Inferring human intent in novice human-in-the-loop control tasks

Rahul B. Warrier

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2018

Reading Committee:

Santosh Devasia, Chair

Brian Fabien

Joseph Garbini

Ashis Banerjee

Linda Ng Boyle

Program Authorized to Offer Degree:
Mechanical Engineering

University of Washington

Abstract

Inferring human intent in novice human-in-the-loop control tasks

Rahul B. Warrier

Chair of the Supervisory Committee:
Santosh Devasia
Mechanical Engineering

Inferring intentions is fundamental to successful interaction among two or more agents. For example, in learning from a teacher the student must be able to first understand the teacher's intent from the displayed demonstrations which is then committed to memory by imitation and self-practice. Analogously, robots may also be taught new skills using this Teaching by Demonstration (TbD) paradigm, in which case inferring the human teacher's intent becomes necessary, especially when precision in learning new skills is important, such as in precision assembly tasks or surgery. The primary challenge in learning from a human-in- the-loop operator is that the actions do not always match the intended goal. This stems from the inherent dynamics of human response, which affects the overall closed loop tracking performance when the human is placed in the control loop. This work proposes an inversion approach to compensate for the human-in-the-loop dynamics to infer the underlying intent of the human-in-the-loop operator's actions. As a result, the task representation using this inferred intent is potentially improved over that based on imperfect human demonstrations.

The following are the main contributions of this work: (1) Stable inversion of human dynamics models to iteratively infer human-in-the-loop intent, with convergence guarantees robust to modeling uncertainty and output noise, (2) One-shot human intent prediction with guarantees on improved tracking accuracy compared to demonstrations, (3) Experimental

validation of iterative and one-shot human-intent prediction and its application to human-guided robot learning with multiple human subjects.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

Glossary

ω_c Gain crossover frequency in rad/s.

ω_{co} (Nominal) Gain crossover frequency in rad/s.

τ_d Time-delay in the controlled system.

τ_e Effective time-delay of human feedback response.

τ_{eo} (Nominal) Effective time-delay of human feedback response.

τ_I Lag time-constant of human equalization characteristics.

ω_i Frequency bandwidth of desired trajectory, y_d in rad/s.

τ_L Lead time-constant of human equalization characteristics.

ϕ_M Phase margin.

d_i Additive internal disturbance.

d_o Additive external disturbance.

e Output error, $y_d - y$.

f_{BW} Frequency bandwidth of desired trajectory, y_d in Hz.

K_c Controlled element gain.

K_p Human operator gain.

u_h Feedback controller input (human input for human-in-the-loop case).

y Controlled system output.

y_d Desired trajectory to be tracked.

ACKNOWLEDGMENTS

I'm deeply indebted to Prof. Santosh Devasia for providing thoughtful and patient guidance throughout the graduate school journey and sincerely appreciate his dedication and enthusiasm in pushing me to succeed even when I doubted myself.

Also, I'm filled with gratitude towards everyone at University of Washington, where I have had the pleasure to work with a great number of honest and friendly colleagues who have contributed immensely to the successful completion of this body of work.

I would also like to acknowledge the National Science Foundation for supporting me through a major part of my graduate school through the NSF Grant CMMI 1536306.

And, lastly, none of this effort would have been possible without the unconditional love and support of my family who deserve this recognition as much as I do.

DEDICATION

to my dear wife, Nishita, and to my wonderful family for always supporting (and at times enduring) me through this long and fulfilling journey.

Chapter 1

INTRODUCTION

Accurately estimating the human operator's intent can enable robot controllers to provide assistive actions and/or learn specific output trajectories (actions) from the human. For example, Teaching or Programming by Demonstration (TbD or PbD), is used for training assistants for people with disabilities [45], and teaching upper limb prostheses natural motion by mimicking the motion of an intact limb [61]. Similarly, known intent can be used to improve shared surgical task performance [29], to aid human-robot collaborative manipulation [52], and to improve lower extremity exoskeleton performance [33]. However, a fundamental limitation is that the human intent y_d is not directly known in human-in-the-loop operation (tele-operation) of robots. Rather, what is observable is human action r_h (human motor response), which is affected by the human-motor dynamics. Directly following the human action r_h (by considering it as the desired intent y_d) can lead to performance loss especially when tracking relatively fast trajectories, as illustrated in Fig. 1.2. Moreover, the dynamics of the system can make it difficult to precisely demonstrate the desired intent. These lead to potentially significant differences between the demonstration y and the intent y_d. Therefore, this work considers the problem of inferring the intended output trajectory y_d from the demonstrations y of a human-in-the-loop (HIL) operator.

A primary challenge with incorporating a human operator into the control loop is that the human-motor dynamics (from sensory perception to motor action) affect the overall closed-loop tracking performance. The effect of the human-motor dynamics is more pronounced if

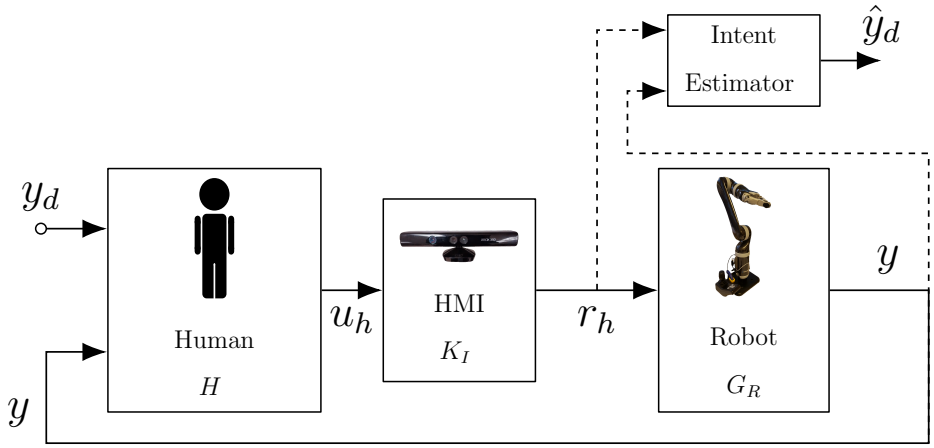


Figure 1.1: Block diagram of (a) the human-in-the-loop output tracking task, where the human operator, H applies an input r_h through the Human-Machine Interface (HMI) to make the output y of the robot system G_R track the intended goal trajectory y_d.

the system is more complex or the task involves higher bandwidth output tracking. Thus, following the human action may lead to an inaccurate representation of the underlying task, especially if the controlled system is complex or the required tracking bandwidth is outside the scope of the human operator's tracking ability. Extensive training can enable humans (i.e., experts) to overcome limitations of human-motor dynamics and achieve precision human-robot operation — even for complex systems. However, such extensive training incurs substantial time and cost, which may be acceptable only in high-end applications such as remote surgery. Moreover, precision performance tends to be restricted to slow operations (i.e., low bandwidth tracking). While approaches that can classify human intent can be used to control machines, e.g., [75], precision operations such as tele-operating a robot to perform a fine task *rapidly* remains challenging [63]. In contrast, the proposed effort seeks to make the machine more intelligent for improving precision even with novice operators, which can make human-machine partnerships economically affordable in broader applications, such as low-volume manufacturing, service industries and home applications.

(a) Slow reference trajectory

(b) Fast reference trajectory

Figure 1.2: Differences between human-in-the-loop demonstrations y (shown in red, – – –), the intended reference trajectory y_d (shown in black, ——) as well as the human action r_h (shown in blue, –.–) are small for slow reference trajectories y_d but become significant for faster trajectories.

