_N) = \frac{1}{2}\boldsymbol{x}_N^T \boldsymbol{Q}_N \boldsymbol{x}_N$. The minimization of this equation is just the discrete-time Hamilton-Jacobi-Bellman (HJB) equation.

$$V_k^*(\boldsymbol{x}_k) = \min_{\boldsymbol{u}_k}(\boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}_k, \boldsymbol{u}_k) + V_k^*(\boldsymbol{x}_{k+1})) \tag{B.4}$$

Equation (B.3) can be used to recursively solve for the optimal control starting from the end of the horizon backward in time.

The Bellman optimality condition states that

$$V(\boldsymbol{x}_k^*) = \boldsymbol{w}^T \boldsymbol{\Phi}(\boldsymbol{x}_k^*, \boldsymbol{u}_k^*) + V(\boldsymbol{x}_{k+1}^*) \tag{B.5}$$

where $V(\boldsymbol{x}_t^*)$ evaluated at state $\boldsymbol{x}_t^*$ is unknown. By differentiating both sides of the Bellman equation with respect to $\boldsymbol{x}_k^*$ and $\boldsymbol{u}_k^*$, we have

$$\begin{aligned} \frac{\partial V(\boldsymbol{x}_k^*)}{\partial \boldsymbol{x}_k^*} &= \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}_k^*}^T \boldsymbol{w} + \frac{\partial V(\boldsymbol{x}_{k+1}^*)}{\partial \boldsymbol{x}_{k+1}^*} \frac{\partial \boldsymbol{x}_{k+1}^*}{\partial \boldsymbol{x}_k^*} \\ 0 &= \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{u}_k^*}^T \boldsymbol{w} + \frac{\partial V(\boldsymbol{x}_{k+1}^*)}{\partial \boldsymbol{x}_{k+1}^*} \frac{\partial \boldsymbol{x}_{k+1}^*}{\partial \boldsymbol{u}_k^*} \end{aligned} \tag{B.6}$$

denoting $\frac{\partial V(\boldsymbol{x}_k^*)}{\partial \boldsymbol{x}_k^*} = \boldsymbol{\lambda}_k^* \in \mathbb{R}^n$

$$\begin{aligned} \boldsymbol{\lambda}_k^* &= \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{x}_k^*}^T \boldsymbol{w} + \frac{\partial \boldsymbol{f}^T}{\partial \boldsymbol{x}_k^*} \boldsymbol{\lambda}_{k+1}^* \\ 0 &= \frac{\partial \boldsymbol{\phi}}{\partial \boldsymbol{u}_k^*}^T \boldsymbol{w} + \frac{\partial \boldsymbol{f}^T}{\partial \boldsymbol{u}_k^*} \boldsymbol{\lambda}_{k+1}^* \end{aligned} \tag{B.7}$$

assuming $\frac{\partial \boldsymbol{x}_{k+1}^*}{\partial \boldsymbol{x}_k^*}$ is invertible, we can eliminate $\boldsymbol{\lambda}_{k+1}^*$ from above and write the equations for the entire horizon. By stacking the above equation for each time step we have

$$\boldsymbol{\Phi}_u \boldsymbol{w} + \boldsymbol{F}_u \boldsymbol{F}_x^{-1}(\boldsymbol{\lambda} - \boldsymbol{\Phi}_x \boldsymbol{w}) = 0 \tag{B.8}$$

where

$$\boldsymbol{F}_x = \begin{bmatrix} \frac{\partial f^T}{\partial \boldsymbol{x}_1^*} & & \\ & \ddots & \\ & & \frac{\partial f^T}{\partial \boldsymbol{x}_{N-1}^*} \end{bmatrix} \in \mathbb{R}^{n_x(N-1) \times n_x(N-1)} \tag{B.9}$$

$$\boldsymbol{F}_u = \begin{bmatrix} \frac{\partial f^T}{\partial \boldsymbol{u}_1^*} & & \\ & \ddots & \\ & & \frac{\partial f^T}{\partial \boldsymbol{u}_{N-1}^*} \end{bmatrix} \in \mathbb{R}^{m_u(N-1) \times n_x(N-1)} \tag{B.10}$$

$$\boldsymbol{\Phi}_x = \begin{bmatrix} \frac{\partial \Phi}{\partial \boldsymbol{x}_1^*} & \cdots & \frac{\partial \Phi}{\partial \boldsymbol{x}_{N-1}^*} \end{bmatrix}^T \in \mathbb{R}^{n_x(N-1) \times r} \tag{B.11}$$

$$\boldsymbol{\Phi}_u = \begin{bmatrix} \frac{\partial \Phi}{\partial \boldsymbol{u}_1^*} & \cdots & \frac{\partial \Phi}{\partial \boldsymbol{u}_{N-1}^*} \end{bmatrix}^T \in \mathbb{R}^{m_u(N-1) \times r} \tag{B.12}$$

$$\boldsymbol{\lambda} = \begin{bmatrix} \boldsymbol{\lambda}_1 & \ldots & \boldsymbol{\lambda}_{N-1} \end{bmatrix}^T \in \mathbb{R}^{n_x(N-1) \times 1} \tag{B.13}$$

simplifying the equation we have

$$\begin{bmatrix} \boldsymbol{\Phi}_u - \boldsymbol{F}_u \boldsymbol{F}_x^{-1} \boldsymbol{\Phi}_x & \boldsymbol{F}_u \boldsymbol{F}_x^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{\lambda} \end{bmatrix} = 0 \tag{B.14}$$

To find $\boldsymbol{w}$ and $\boldsymbol{\lambda}$, we can minimize the 2-norm of its residual

$$\begin{aligned} &\min_{\boldsymbol{w}, \boldsymbol{\lambda}} || \begin{bmatrix} \boldsymbol{\Phi}_u - \boldsymbol{F}_u \boldsymbol{F}_x^{-1} \boldsymbol{\Phi}_x & \boldsymbol{F}_u \boldsymbol{F}_x^{-1} \end{bmatrix} \begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{\lambda} \end{bmatrix} ||_2^2 \\ &\text{s.t.} \quad \sum_{i=1}^{r} \boldsymbol{w} = 1. \end{aligned} \tag{B.15}$$

B.2 Iterative Algorithm using Hamiltonian Function

The previous section represents the formulation that estimates both the cost function and value function at once; however, the final matrix always become undetermined. Therefore, it is not possible to estimate the cost function. To solve the issue, we propose to use an iterative algorithm instead.

Let us define the performance index to be minimized in the quadratic form

$$J(\boldsymbol{x}_0) = \frac{1}{2}\boldsymbol{x}_N^T \boldsymbol{Q}_N \boldsymbol{x}_N + \sum_{k=0}^{N-1}(\frac{1}{2}\boldsymbol{x}_k^T \boldsymbol{Q}\boldsymbol{x}_k + \frac{1}{2}\boldsymbol{u}_k^T \boldsymbol{R}\boldsymbol{u}_k), \tag{B.16}$$

$\boldsymbol{x}_N$ is the final state. $\boldsymbol{Q}$ and $\boldsymbol{Q}_N$ are symmetric positive semi-definite ($\mathbb{S}_+^n$) state penalty and terminal state penalty matrices, and $\boldsymbol{R}$ is a symmetric positive definite ($\mathbb{S}_{++}^m$) input-penalty matrix. In this problem, we only focus on recovering the weighting related to the state penalty. Let us represent the penalty on the states in the running cost function with $\boldsymbol{q}(\boldsymbol{x}_k) = \frac{1}{2}\boldsymbol{x}_k^T \boldsymbol{Q}\boldsymbol{x}_k$. Therefore the Hamiltonian function can be states as

$$\mathcal{H}(\boldsymbol{x}_k, \boldsymbol{u}_k, \Delta V(\boldsymbol{x}_k)) = \boldsymbol{q}(\boldsymbol{x}_k) + \frac{1}{2}\boldsymbol{u}_k^T \boldsymbol{R}\boldsymbol{u}_k + V_k(\boldsymbol{x}_{k+1}) - V_k(\boldsymbol{x}_k),$$

The optimal value of the cost-to-go function, $V^*(\boldsymbol{x}_k)$ can be estimated by the function approximation method. Let us define it as:

$$V^*(\boldsymbol{x}_k) = \sum_{i=1}^{m} w_i \boldsymbol{\phi}_i(\boldsymbol{x}_k), \tag{B.17}$$

where

$$\boldsymbol{\phi}_i(\boldsymbol{x}_k) = exp(-\frac{||\boldsymbol{x}_k - c_i||^2}{r_i^2}), \tag{B.18}$$

is the i^th Gaussian radial basis function centered at c_i and of width r_i. w_i is the associated weight of $\boldsymbol{\phi}_i(\boldsymbol{x}_k)$.

In practice, the approximation of the value function can have some approximation error such that as the number of basis function increases, the error converges to zero. The error of the value function approximation leads to some residual error e_h on the Hamiltonian function. [184] showed that this residual error is bounded on a compact set under the Lipschitz assumption on the dynamics. To estimate both $\boldsymbol{w}$ (weights of the value function)

and $\boldsymbol{q}(\boldsymbol{x}_k)$, we can formulate the optimization problem based on minimizing the 2-norm of the Hamiltonian function $\mathcal{H}_k(\boldsymbol{w})$ and the condition of having non-negative $\boldsymbol{q}(\boldsymbol{x}_k)$.

$$\begin{aligned} &\min_{\boldsymbol{w}} ||\mathcal{H}_k(\boldsymbol{w})||_2^2 \\ &\text{s.t.} \quad \boldsymbol{q}(\boldsymbol{x}_k) \geq 0, \;\; k = 1, ..., N-1. \end{aligned} \tag{B.19}$$

where

$$\mathcal{H}_k(\boldsymbol{w}) = \boldsymbol{w}^T(\boldsymbol{\Phi}(\boldsymbol{x}_{k+1}) - \boldsymbol{\Phi}(\boldsymbol{x}_k)) + \boldsymbol{q}(\boldsymbol{x}_k) + \frac{1}{2}\boldsymbol{u}_k^T \boldsymbol{R}\boldsymbol{u}_k, \tag{B.20}$$

The structure of this problem is a constrained Quadratic Programming problem. As $\boldsymbol{q}(\boldsymbol{x}_k)$ is unknown and appears in the objective function, we need to have an algorithm that for each pair of $(\boldsymbol{x}_k, \boldsymbol{u}_k)$ the problem is solved and update $\boldsymbol{q}(\boldsymbol{x}_k)$ based on that for the next iteration. Algorithm 6 is the preliminary approach to implement this optimization problem. Now by considering $\boldsymbol{A}_k = \boldsymbol{\Phi}(\boldsymbol{x}_{k+1}) - \boldsymbol{\Phi}(\boldsymbol{x}_k)$, $b_k = q(\boldsymbol{x}_k) + \frac{1}{2}\boldsymbol{u}_k^T \boldsymbol{R}\boldsymbol{u}_k$, and $c_k = \frac{1}{2}\boldsymbol{u}_k^T \boldsymbol{R}\boldsymbol{u}_k$ the general standard form of the problem becomes:

$$\begin{aligned} &\min_{\boldsymbol{w}} \boldsymbol{w}^T \boldsymbol{A}_k \boldsymbol{A}_k^T \boldsymbol{w} + 2b_k \boldsymbol{A}_k^T \boldsymbol{w} \\ &\text{s.t.} \quad \boldsymbol{w}^T \boldsymbol{A}_k - c_k \leq 0, \;\; k = 1, ..., N-1. \end{aligned} \tag{B.21}$$

at each iteration this optimization problem can be solved using the active-set method which handles inequality constraints.

The output of equation (B.21), is a constant vector over time and states. If $\boldsymbol{q}(\boldsymbol{x}_k)$ was known, then we could simply stack all $\boldsymbol{A}_k$, b_k, and c_k as $\boldsymbol{A}$, b, c and solve the following optimization problem once

$$\begin{aligned} &\min_{\boldsymbol{w}} \boldsymbol{w}^T \boldsymbol{H}\boldsymbol{w} + \boldsymbol{f}\boldsymbol{w} \\ &\text{s.t.} \quad \boldsymbol{I} \leq 0, \;\; k = 1, ..., N-1. \end{aligned} \tag{B.22}$$

where $\boldsymbol{H} = \begin{pmatrix} \boldsymbol{A}_1 \boldsymbol{A}_1^T \\ \vdots \\ \boldsymbol{A}_{N-1} \boldsymbol{A}_{N-1}^T \end{pmatrix}$, $\boldsymbol{f} = \begin{pmatrix} 2b_1 \boldsymbol{A}_1^T \\ \vdots \\ 2b_{N-1} \boldsymbol{A}_{N-1}^T \end{pmatrix}$, and $\boldsymbol{I} = \begin{pmatrix} \boldsymbol{w}^T \boldsymbol{A}_1 - c_1 \\ \vdots \\ \boldsymbol{w}^T \boldsymbol{A}_{N-1} - c_{N-1} \end{pmatrix}$. The results for three different systems are shown in Fig. B.1.