Towards such fast, precision human-robot partnerships, this research considers the problem of a robotic controller inferring an intended output trajectory from the actions of a human-in-the-loop (HIL) operator. Of particular interest is this case where the desired trajectory is known/available to only the human operator, and the goal is for the robot to learn the desired trajectory from the human operator's actions, as shown in Fig. 1.1. Such problems have important applications in the field of Robot Learning from Demonstration (LfD), for example in training assistants for people with disabilities [45], or teaching upper limb prostheses natural motion by mimicking the motion of an intact limb [61]. Similarly, known intent can be used to improve shared surgical task performance [29], to aid human-robot collaborative manipulation [14], and to improve lower extremity exoskeleton performance [33].

Motivation

In this work, the human intent is defined as the reference trajectory y_d that the human operator is trying to track using the system output y. In this context, the human operator is

not assumed to be an expert at the tele-operated tracking task, i.e., there may be significant difference in the human action r_h and the intended goal y_d as seen in Fig. 1.2, and so using the human demonstration directly will result in lower precision. Additionally, the intent is only available to the human operator, i.e., no predefined set of trajectories are available for a classification approach. This motivates the proposed intent estimation approach for human-in-the-loop output tracking tasks by inverting human response models.

1.1 Research Goal and Contributions

The primary goal of this research is stated as follows:

> *Improve the precision of robot-motion primitives learned through observations of a novice human operator, by modeling and compensating for the human-response dynamics.*

In other words, recover the underlying human intent by inverting human-response dynamics and use the recovered intent to provide assistance to the HIL operator.

1.1.1 Research question and challenges

Towards this goal, the primary research questions that will be addressed are as follows:

(P1) Desired (intended) output inference: Can we infer the human intent by inverting human-in-the-loop dynamics models?The main challenge in inferring human intent by model-inversion is that human models exist for simple types of controlled systems but not for general controlled systems. Finding specific models for each task is cumbersome and limits the widespread use of human-robot systems.

(P2) Iterative intent learning: When do iterative human-in-the-loop intent estimation methods converge under modeling uncertainty and output noise?The main challenge in this case is that modeling uncertainty leads to instability in a model-inversion-based iterative learning controller especially at higher frequencies. This results in either slower

convergence or limited bandwidth. Convergence conditions that depend on modeling uncertainty will help in designing robust and aggressive learning schemes.

(P3) One-shot intent learning: When do human-in-the-loop dynamics models successfully estimate the human intent better than the human demonstration?In this case, the primary challenge is the absence of iterations to compensate for the modeling uncertainty, in which case directly using the computed human-in-the-loop dynamics models may lead to larger tracking errors than the human-demonstrations especially at frequencies where modeling uncertainty is large.

1.1.2 Main contributions

With regard to the research questions posed in the previous section, the following are the main contributions of this thesis that address each of the posed challenges.

(C1) Stable inversion of simple human models by modifying the perceived controlled system (that is potentially complex) using an online inverse controller

(C2) Convergence conditions for iterative human-in-the-loop robot learning based on modeling uncertainty; experimental validation with multiple human subjects; extension of inversion-based learning to collaborative tasks.

(C3) Conservative inversion condition that guarantees the one-shot inversion of human-in-the-loop dynamics models results in better intent estimation than the human demonstration.

1.2 Organization

The rest of the report is organized as follows.

- Chapter 2 provides a brief survey of literature for related work in estimating human intent, and discusses the various approaches to modeling human response dynamics, with a particular emphasis on the *crossover model*, which has been used as a proof-of-concept in this work. Also, a brief survey of other human operator modeling techniques available in existing literature are summarized. Finally, the chapter is concluded with a brief survey of parametric and non-parametric system identification techniques which proves useful in modeling human-in-the-loop response from experimental training data.

- Chapter 3 addresses research problems (P1) and (P2) in the frequency-domain, specifically, it introduces the iterative learning framework for batch-wise human-in-the-loop intent estimation and shared control (C1). A rigorous convergence analysis is presented for the proposed iterative update scheme, with robustness guarantees based on modeling uncertainty and output noise levels (C2). Also, methods to account for general linear controlled systems for which nominal human models are not available are presented, with an additional non-parametric model update algorithm when additional training data is present (C1). Finally, the chapter concludes with a discussion of human-in-the-loop output tracking experiments with multiple human subjects, and the limitations of the proposed method.

- Chapter 4 addresses research problem (P3) in the frequency-domain, specifically it approaches the human intent estimation problem as a one-shot learning method, where a pre-computed human-in-the-loop dynamics model (parametric/non-parametric obtained from training data) may be inverted robustly (in the presence of modeling uncertainty) to result in better intent estimation than the human demonstrations (C3).

- Chapter 5 addresses research problem (P3) in the time-domain, specifically it discusses the implementation of the model-inversion based human intent estimation method

in time-domain which allows for evaluating the feasibility of using online inverse controllers to estimate the human intent during human-in-the-loop output tracking. Specifically, the main contribution of this chapter is to study the amount of preview information required to estimate intent using pre-computed inverse impulse response functions such that the prediction error is reduced compared to the human-demonstrations.

- Finally, Chapter 6 summarizes the main contributions and results from this work. Additionally, a brief discussion of the limitations is presented followed by potential directions of future work.

1.3 List of publications

1. Rahul B Warrier and Santosh Devasia. "Iterative learning from novice human demonstrations for output tracking". In: *IEEE Transactions on Human-Machine Systems* 46.4 (2016), pp. 510–521

2. Rahul B Warrier and Santosh Devasia. "Inferring Intent for Novice Human-in-the-Loop Iterative Learning Control". In: *IEEE Transactions on Control Systems Technology* (2016)

3. Rahul B Warrier and Santosh Devasia. "Inverse control for inferring intent in novice human-in-the-loop iterative learning". In: *American Control Conference (ACC), 2016.* IEEE. 2016, pp. 2148–2154

4. Rahul B Warrier and Santosh Devasia. "Data-based Iterative Human-in-the-loop Robot-Learning for Output Tracking". In: *IFAC-PapersOnLine* 50.1 (2017), pp. 12113–12118

5. Jonathan Realmuto, Rahul B Warrier, and Santosh Devasia. "Iterative learning control for human-robot collaborative output tracking". In: *Mechatronic and Embedded*

Systems and Applications (MESA), 2016 12th IEEE/ASME International Conference on. IEEE. 2016, pp. 1–6

6. Jonathan Realmuto, Rahul B Warrier, and Santosh Devasia. "Data-Inferred Personalized Human-Robot Models for Iterative Collaborative Output Tracking". In: *Journal of Intelligent & Robotic Systems* (2017), pp. 1–17

7. Rahul B Warrier and Santosh Devasia. "Kernel-based human-dynamics inversion for precision robot motion-primitives". In: *Intelligent Robots and Systems, 2018 IEEE/RSJ International Conference on*. IEEE. 2018, (accepted, in print)

8. Rahul B Warrier and Santosh Devasia. "Kernel-based intent estimation to improve robot learning from human-in-the-loop demonstrations". In: (2018), (submitting, in review)

(P1) modeling	**(P2) iterative learning**	**(P3) one-shot learning**
1. IEEE THMS, Oct. 2016 (Journal)	4. Springer JIRS, 2017 (Journal)	7. IROS 2018, Madrid (Conf.)
2. IEEE TCST, Dec. 2016 (Journal)	5. MESA 2016, Auckland (Conf.)	8. IEEE Robotics, In Review (Journal)
3. ACC 2016, Boston (Conf.)	6. IFAC 2017, Toulouse (Conf.)	

Chapter 2
BACKGROUND AND RELATED WORK

This chapter presents a brief survey of related work in the literature thereby providing contextual background for the work in this thesis. Mainly, the following three broad areas are discussed: (1) human intent prediction/estimation, (2) human operator performance modeling, and (3) system model identification methods.

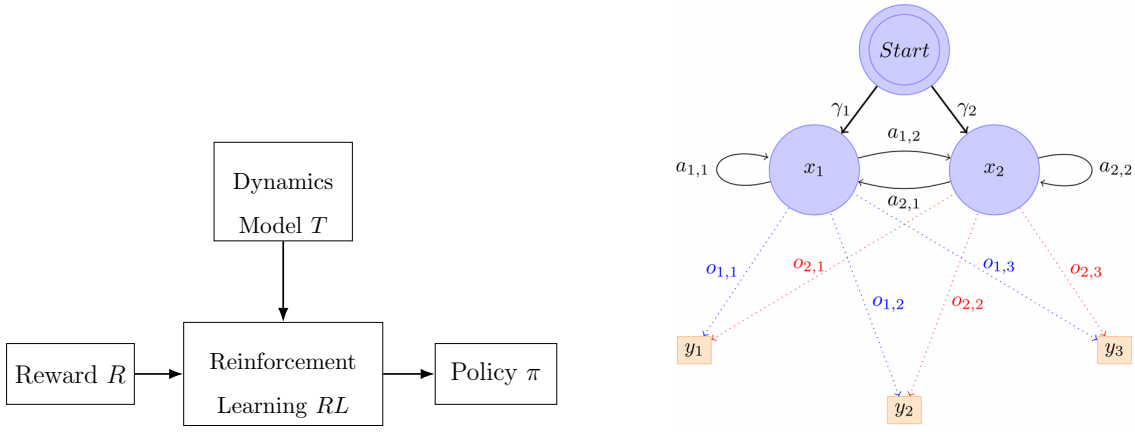
2.1 Human Intent Prediction

In any team, successful collaboration is predicated on the ability of each team member to understand the intentions of the other agents while performing a shared task. The same holds true for human-robot shared control, where the ability of the robot to estimate the intention of the human operator allows the robot to plan ahead and adapt its behavior accordingly. The following sections describe some of the intent estimation approaches found in current literature.

2.1.1 Derive a task representation from expert demonstrations

One method of learning task intent from demonstrations is to consider the human demonstrator to be an expert and derive a task representation directly from the expert demonstration. For example, the human operator may be assumed to maximize some unknown reward function while performing the task, which may be inferred using Inverse Reinforcement Learning [1], e.g., using Gaussian process reward representations [44].

Another related approach is to use Hidden Markov Models (HMMs) to represent the human demonstrations, e.g., [6] discusses driver intention estimation near a road intersection



(a) Inverse Reinforcement Learning (b) Hidden Markov Model

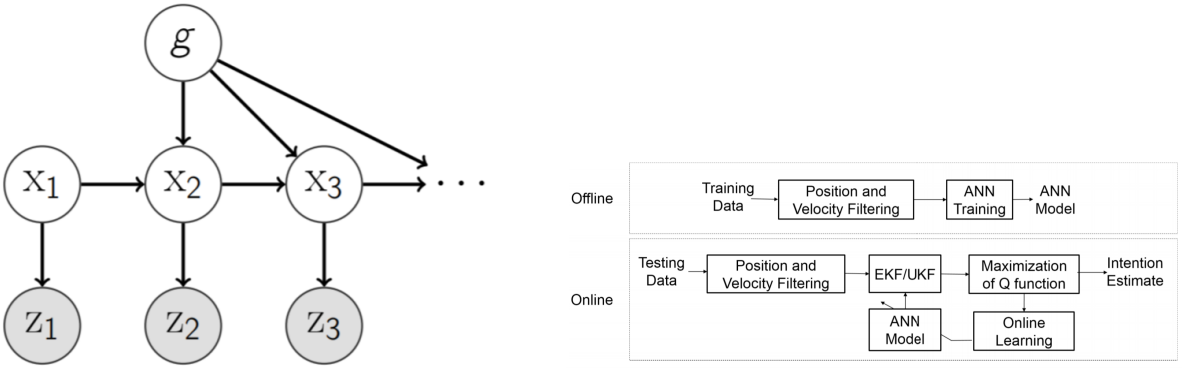
Figure 2.1: Emulating expert human behavior: (a) Inverse Reinforcement Learning framework: Given the control policy π and the dynamics model of the task T (or successful execution demonstrations) can we learn the reward function R? [1], (b) Modeling the driver decisions at a road intersection as a Hidden Markov Model (HMM) which upon training with labeled data results in an HMM capable of predicting driver intent based on the input information such as vehicle speed and yaw-rate [6].

using discrete HMMs where the model is trained on an expert driver's decision making based on the vehicle's speed and yaw-rate at each time step.

The level of performance of such algorithms depend on the ability of the expert human demonstrator to specify their intention y_d as accurately as possible in the demonstrations y. In contrast, the current work relaxes the assumption that the human demonstrator is an expert and allows for sub-optimal demonstrations, which are postulated as the effect of the human operator's dynamics. Using the estimated intent $\hat{y}_d$ with the proposed approach could improve current task-learning approaches (that are based on the demonstrations y) when the demonstrator is not an expert, and the demonstrations differ from the intent, i.e, $y \neq y_d$.

2.1.2 Classify from a set of known intentions

A second class of approaches assumes that there is a finite set of known intentions which must then be recognized from the human demonstrations. For example, [65] uses Gaussian



(a) Intention-driven dynamics model [65] (b) Adaptive-neural-intention-estimator [49]

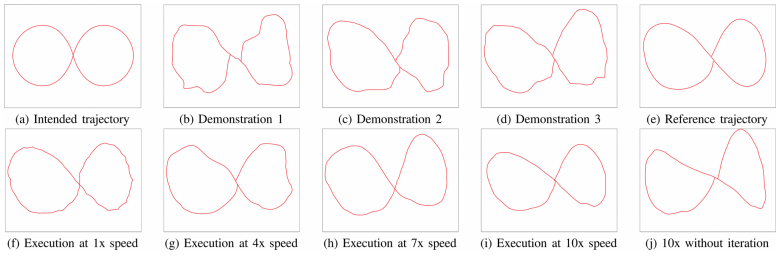
Figure 2.2: Motion intent classification methods (a) Intention-driven dynamics model (IDDM) based on Gaussian Process Dynamics Modeling, adapted from [65], (b) Adaptive neural intention estimator (ANIE) where a neural network represents the nonlinear human arm dynamics, adapted from [49].

Process Dynamics Modeling (GPDM) to model and then estimate intended goal positions based on observed movement, applied to a table-tennis playing robot, while [49] infers goal locations of reaching motions by observing human operator arm movements whose unknown nonlinear dynamics are learned by training a neural network. In the former case, the final location of the ball represents the task intent, while in the latter example the parameters of the human arm dynamics represents the unknown intent corresponding to each goal location.