However, the challenge here is that $\boldsymbol{q}(\boldsymbol{x}_k)$ is not known, and it appears both in the objective function and the inequality constraint. One possible approach to update $\boldsymbol{q}(\boldsymbol{x}_k)$ is to update the $\boldsymbol{Q}$ as $\boldsymbol{q}(\boldsymbol{x}_k)$ has a structure.

$$\boldsymbol{Q}_{i,j}^{new} = \boldsymbol{Q}_{i,j}^{old} + \eta P_{i,j} \tag{B.23}$$

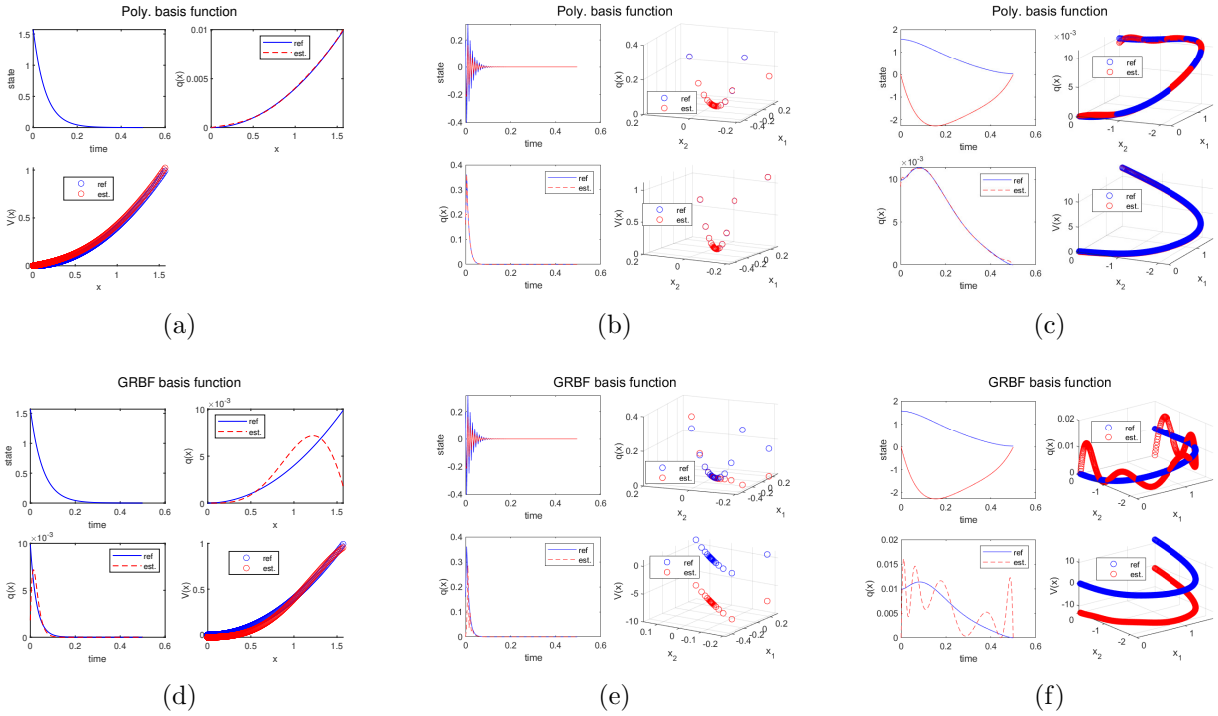


Figure B.1: Estimation of $\boldsymbol{w}$ when $q(\boldsymbol{x}_k)$ is known. (a) and (d) linear system with one state, (b) and (e) linear system with two states, (c) and (f) inverted pendulum.

where η is a nonnegative relaxation factor which is between zero and one, and $\boldsymbol{P}_k$ is the search direction. The search direction can be defined on minimizing the squared residual error on the Hamiltonian function. If we assume that $\mathcal{H}_k(\boldsymbol{w}) = e_k$ and $E = \frac{1}{2}e^T e$ then the search direction is

$$P_{i,j} = -\frac{\partial E}{\partial Q_{ij}} = -\begin{bmatrix} e_1 & \cdots & e_N \end{bmatrix} \begin{bmatrix} \frac{\partial e_1}{\partial Q_{ij}} \\ \vdots \\ \frac{\partial e_N}{\partial Q_{ij}} \end{bmatrix} = -\begin{bmatrix} e_1 & \cdots & e_N \end{bmatrix} \begin{bmatrix} \frac{1}{2}\boldsymbol{x}_{i,1}\boldsymbol{x}_{j,1} \\ \vdots \\ \frac{1}{2}\boldsymbol{x}_{i,N}\boldsymbol{x}_{j,N} \end{bmatrix} \tag{B.24}$$

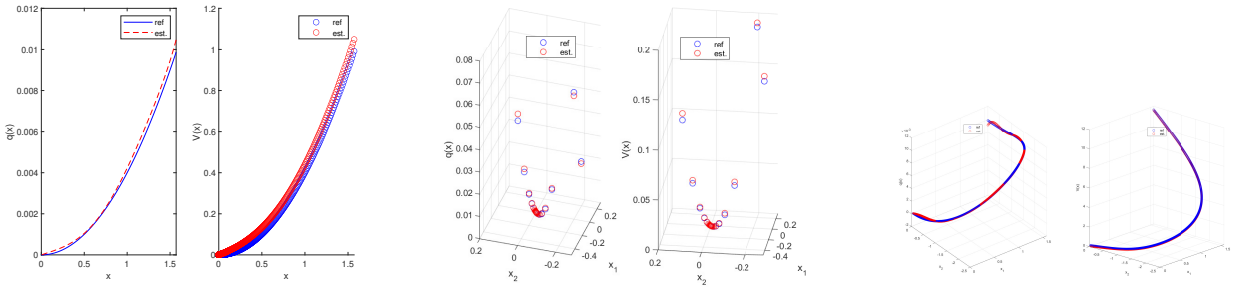
where e_i is calculated using $Q_{i,j}^{old}$. Algorithm. 6 shows the step of implementing the idea for estimating both $\boldsymbol{w}$ and $q(\boldsymbol{x}_k)$ based on minimization of Hamiltonian function. This algorithm for systems with more than one state is implemented as a bilevel optimization problem. In another word, the update explained above is just a search in the feasible space to find the closest weights that give the estimated trajectory similar to the reference trajectory. The initialization should start in such a way that all the space is covered.

Algorithm 6 Estimation of $\boldsymbol{w}$ and $q(\boldsymbol{x}_k)$ based on minimization of Hamiltonian function (constrained QP optimization)

Input: Optimal state and control sequence $\boldsymbol{u}^*_{1:N-1}$ and $\boldsymbol{x}^*_{1:N}$, and system dynamics (D.1), with $\boldsymbol{R}$

Initialize $\bar{\boldsymbol{Q}} \leftarrow \alpha$
while $\bar{\boldsymbol{Q}} > 0$ and $||\boldsymbol{x}^*_{1:N} - \hat{\boldsymbol{x}}_{1:N}||_2 < \gamma$ **do**
  $\min_{\boldsymbol{w}} \boldsymbol{w}^T \boldsymbol{H} \boldsymbol{w} + \boldsymbol{f}\boldsymbol{w}$,
  s.t. $\boldsymbol{I} \leq 0, \quad k = 1, ..., N-1.$
  $\bar{\boldsymbol{Q}}_{ij} \leftarrow \bar{\boldsymbol{Q}}_{ij} - \eta \frac{\partial E}{\partial \boldsymbol{Q}_{ij}}$
end while
$q(\boldsymbol{x}_k) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_k) - \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_{k+1}) - \frac{1}{2} u_k^T R u_k$
$\boldsymbol{Q} \leftarrow \bar{\boldsymbol{Q}}$
return $q(\boldsymbol{x}_k)$ and $\boldsymbol{Q}$

Now by implementing Algorithm 6, results are obtained in Fig. B.2.



(a) (b) (c)

Figure B.2: Estimation of $\boldsymbol{w}$ when $q(\boldsymbol{x}_k)$ is unknown with polynomial basis functions. (a) linear system with one state, (b) linear system with two states, (c) inverted pendulum.

Appendix C

Human Squat Motion Modeling by Libraries

This appendix summarizes the human squat modelling using the Rigid Body Dynamics Library [57] and Bioptim [125].

C.0.1 Mechanical Model

The human model is modeled as a four DOF system. The dynamics of the model is described with the equation of motion:

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) \tag{C.1}$$

where the joint angles, joint velocities, and torques are $\boldsymbol{q} \in \mathbb{R}^4$, $\dot{\boldsymbol{q}} \in \mathbb{R}^4$, and $\boldsymbol{\tau} \in \mathbb{R}^4$ respectively. $\boldsymbol{M}(\boldsymbol{q})$ is the inertia matrix, $\boldsymbol{C}$ is the coriolis and centrifugal term, and $\boldsymbol{g}$ is the gravity vector. Dynamics of the system is computed using the Rigid Body Dynamics Library [57], and the model is defined in a .lua model. Another model named .bioMod model is also generated by converting the .lua model to .bioMod model using the Biorbd package [126]. The dynamic parameters are shown in Table. C.1. The total height (L_{total}) and the total mass (m_{total}) of the human model is updated based on the measurements of the female or male models reported on [44]. L_i is the length of each segment, and m_i is the mass. The upper body segment is a combination of head, trunk, upper arms, forearms, and hands. L_{ci} takes the longitudinal center of mass (CoM) of each body segment.

Table C.1: Dynamics parameters of the human body from [44]

parameter	Shank	Thigh	Pelvis	Upper Body
$L_i(m)$	$0.25L_{total}$	$0.21L_{total}$	$0.10L_{total}$	$0.34L_{total}$
$L_{ci}(m)$	$0.14L_1$	$0.13L_2$	$0.05L_3$	$0.06L_4$
$m_i(kg)$	$0.10m_{total}$	$0.30m_{total}$	$0.12m_{total}$	$0.37m_{total}$

C.0.2 Formulation of Optimal Control Problem

We formulate the problem of squat motion as a two-phase optimal control problem. The model starts the motion in upright standing position:

- **Phase 1 (Squatting Phase)**: The initial posture is the standing pose, and then it goes down to squat.
- **Phase 2 (Standing Phase)**: The model goes back from squat to the upright position.

A general multiphase optimal control can be formulated as:

$$\begin{aligned} &\min_{\boldsymbol{x},\boldsymbol{u}} \int_0^T \Phi(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) \\ \text{s.t.} \quad &\dot{\boldsymbol{x}} = f_j(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \\ &t \in [s_{j-1}, s_j], \\ &j = 1, ..., n_{ph}, \\ &s_0 = 0, s_{n_{ph}} = T, \\ &r^{eq}(\boldsymbol{x}(0), ..., \boldsymbol{x}(T)) = 0, \\ &r^{ineq}(\boldsymbol{x}(0), ..., \boldsymbol{x}(T)) \geq 0. \end{aligned} \tag{C.2}$$

The constraints of the optimal control problem are defined in equation (C.2) where $\boldsymbol{f}_j(.)$ is the differential equation describing the phase j, s_j is the time boundary of the phase j, n_{ph} is the number of phases, and $r^{eq}(.)$ and $r^{ineq}(.)$ are the equality and inequality constraints of the problem. The final time T is fixed in this formulation.

States and Control:

The state vector is $\boldsymbol{x} = \begin{bmatrix}\boldsymbol{\theta}, & \dot{\boldsymbol{\theta}}\end{bmatrix}^T$ where $\boldsymbol{\theta} = \begin{bmatrix}\theta_{ankle}, & \theta_{knee}, & \theta_{hip}, & \theta_{midTrunk}\end{bmatrix}^T$. The control vector is $\boldsymbol{u}(t) = \boldsymbol{\tau}(t) = \begin{bmatrix}\tau_{ankle}, & \tau_{knee}, & \tau_{hip}, & \tau_{midTrunk}\end{bmatrix}^T$. The right hand side of the $\boldsymbol{f}_j(.)$ is represented by the forward dynamics formulation, with the equation of motions and actuation torques described in the previous section.

Objective function: The following objective terms are implemented to simulate the squat motion.

- Minimization of joint torques squared which had shown good motion optimization results in dynamic motions is considered.

$$\Phi(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) = \boldsymbol{u(t)}^T \boldsymbol{u(t)} \tag{C.3}$$

- In addition, minimization of absolute power [132] is considered as well with appropriate weights

$$\Phi(t, \boldsymbol{x}(t), \boldsymbol{u}(t)) = w_1 \boldsymbol{u(t)}^T \boldsymbol{u(t)} + w_2 |\boldsymbol{u}^T \dot{\boldsymbol{\theta}}| \tag{C.4}$$

- To compare the results, we make the model track the dataset collected by [114] with a least squares objective function. As the dataset will make sure the there is a squat, this problem is implemented as a single phase problem. The first term minimizes the residual between joint angles, and the second term is to make the trajectory smooth.