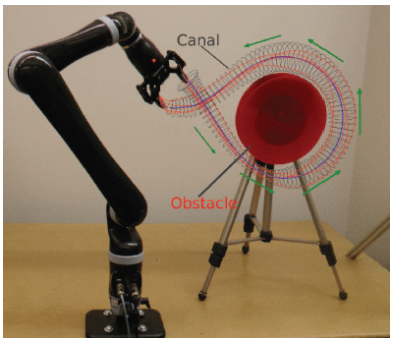
In this case, the learning and classification of intentions depends on the assumption that the human demonstrations match the underlying intent among the set of known intents. If the human demonstrations deviate from the actual intent, e.g., when the tracking bandwidth is outside the capability of the human demonstrator, the intent classification accuracy might

suffer if it is based on the human demonstrations directly as compared to the current approach of compensating the human-in-the-loop dynamics.

2.1.3 Encode average demonstration for task emulation



(a) Dynamic Movement Primitives [60]



(b) Skill encoded as Canal Surfaces [3]

Figure 2.3: Trajectory-based learning methods that encode the task from the average human demonstrations using (a) Dynamical Movement Primitives [28], (b) Canal surfaces [3].

For trajectory-based learning, the human demonstrations are usually averaged in various ways to better represent the task intent. For example, the method of Dynamical Movement Primitives (DMPs) [28], encodes the average of the human demonstrations as the output of a canonical dynamical system with a nonlinear forcing term that is fit to the training data. Moreover, DMPs have the ability to scale temporal variations with a phase variable, and by learning the time mapping between demonstrations and the intention the learned intent may be executed at a higher speed than the demonstration. For example, in [60] iterative learning was used to learn the time mapping between demonstrations and intent, enabling a surgical robot to tie a knot faster than the human demonstrator. Geometrical approaches have also been explored, e.g., in [3] spatial correlations are identified from different demonstrations

which are then used to form the complete task representation in the form of Canal surfaces, which are envelopes around the expert demonstrated trajectories.

In summary, the common assumption in all of these approaches is that the human demonstrator is an expert and so the task representation is based on the average of the human demonstrations. But, in practice the actual intention may not match the demonstrations when the human operator dynamics begin to adversely affect the human demonstrations. As a result, directly emulating the human demonstrations leads to an inaccurate representation of the task or will be limited to slower speeds (or, lower frequencies) to avoid distortion by the human operator dynamics. This article is thus focused on improving the learning of motion primitives from human demonstrations by learning and compensating the human operator dynamics using a data-based model-inversion approach.

2.2 Human operator models

Modeling of human operators "in-the-loop" has been studied extensively in literature, emanating primarily from early aircraft pilot studies [38, 11, 19, 31, 36, 42, 43, 40, 41, 56, 72]. These early mathematical models of human pilot behavior, for example [41], aimed to describe the experimentally observed characteristics of human response to varying types of systems and stimuli. The objective of these studies was to develop a describing function that best matches the experimentally observed human pilot response. In this work, the simplest model of human response to pursuit and compensatory tracking tasks is considered, namely the *analytic-verbal model* [41] based on the empirically determined *crossover model* [38] for human-machine systems.

2.2.1 The Crossover Model

The pioneering work by Mcruer et al. introduced the widely accepted analytic *crossover model* of human feedback response to single-loop compensatory tracking tasks [41]. The

crossover model, in its various levels of precision, was based on extensive data collected from pilot response studies to various simple controlled system dynamics typically found in aircraft systems. The primary stated purpose of these analytic models was to quantitatively describe the experimentally observed trends in terms of analytic relationships between model parameters and task variables. In addition, it was also expected to form a basis for quantitative extrapolation in order to predict pilot behavior in novel situations. Three basic levels of precision for the analytic model were introduced, with greater accuracy at higher precision levels offset by increased complexity in their structures. In this section, we analyze the *open-loop crossover model*, which has the simplest structure, and which forms the basis for the more complex model structures.

The human as a "good" feedback controller

The task of finding an analytic model to explain the behavior of a human operator is decidedly formidable, because of the bewildering complexity of ways in which a human operator may respond. Even if all the environmental and task conditions are controlled, the human operator will still show an assortment of response strategies that are intractable to generalize. This was precisely the reason why early researchers in the field were inclined to avoid a mathematical analysis of the pilot-vehicle system, instead focusing on models for the machine to explain experimental data [37]. Although this approach did lead to some success, it is readily seen that such an approach is very narrow and tends to be wasteful, since each new configuration introduces new dimensions to the experiment, requiring a fresh analysis.

The distinguishing feature of the work by McRuer et al. was to recognize that to control a complex system, the characteristics of *successful* behavior of a feedback controller are narrowly limited. This approach constrains the successful behavior by the human operator as a feedback controller in terms of well established rules that govern the performance of a "good" feedback controller. This seemingly simple but profound insight was proven to match

the carefully obtained experimental evidence to reasonable accuracy, and thus, paved the way for a truly tractable and scale-able mathematical analysis of human-machine systems.

Requirements of a good feedback controller

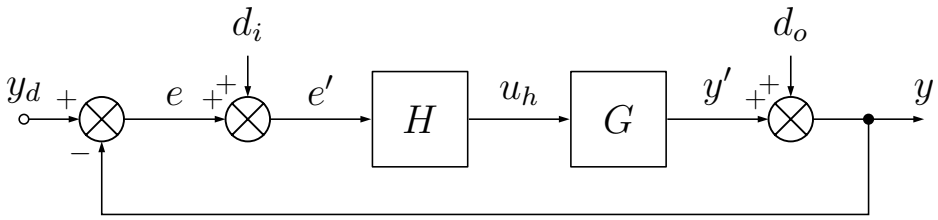


Figure 2.4: An example closed-loop system for compensatory tracking of reference signal, y_d by the controller, H. The controlled system, G acted upon by the controller input, u_h generates an output, y' which is corrupted by an additive external disturbance, d_o resulting in the net output, y. Note that in this setting, the controller, H has access only to the modified error signal, $e' = e + d_i$, where $e = y_d - y$ and d_i is an internal disturbance.

A "good" feedback controller has the following desirable properties: 1) provide specified command-response relationships, 2) suppress unwanted inputs and disturbances. Consider the closed-loop system shown in Fig. 2.4. The feedback controller, $H(s)$ with the above desirable properties will have the following open-loop characteristics, as shown in Fig. 2.5.

1. For low frequencies, $\omega/\omega_c \ll 1$, which contains the bandwidth, ω_i of the desired output, y_d, the open-loop gain must satisfy, $|G_{ol}(\omega)| = |G(\omega)H(\omega)| \gg 1$, which ensures that the closed-loop gain,

$$|G_{fb}(\omega)| = \frac{|y(\omega)|}{|y_d(\omega)|} = \left|\frac{G(\omega)H(\omega)}{1 + G(\omega)H(\omega)}\right| \to 1 \tag{2.1}$$

for good tracking performance, i.e. $y(\omega) = y_d(\omega)$. Also, we require the output disturbance, d_o to have significant frequencies in this range such that the sensitivity of the