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{u}} & \sum_{j=0}^{m} \frac{1}{2} ||\boldsymbol{q}_j^D - \boldsymbol{q}(t_j)||_2^2 + \gamma ||\boldsymbol{u}(t_j)||_2^2 \\ \text{s.t.} \quad & \dot{\boldsymbol{x}} = f(t, \boldsymbol{x}(t), \boldsymbol{u}(t)), \\ & r^{eq}(\boldsymbol{x}(0), ..., \boldsymbol{x}(T)) = 0, \\ & r^{ineq}(\boldsymbol{x}(0), ..., \boldsymbol{x}(T)) \geq 0. \end{aligned} \tag{C.5}$$

Equality Constraints

- System dynamics for both phases are based on equation (C.1).
- *Initial and end positions*: The motion starts and ends at the standing position where the joint angles and joint velocities are zero.

$$\boldsymbol{x}(0) = \boldsymbol{x}(T) = \boldsymbol{0}_{8\times 1} \tag{C.6}$$

- *Constraint on Hip*: For describing the task, we can say that the pelvis should go down to a certain height. We can have an equality constraint on the beginning of the second phase. This constraint is defined on the vertical distance between the hip and the feet d_{HtoF}, where the desired squat depth is set to the seventy percent of $d_{standing}$ as

$$d_{HtoF} - 0.7d_{standing} = 0 \tag{C.7}$$

This constraint is defined as the starting condition of the second phase.

Inequality Constraints

- *Constraint on Knees*: In the squat motion, knees should not pass the toes. This can be implemented by assuming that knees should not go more than 20cm. This constraint can be transformed into the angle constraint, by calculating the angle between the foot and the shank. The limits for the ankle angle is

$$-2^o \leq \theta_{ankle} \leq 68^o \tag{C.8}$$

- *Constraint on feet*: The feet would be flat and parallel to the ground. This can be defined by the constraint on the ankle torque. The ankle torque should be compensated by the force on the foot times the lever arm. As a first guess, the force is equivalent to the weight of the person. It changes a bit as it starts moving. The force cannot travel forward more than the toes and backward than the heel. Essentially that helps to limit the torque in two directions such that the force travel limits are $0.6l_{foot}$ forward and $0.14l_{foot}$ backward where L_{foot} is the length of the foot [168].

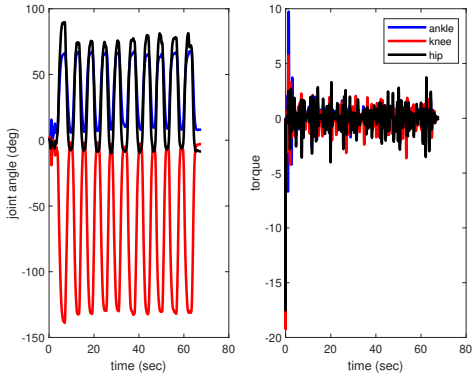
$$\begin{aligned} &\tau_{ankle} = F_{ankle} l_{foot} \\ &F_{ankle} = m_{total} g \\ &-0.14l_{foot} \leq l_{foot} \leq 0.6l_{foot} \end{aligned} \tag{C.9}$$

This constraint is implemented as a range constraint on the ankle torque for a person with 62kg as:

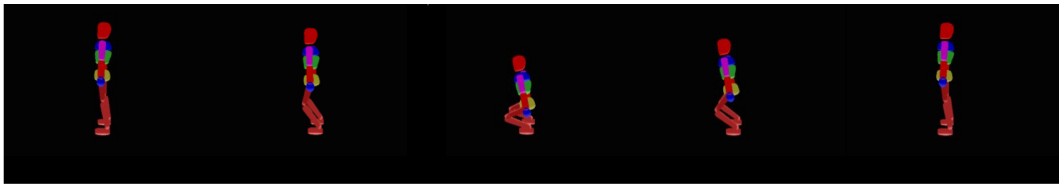
$$-110 \leq \tau_{ankle} \leq 0 \tag{C.10}$$

- Box constraints: For the joint angles, the ranges are defined based on the range obtained from the dataset [114] as Fig. C.1.

$$
\begin{aligned}
&- 2^o \leq \theta_{ankle} \leq 68^o \\
&- 139^o \leq \theta_{knee} \leq 3^o \\
&- 10^o \leq \theta_{hip} \leq 90^o \\
&- 15^o \leq \theta_{midTrunk} \leq 15^o \\
&- 10 \leq \dot{\theta}_j \leq 10 \;\; j = 1, 2, 3, 4. \\
&- 110 \leq \tau_{ankle} \leq 0 \\
&- 200 \leq \tau_i \leq 200 \;\; i = 2, 3, 4.
\end{aligned}
\tag{C.11}
$$



(a)



(b)

Figure C.1: The joint angles and torque for each degree of freedom from the dataset [114]

Software tool: we solve the optimal control problem with two packages: (1) MUSCOD-II [104] and (2) open-source software package Bioptim [124]. At the beginning of formulat-

ing the problem, MUSCOD-II was used but as it is not freely available, we have switched to Bioptim.

C.0.3 Computational Results using MUSCOD-II

This section illustrates the numerical solutions of the optimal control problem. To better understand the role of the mid-trunk joint and observe its effect for describing the squatting task more naturally, we compared the results of two models. One model with four degrees of freedom including the mid-trunk joint and another model with three degrees of freedom excluding the mid-trunk joint. The results from MUSCOD are validated to simulate the intended motion based on the visualization output from RBDL Toolkit.

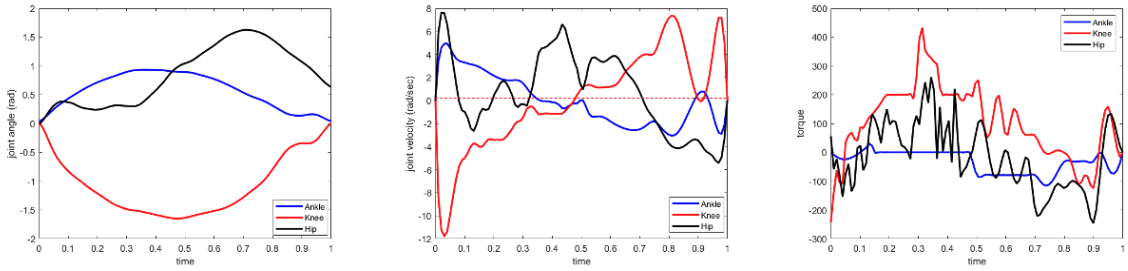
In MUSCOD-II [104], the solution of the optimal control problem is carried out using the direct multiple shooting method described in [21], in which controls and states are discretized. The phase times $s_{n_{ph}}$ are divided into m_{ph} intervals, over which a simple function (constant or linear) is chosen for the controls. State variables are parameterized with the multiple shooting method which uses the same grid as the control discretization. From these two discretizations we obtain a large but structured nonlinear programming problem (NLP), which is solved using an adapted sequential quadratic programming (SQP) method. The dynamics of the systems is handled on all multiple shooting intervals in parallel to the NLP solution by means of efficient integrator at a desired precision.

Two-Phase Optimal Control

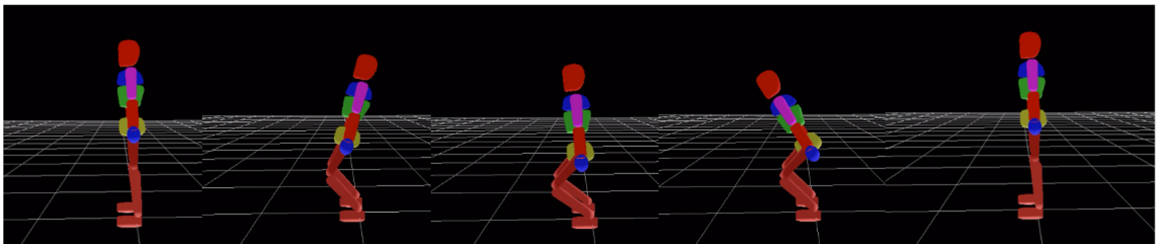
First implementation is the minimum energy consumption problem. Fig. C.2 shows the joint angles, joint velocities and torques for the three DOF model. In three DOF model, it is observed that there is a negative joint velocity for the first phase which makes the model go downward, and there is a positive joint velocity for the second phase which makes the model come back into the upright position.

Fig. C.3 illustrates the same problem on the four DOF model. The ankle, knee and hip joints have bell shaped profile which are more similar to the dataset in Fig. C.1. These profiles are more symmetric than the results from the three DOF model in Fig. C.2(a). These profiles potentially indicate more natural squat motion. The mid trunk angle is between -0.5 rad and 1 rad. In comparison with the torques reported in Fig. C.1 higher torques are obtained here, and high variation is observed.

By comparing the images of one cycle of motion in Fig. C.2(b) and Fig. C.3(b), the posture of the four DOF model at the squatting looks more realistic, but after moving back



(a)



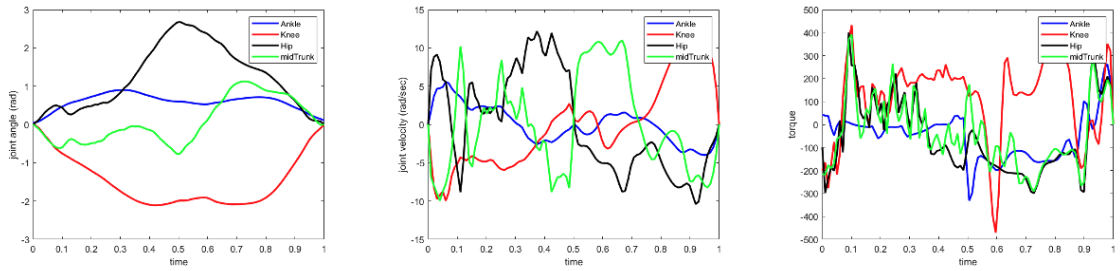
(b)

Figure C.2: Two Phase Minimum Energy Problem in 3 DOF (a) joint angles, joint velocities, and torques, (b) image of one cycle of motion.

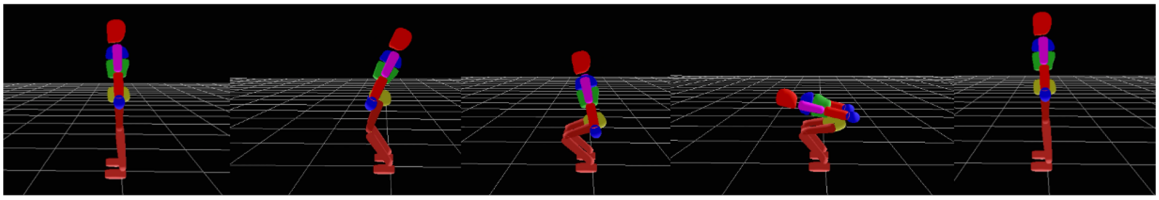
to the standing position the model leans forward more than the three DOF model where the midtrunk joint is also affecting this posture. Also midtrunk joint has high velocity profile which impacts the motion in the 4DOF. By looking at the peak of the hip and knee angles, we can observe that they are higher than the three DOF which comes from satisfying the equality constraint defined in equation (C.7).

Motion Reconstruction

Second implementation is the motion reconstruction by tracking the dataset [114] in the single phase optimal control problem. Results reported here are based on the second squat repetition of the first participant in the dataset. Fig. C.4 shows the tracking results for the three DOF model and Fig. C.5 is for the four DOF model. The optimization was able to generate the squat motion in the single phase problem as the dataset has the squatting profile inside. The knee joint velocity goes from a negative to a positive in a single phase. In the four DOF model, the range of motion of the mid trunk is between -2 to 2 degrees



(a)



(b)

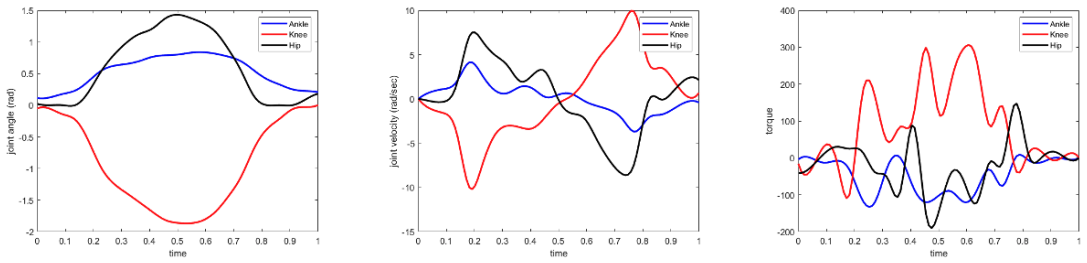
Figure C.3: Two Phase Minimum Energy Problem in 4 DOF (a) joint angles, joint velocities, and torques, (b) image of one cycle of motion.

Fig. C.5 (a). In comparison with the two-phase problem Fig. C.3 (a), the midtruk has lower range of motion and joint velocities. The results on the midtrunk show that, for the two-phase optimal control problem a suitable range for this joint should be considered in order to avoid the model to lean forward less and more naturally.

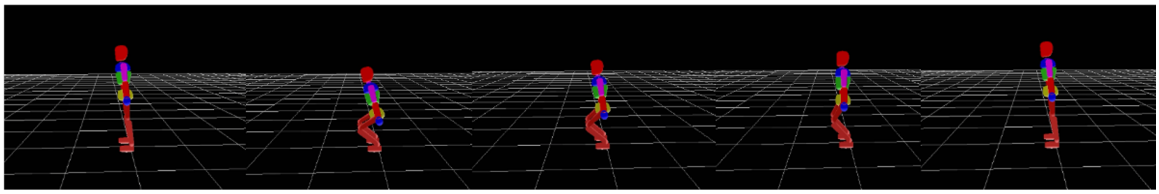
The visualization output of RBDL toolkit in Fig. C.4 (b) shows the three DOF model has a fixed upper trunk with stiff motion. And Fig. C.5 (b) for the four DOF model shows slight movement in the trunk which makes the movement more realistic.