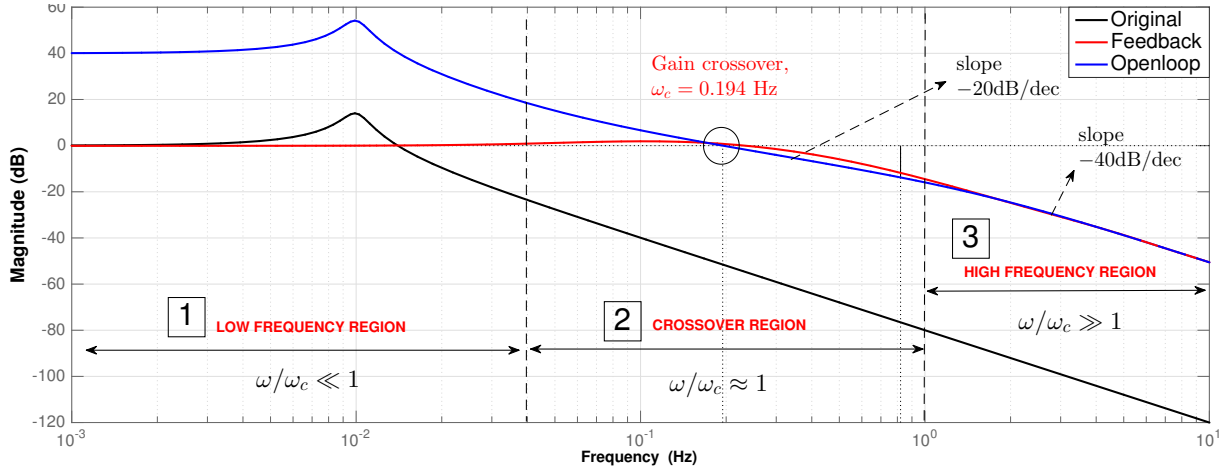


Figure 2.5: Characteristics of a good feedback controller. The solid black line depicts the magnitude of the frequency response of the controlled system, G, which is chosen as an under-damped second order system, $G(s) = K_c/(s^2 + 2\zeta\omega_n s + \omega_n^2)$. The solid blue line depicts the magnitude of the open-loop frequency response, $G_{ol}(\omega) = G(\omega)H(\omega) = y(\omega)/e(\omega)$. The solid red line depicts the magnitude of the closed-loop frequency response, $G_{fb}(\omega) = y(\omega)/y_d(\omega)$. The controller, $H(\omega)$ is chosen based on the verbal adjustment rules described in Section 2.2.1.

closed-loop system to output disturbances,

$$|S_o(\omega)| = \frac{|y(\omega)|}{|d_o(\omega)|} = \left|\frac{1}{1 + G(\omega)H(\omega)}\right| \to 0 \tag{2.2}$$

2. For frequencies in the range, $\omega/\omega_c \approx 1$, we require the closed-loop system to be stable, i.e. the open-loop frequency response must have a positive phase margin, $\phi_M > 0$. Typical feedback controllers have phase margins, $\phi_M \approx 45°$. Thus, choosing an integrator form for the open-loop dynamics in this frequency range, the nominal phase margin is $\phi_M = 90°$, i.e.

$$G_{ol}(\omega) = G(\omega)H(\omega) = \frac{1}{j(\omega/\omega_c)} \tag{2.3}$$

Note that if the controller, $H(s)$ has some delays associated with it, the phase margin,

ϕ_M is reduced, justifying the choice of the integrator form for the open-loop dynamics. This ensures that the resulting phase margin is still positive, for stability.

3. For high frequencies, $\omega/\omega_c \gg 1$, beyond the bandwidth, ω_i of the desired output, y_d, and the bandwidth of the output disturbances, d_o, the input disturbance, d_i is typically significant. From Fig. 2.5, the open-loop gain, $|G(\omega)H(\omega)| \ll 1$, which implies,

$$|S_i(\omega)| = \frac{|y(\omega)|}{|d_i(\omega)|} = \left| \frac{G(\omega)H(\omega)}{1 + G(\omega)H(\omega)} \right| \to 0 \tag{2.4}$$

Note that in this frequency range, $|G_{fb}(\omega)| \to 0$, which implies the closed-loop system does not track the desired output, y_d in this frequency range. But, this is not an issue because ideally the desired output, y_d and the output disturbance, d_o do not have components in this frequency range.

With the above requirements in mind, McRuer et al. [38] proposed the *crossover model* for the open-loop response, $G_{ol}(\omega) = G(\omega)H(\omega)$ for the closed-loop compensatory tracking with the human operator as the feedback controller, $H(\omega)$,

$$G_{ol}(\omega) = G(\omega)H(\omega) = \frac{\omega_c}{j\omega} e^{-j\omega\tau_e}, \quad \text{near } \omega_c, \tag{2.5}$$

where, ω_c is the gain crossover frequency, τ_e is the effective time-delay that includes transport delays and high frequency neuromuscular lags [40]. Note that the adjustment rules of the crossover model are designed to more closely approximate the amplitude, rather than the phase, of the open-loop frequency response. The actual phase of the open-loop response is affected by unknown time-delays in the control loop, and is thus, not easy to closely match with a simple linear model.

The Analytic-Verbal model

The human feedback response, $H(\omega)$ that corresponds to the open-loop crossover model is given by,

$$H(\omega) = K_p \frac{(\tau_L j\omega + 1)}{(\tau_I j\omega + 1)} e^{-j\omega\tau_e} \tag{2.6}$$

where,

$$K_p = \text{ operator static gain}, \quad \tau_e = \text{ effective time delay}$$
$$\tau_L = \text{ lead time constant}, \quad \tau_I = \text{ lag time constant}$$

The human operator describing function, $H(\omega)$ in the above equation is called the *analytic-verbal* model, since it includes a set of *verbal* adjustment rules, obtained empirically by analyzing experimental data. These rules ensure that the corresponding open-loop frequency response, $G_{ol}(\omega) = G(\omega)H(\omega)$ matches the crossover model in Eq. (2.5) for different task variables, namely, the controlled system, G, and the bandwidth, ω_i of the reference signal, y_d. These rules are described below in brief (the reader is referred to [40] for further details):

- **Rule 1: (Equalization (τ_L, τ_I) selection and adjustment)** The particular equalization form is chosen from the general form $K_p(\tau_L j\omega + 1)/(\tau_I j\omega + 1)$ to achieve the following properties:

  (a) The controlled system, G can be stabilized by proper selection of K_p preferably over a broad range of frequencies.

  (b) Over a wide range of frequencies, near the crossover region, the magnitude of the open-loop frequency response, $|G_{ol}(\omega)| = |G(\omega)H(\omega)|$ has approximately a -20 dB/decade slope.

  (c) For low frequencies, $|G_{ol}(\omega)| = |G(\omega)H(\omega)| \gg 1$ to provide good low frequency closed-loop response and suppression of disturbances.

Table 2.1: Human feedback response describing functions for various types of controlled plants

Controlled element[2] $G(s)$[1]	**Equalization Characteristic** $K_p\frac{(\tau_L s+1)}{(\tau_I s+1)}$	**Equalization adjustment**	**Human describing function** $H(s)$[1]
K_c	$\frac{K_p}{\tau_I s+1}$	$\frac{1}{\tau_I} \ll \omega_c$	$\frac{K_p e^{-\tau_e s}}{\tau_I s+1}$
$\frac{K_c}{s}$	1	–	$K_p e^{-\tau_e s}$
$\frac{K_c}{s^2}$	$K_p(\tau_L s+1)$	$\frac{1}{\tau_L} \ll \omega_c$	$K_p(\tau_L s+1)e^{-\tau_e s}$
$\frac{K_c}{s(Ts+1)}$	if $T > \tau_e$: $K_p(\tau_L s+1)$ if $T < \tau_e$: K_p	$\frac{1}{\tau_L} = \frac{1}{T}$ –	$K_p(\tau_L s+1)e^{-\tau_e s}$ $K_p e^{-\tau_e s}$
$\frac{K_c}{\left(\frac{s}{\omega_n}\right)^2 + \frac{2\zeta}{\omega_n}s+1}$	if $\omega_n \ll 1/\tau_e$: $K_p(\tau_L s+1)$ if $\omega_n > 1/\tau_e$: $\frac{K_p}{\tau_I s+1}$	$\frac{1}{\tau_L} \ll \omega_c$ $\frac{1}{\tau_I} \ll \omega_c$	$K_p(\tau_L s+1)e^{-\tau_e s}$ $\frac{K_p e^{-\tau_e s}}{\tau_I s+1}$

[1] $s = j\omega$ where $j = \sqrt{-1}$ and ω is frequency in rad/s.

[2] Controlled system form in the crossover region. Crossover frequency, ω_c is defined as the frequency at which the magnitude of the open-loop transfer function, $G_{ol}(s) = G(s)H(s)$ crosses the 0 dB line. Crossover region is defined as the 2 decade interval centered around the crossover frequency, ω_c.

Examples of equalization selection and adjustment for some simple types of controlled systems, G are provided in Table 2.1, adapted from [40].

- **Rule 2: (Adjustment of crossover frequency, ω_c and effective time-delay, τ_e)** Once the equalization form is chosen as above, the human operator adjusts ω_c and τ_e to minimize the mean square error (in the low frequency region, i.e. $\omega_i \ll \omega_c$), within the stability limitation,

$$\phi_M = \frac{\pi}{2} - \tau_e \omega_c > 0. \tag{2.7}$$

Note that the specific values of ω_c and τ_e depend on the type of controlled system, G and the bandwidth of the reference signal, y_d, i.e. $\omega_c = \omega_c(G, \omega_i)$, and $\tau_e = \tau_e(G, \omega_i)$.

- **Rule 3: (Nominal crossover frequency, ω_{co})** By extrapolating experimentally obtained values of ω_c and τ_e for $\omega_i \to 0$, the nominal values for ω_{co} and τ_{eo} are related as,

$$\omega_{co} = \frac{\pi}{2\tau_o} \tag{2.8}$$

where, $\omega_{co} = \omega_{co}(G)$, and $\tau_o = \tau_o(G)$, for example see Table 2.2. Then, the effective

Table 2.2: Nominal values of ω_{co}, τ_{eo} and $\Delta\tau$ vs. G

G in crossover region	ω_{co} (rad/s)	τ_{eo} (seconds)	$\Delta\tau$ (seconds)
K_c	5 to 6	0.33	$0.070\ \omega_i$
K_c/s	4.3	0.36	$0.065\ \omega_i$
K_c/s^2	3.3	0.50	$0.065\ \omega_i$

time-delay, τ_e is given by,

$$\tau_e(G, \omega_i) = \tau_{eo}(G) - \Delta\tau(\omega_i) \tag{2.9}$$

where, $\Delta\tau \approx 0.07\omega_i$, obtained from experimental observations [40], see Table 2.2.

- **Rule 4: (ω_c invariance properties)** The crossover frequency, ω_c is known to have the following properties:

  (a) $\omega_c - K_c$ **independence:** A change in K_c is offset by a change in K_p by the human operator after initial adjustment of crossover frequency, resulting in a crossover frequency that is invariant to changes in controlled system static gain, K_c.

  (b) $\omega_c - \omega_i$ **independence:** ω_c is fairly constant for $\omega_i < 0.8\omega_c$

  (c) ω_c **regression:** For ω_i near or greater than $0.8\omega_{co}$, the crossover frequency, ω_c regresses to a value much smaller than ω_{co}.

- **Rule 5: (One-Third Rule)** For favorable tracking conditions, i.e. $\omega_i \ll \omega_c$, the tracking error, $e = y_d - y$ satisfies the following relation,

$$\frac{\overline{e}^2}{\overline{y}_d^2} = \frac{1}{3}\left(\frac{\omega_i}{\omega_c}\right)^2, \tag{2.10}$$

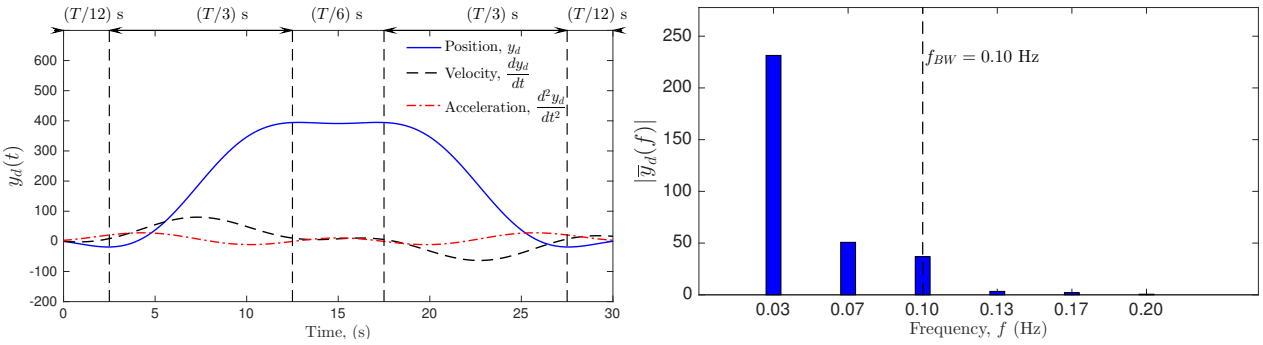
  where, $\overline{e}^2$ is the mean squared tracking error, $\overline{y}_d$ is the average value of the reference signal, y_d, and $\omega_i \ll \omega_c$. This relation is useful in scaling of displays and inputs, and in assessing the relative significance of alternative control loops employed by the human operator [40].

Additionally, a nonlinear remnant term, ρ is considered as an additive output noise to account for nonlinear behavior of the human operator, which becomes significant at higher frequencies or for difficult tracking conditions.

Using these verbal adjustment rules, a nominal model for human feedback response, $H(\omega)$, for the compensatory loop shown in Fig. 2.4, may be obtained for the given task variables (i.e. the controlled system, G and the bandwidth, ω_i of the reference signal, y_d).

The following section provides an example of how such a nominal model for the human-feedback response, $\hat{H}(\omega)$ can be estimated, given the model for the controlled system, G and the bandwidth of the reference signal to be tracked, ω_i.

Example: (Finding a Nominal Model, $\hat{H}(\omega)$)



(a) $y_d(t)$ in time domain

(b) $|\overline{y}_d(\omega)|$ in frequency domain

Figure 2.6: Reference trajectory to be tracked, y_d. (a) $y_d(t)$ in time domain (shown as a blue solid line) and its first time derivative (shown as a black dashed line) and second time derivative (shown as a red dash-dotted line), (b) Magnitude of Fourier spectrum, $\overline{y}_d(\omega)$.