Discussion

The motion reconstruction results verifies that the four DOF model has more potential to simulate the squat motion closer to the natural human squat movement. For the choice of the objective function in the two-phase problem, the power was also implemented. Based on the results, we observed that further investigations on objective functions is needed to study the weighting of the torques and the absolute power in more detail. In the optimal



(a)



(b)

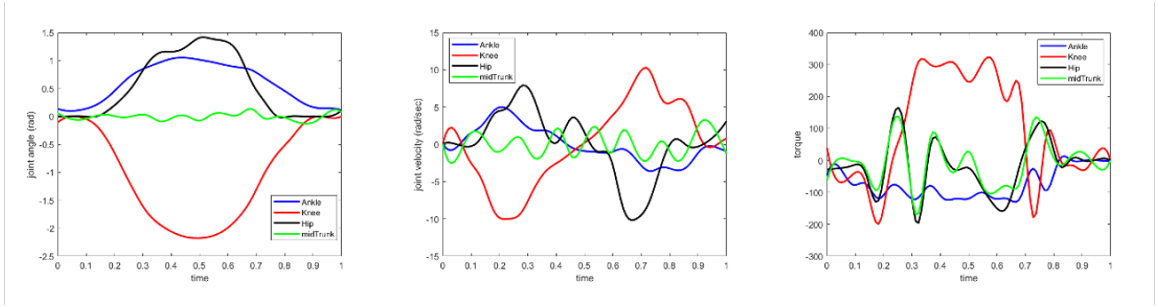
Figure C.4: Motion Reconstruction in 3 DOF

control problem, we explained how the torque on the ankle can be limited to make sure stability. There should also be exploration on suitable ranges for knee, hip and mid trunk torque to get results closer to the tracking problem as well.

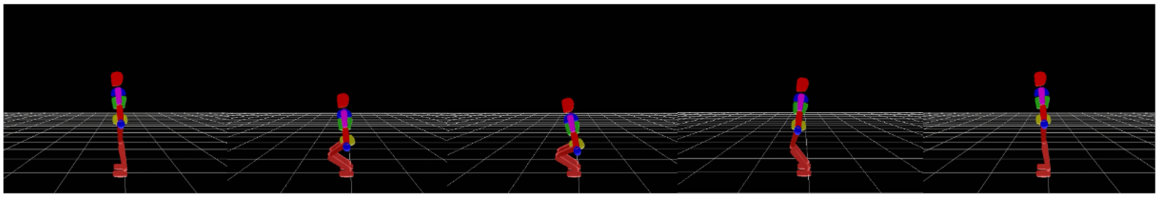
The optimal control results are compared in Fig. C.6 from the absolute error between joint angles from MUSCOD and the dataset. Error in motion reconstruction with four DOF is observed to be the lowest. By comparing the results from two optimal control problems, motion reconstruction outperforms the two-phase problem in terms of the errors for both three DOF and four DOF models. The hip error for the tracking results are less than 0.1 rad which shows deeper squat posture. Overall, Fig. C.6 suggests that the box constraints summarized in equation (C.11) should be modified in order to achieve better optimal solutions.

C.0.4 Computational Results using Bioptim

The model used in Bioptim is shown in Figure C.7. The blue spheres are the marker sets used during the data collection which is from Chapter 5. From the calibration data in the form of .c3d in Vicon, a few frames are taken which are static. Then the data is converted



(a)

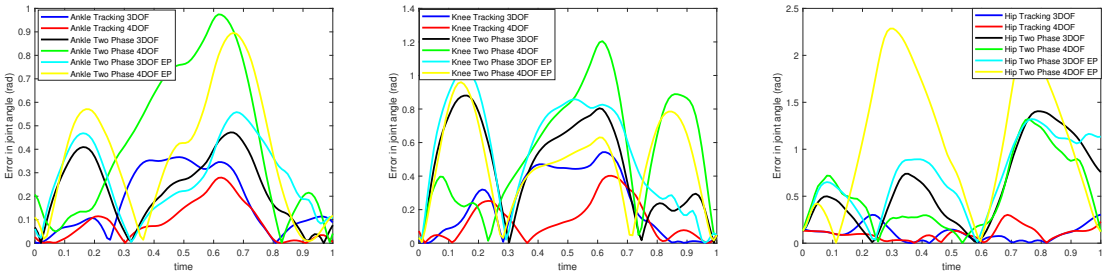


(b)

Figure C.5: Motion Reconstruction in 4 DOF

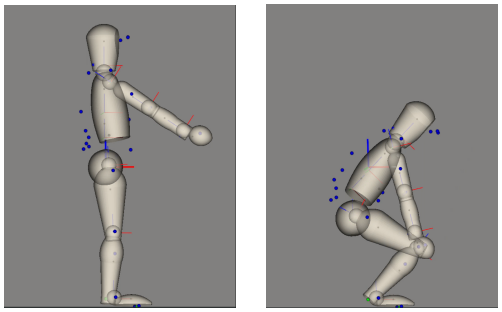
to a .lua model. After that a .bioMod model is manually constructed from the .lua model. In this model, the arms are fixed at some angle in front of the body. By applying inverse kinematics on the data collected in Chapter 5, the joint angles are obtained as Figure C.8 for an unperturbed trial. This model is a floating-based model where the root is in the pelvis and contact points are defined in the feet. To fix the foot in the ground, contact points are defined in the model.

In the next step, we implemented the optimal control to check the motion. The results are obtained as Figure. C.9 where the objective functions include minimizing the torque squared and the final time. Through this implementation, the optimal solution is found; however, the contact point is not well defined, and unilateral constraints have to be included in the dynamics. This part is left as the future work.



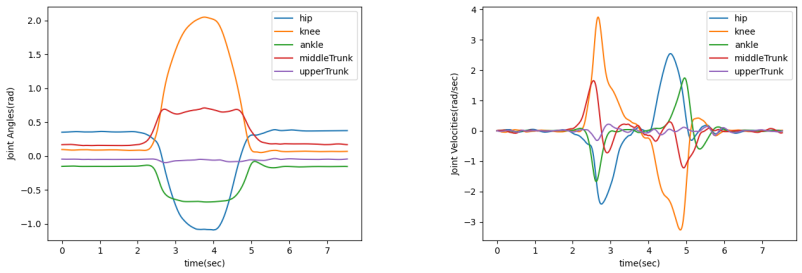
(a) (b) (c)

Figure C.6: Absolute error of joint angles. The results are from (blue): tracking the three DOF model, (red): tracking the four DOF model, (black): two-phase minimum energy three DOF model, (green): two-phase minimum four DOF model, (light blue): two-phase minimum energy plus power three DOF model, (yellow): two-phase minimum energy plus power four DOF model.