The visuo-manual output tracking task, described in Appendix A, was used to study the human-feedback response for a simple controlled system, G,

$$G(s) = K_c, \qquad \text{where } s = j\omega, \quad \text{and} \quad j = \sqrt{-1}, \tag{2.11}$$

which is typically found in human-machine interaction, for example in input devices such as trackballs, automobile accelerator pedals, and video game joysticks [22]. Also, the reference output signal, $y_d(t)$ as shown in Fig. 2.6a, was considered. This represents a smooth reach

and retract motion, whose acceleration profile is given by,

$$\frac{d^2y_d}{dt^2} = \begin{cases} 0, & t \in [0, 0.5) \\ A^* \sin(2\pi f_{BW} t), & t \in [0.5, 0.5 + 1/f_{BW}) \\ 0, & t \in [0.5 + 1/f_{BW}, 1 + 1/f_{BW}) \\ -A^* \sin(2\pi f_{BW} t), & t \in [1 + 1/f_{BW}, 1 + 2/f_{BW}) \\ 0, & t \in [1 + 2/f_{BW}, 1.5 + 2/f_{BW}] \end{cases} \tag{2.12}$$

with amplitude, $A^* = A(2\pi f_{BW}^2)$, where A is the position amplitude, and $f_{BW} = \omega_i/(2\pi)$ is the bandwidth in Hz. In order to allow smooth pursuit tracking by the human user, the bandwidth of the reference trajectory, f_{BW} was chosen to be below the visual smooth-pursuit tracking limit of $f_T^* = 0.5$ Hz for typical healthy human users [8], i.e.

$$\{f_{BW} = 0.1 \text{ Hz}\} \leq \{f_T^* = 0.5 \text{ Hz.}\} \tag{2.13}$$

Fig. 2.6b depicts the magnitude of the Fourier spectrum of the signal, $|\overline{y}_d(\omega)|$. Note that only the first three harmonics are significant, and thus, the bandwidth of the signal can be approximated to $f_{BW} \approx 0.1$ Hz, or

$$\omega_i = 2\pi f_{BW} \approx 0.6 \text{ rad/s.} \tag{2.14}$$

For the chosen controlled system $G = K_c$, the form of the human-feedback response model is obtained from Table 2.1 as,

$$\hat{H}(s) = \frac{K_p e^{-\tau_e s}}{\tau_I s + 1}, \quad \text{where } s = j\omega, \text{ and } j = \sqrt{-1}, \tag{2.15}$$

where, the equalization adjustment rule is specified as, $\tau_I \omega_c \gg 1$. From Table 2.2, we obtain the nominal values for the open-loop gain crossover frequency, $\omega_{co} \approx 5$ rad/s, and the nominal effective time delay, τ_e, as

$$\tau_e = \tau_{eo} + \Delta\tau = 0.33 + 0.07\omega_i \approx 0.55 \text{ s} \tag{2.16}$$

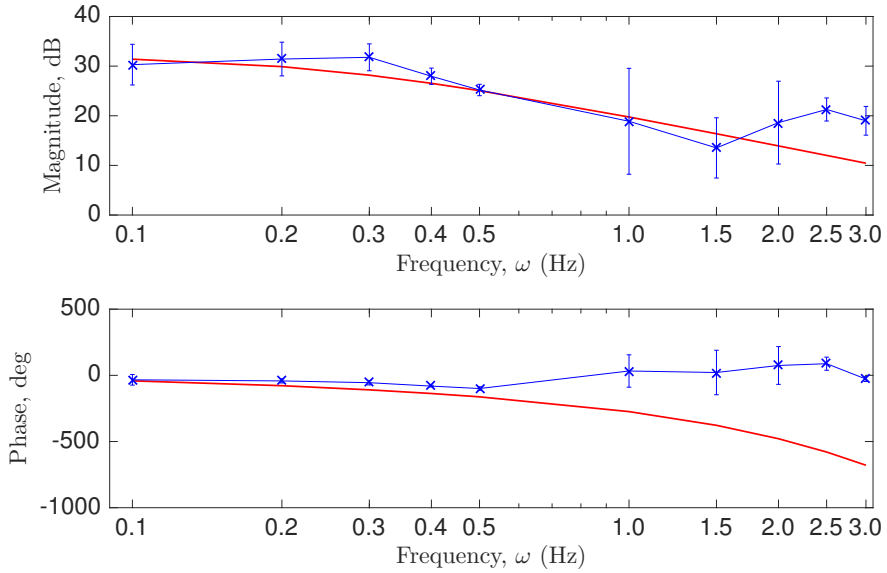


Figure 2.7: Human-feedback response, $H(\omega)$, measured experimentally (shown in blue) compared to the nominal model, $\hat{H}(\omega)$ (shown in red). Vertical lines indicate one standard deviation from the mean value for 10 trials for a single subject.

where, $\omega_i \approx 0.6$ rad/s is the bandwidth of the reference signal, y_d.

Next, to find the nominal value of τ_I, consider the magnitude of the open-loop frequency response at the gain crossover frequency, ω_c,

$$|G_{ol}(\omega)|_{\omega=\omega_c} = |G(\omega)H(\omega)|_{\omega=\omega_c} = \left|\frac{K_c K_p}{j\tau_I\omega + 1} e^{-j\tau_e\omega}\right|_{\omega=\omega_c} = \frac{K_c K_p}{\sqrt{1+{\tau_I}^2{\omega_c}^2}} = 1 \qquad (2.17)$$

which yields, for the equalization adjustment, $\tau_I\omega_c \gg 1$ (such that Rule 1 (b) is satisfied for the open-loop frequency response)

$$\frac{K_c K_p}{\tau_I \omega_c} = 1 \quad \Rightarrow \quad \tau_I = \frac{K_c K_p}{\omega_c} \qquad (2.18)$$

where, the value of K_p depends on the task conditions which can be specified, e.g. the gain of the input device used by the human operator. Thus, a nominal model for the human-feedback response, $\hat{H}(\omega)$ was determined for the given task variables, and is compared to the measured frequency response in Fig. 2.7.

Note that the verbal adjustment rules used to determine the nominal human-response model, $\hat{H}(\omega)$ are available only for certain simple types of controlled systems, G typically found in aircraft interfaces, for which the studies were carried out, for example see Table 2.1. Also, in the above discussion, the effect of time delays in the control loop was not discussed, which proves to be important in the current work. This was studied by Hess [26], and is described in the following section.

2.2.2 Effect of time-delays in the controlled system, G

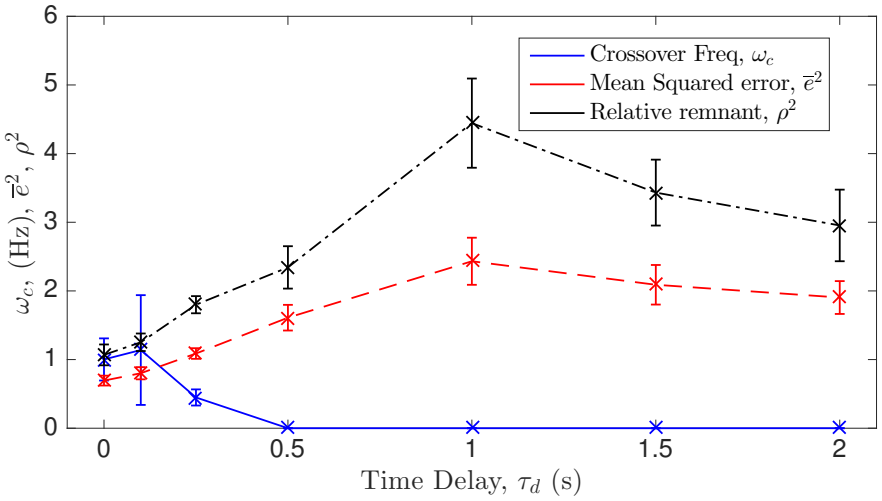


Figure 2.8: Effect of time-delay, τ_d on human operator's feedback performance. Vertical lines indicate one standard deviation from the mean for 10 trials.