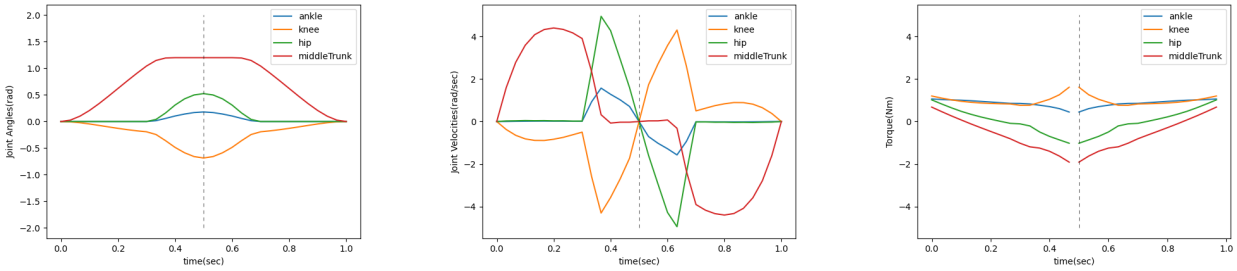
(a) (b)

Figure C.7: Visualization of the .bioMod model for (a) standing phase and (b) squatting phase.



(a) (b)

Figure C.8: Inverse kinematics results (a) joint angles, and (b) joint velocities.



(a) (b) (c)

Figure C.9: Direct optimal control results (a) joint angles , (b) joint velocities, (c) torques.

Appendix D

Estimating Feedback Component using Bellman Equation

One approach of estimating the feedback control signal dependent on state is by solving the discrete-time Bellman optimality condition. Let's reformulate the discrete-time dynamic equations in the affine state space difference equation form

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k) + \boldsymbol{G}(\boldsymbol{x}_k)\boldsymbol{u}_k, \tag{D.1}$$

with the feedback control law $\boldsymbol{u}_k$ as

$$\boldsymbol{u}_k = \boldsymbol{\psi}(\boldsymbol{x}_k), \tag{D.2}$$

Let us define the discrete-time Hamiltonian as [108]

$$\mathcal{H}(\boldsymbol{x}_k, \boldsymbol{u}_k, \Delta V(\boldsymbol{x}_k)) = \boldsymbol{q}(\boldsymbol{x}_k) + \frac{1}{2}\boldsymbol{u}_k^T \boldsymbol{R}\boldsymbol{u}_k + V_k(\boldsymbol{x}_{k+1}) - V_k(\boldsymbol{x}_k), \tag{D.3}$$

the above equation is the temporal difference error [108]. The Bellman equation requires that the Hamiltonian be equal to zero for the value associated with a prescribed policy. The first order necessary condition for a control law to be optimal is to calculate the gradient of the right hand side of HJB equation (D.3) with respect to $\boldsymbol{u}_k$ as [6]

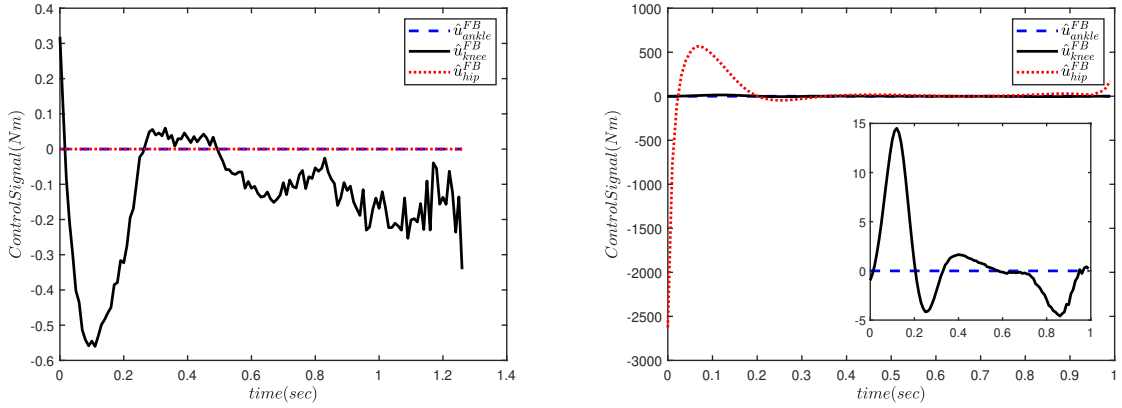
$$\frac{\partial(\boldsymbol{q}(\boldsymbol{x}_k) + \frac{1}{2}\boldsymbol{u}_k^T \boldsymbol{R}\boldsymbol{u}_k)}{\partial \boldsymbol{u}_k} + \frac{\partial \boldsymbol{x}_{k+1}^T}{\partial \boldsymbol{u}_k}\frac{\partial V_k(\boldsymbol{x}_{k+1})}{\partial \boldsymbol{x}_{k+1}} = 0 \tag{D.4}$$

and therefore the predicted feedback part at each sample time is

$$\hat{\boldsymbol{u}}_k^{FB}(\boldsymbol{x}_k) = -\boldsymbol{R}^{-1}\boldsymbol{G}(\boldsymbol{x}_k)^T \frac{\partial V_{k+1}^*(\boldsymbol{x}_{k+1})}{\partial \boldsymbol{x}_{k+1}}, \tag{D.5}$$

In this formulation, if we have objective terms for human squat motion in chapter 5 on torque and power the matrices can be defined as $\boldsymbol{R} = \begin{bmatrix} w_1 + w_4\dot{\theta}_{1,k} & 0 & 0 \\ 0 & w_2 + w_5\dot{\theta}_{2,k} & 0 \\ 0 & 0 & w_3 + w_6\dot{\theta}_{3,k} \end{bmatrix}$ and $\boldsymbol{G} = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{M}^{-1}(\boldsymbol{\theta})\boldsymbol{B} \end{bmatrix}$. By inputting $\hat{\boldsymbol{u}}_k^{FB}(\boldsymbol{x}_k)$ to the discrete form of the system dynamics, we can get the estimated state of the system for the next sample time.

By estimating the control signal of the dataset in chapter 5 using this approach, we get the results as Fig. D.1. For the unperturbed trial, the magnitude of the control signal in the state feedback form is zero for the hip and the ankle, and approximately zero for the knee. The output of the perturbed trial shows that the feedback is active for perturbed trial in the knee and hip. However, the magnitude on the hip is really large. To get reasonable range for the torque, other objective terms have to be included in the Hamiltonian to extract information correctly.



(a) (b)

Figure D.1: Estimated feedback part for (a) unperturbed trial A after perturbation, and (b) perturbed trial A.