The same visuo-manual tracking experiment, described in Appendix A, was used to study the effect of time delay in the control loop on the human-feedback response. The controlled system, G was chosen as,

$$G(s) = K_c e^{-\tau_d s}, \quad \text{where } s = j\omega, \text{ and } j = \sqrt{-1}, \tag{2.19}$$

where, τ_d is the pure input-output time-delay present in the controlled system, G. The time-delay was varied in the range, $\tau_d \in [0, 2]$ s to study the effect of increasing time-delay values in the human's feedback performance. The following observations were made:

- With increasing time-delay, τ_d, the human operator's tracking performance deteriorates, which can be seen from Fig. 2.8 where the mean squared tracking error, $\overline{e}^2$ increases. Note that for $\tau_d \geq 0.5$ s, the human operator is not able to track the reference signal, evidenced by the saturation of the mean-squared error after this point.

- Consequently, the open-loop gain crossover frequency, ω_c reduces until the human operator switches to a low-gain tracking strategy (with no open-loop crossover) in the range of time-delays for which the subject was not able to track.

- The relative remnant, ρ^2 was computed as,

$$\rho^2 = \sum_{i=1}^{N} \frac{|E(j\omega_i)|^2}{\overline{e}^2} \tag{2.20}$$

  where, $\overline{e}^2$ is the mean-squared tracking error given by,

$$\overline{e}^2 = \frac{1}{T} \int_0^T e^2(t)\, dt \tag{2.21}$$

  where, T is the time duration of each trial, and N is the number of frequencies considered in the error spectrum, $E(j\omega)$. From Fig. 2.8, it can be seen that the relative remnant, ρ^2 follows the trend displayed by the mean square error, $\overline{e}^2$, indicating that with increasing time-delay, the human operator employs greater amount of nonlinear characteristics (and also indicates that the task difficulty is increased).

The above results and insights match those described in the original work by Hess [26], where the analysis was only conducted for small values of time-delay, τ_d, typically found in aircraft systems. The current work extends these results for larger values of time-delays, which are

required in the proposed framework for the online implementation of preview-based stable inversion of nonminimum-phase systems [76].

2.2.3 Alternative human-operator models

The crossover model described in the previous section is one of several approaches available to model the human-operator's output tracking performance. A significant number of recent studies on human operator modeling are focused on models for understanding and emulating driver control behavior, due in part to the increasing interest in active passenger safety systems and autonomous vehicles [18, 15, 4, 34]. A wide range of driver models exist in literature that employ various modern control methods, ranging from optimal control [18], and PD controllers [63], to more complex models which use fuzzy logic [15] and Markov models [2]. There have also been advances in modeling intermittent control behavior of human operators to model sustained human motor control, for example in emulating human bipedal balance in humanoid robots [35].

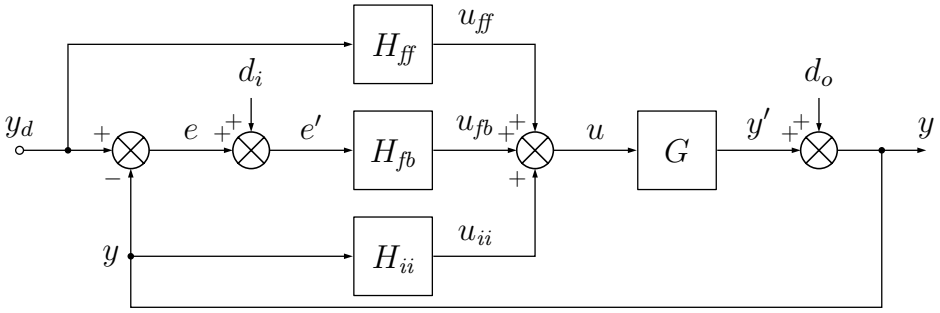


Figure 2.9: General block diagram of a human-in-the-loop pursuit tracking task, adapted from [72]. (a) H_{ff} refers to the feed-forward response of the human operator in response to the reference signal, y_d resulting in the feed-forward action, u_{ff}. (b) H_{fb} refers to the feedback response of the human operator in response to the error, e resulting in the feedback action, u_{fb}. (c) H_{ii} refers to the internal response of the human operator in response to the system output, y resulting in the control input, u_{ii}. The human control input, u_h is then a linear combination of these component loop signals, i.e. $u_h = u_{ff} + u_{fb} + u_{ii}$.

Mostly, such human performance models focus on representing only the feedback response characteristics of the human operator to unknown reference trajectories in an output tracking task. It has been suggested early on that the human operator "in-the-loop" may use additional control loops in performing a pursuit tracking task, such as a feed-forward loop, or an internal loop, in addition to the compensatory loop [72], see Fig. 2.9. Experimentally each of these control architectures have been identified using various model identification techniques, for example [5, 72, 24]. In [72], the authors hypothesized that an indirectly measured feed-forward response can be calculated by assuming that the feedback controller is the same as the McRuer's crossover model for compensatory tracking. Their results showed that the feed-forward response measured thus, is very close to the inverse of the controlled system. Another approach to identify both the feedback and feed-forward controllers is to add a disturbance signal as a second input [5, 24]. This allows the application of time-domain model identification methods to fit the observed time-domain data to parametric ARX (Auto-Regressive with eXogenous terms) models for the feedback and feed-forward controllers. The results from [24] showed that the feed-forward controller was similar to the inverse of the controlled system dynamics. In summary, such model identification methods are useful to characterize the human control behavior, but are not readily applicable to be used for prediction or inference of human intent in a human-in-the-loop shared control task. For this purpose, we require analytic or algorithmic representations of the alternative loops, especially the feed-forward path.

One of the first steps taken towards an algorithmic representation was by Kawato [30] with the Feedback Error Learning (FEL) model, in which it was postulated that the human control input has feedback and feed-forward components, see Fig. 2.10. It was initially presented from a biological perspective as a model of the learning process in the cerebellum through which an inverse model of the controlled system, $\hat{G}^{-1}$ is acquired in the feed-forward path. The feedback controller is a conventional feedback controller, for example a PID con-

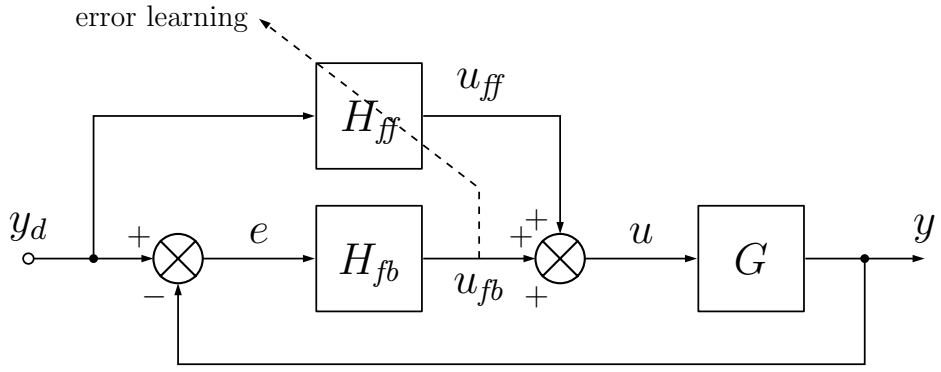


Figure 2.10: Feedback error learning model originally proposed by Kawato [30], where the human control input is postulated to increasingly shift to the feed-forward path from the feedback path, using the feedback error, u_{fb}.

troller, while the feed-forward inverse model is implemented as an adaptive controller, such as a neural network. The feedback input, u_{fb} acts as the input stimulus to the learning algorithm that updates the inverse model, for example see [53]. Note that the use of the nonlinear adaptive controller in the feed-forward path allows for incorporating potentially nonlinear control behavior by the human operator, implying better modeling accuracy compared to linear/quasi-linear models discussed previously. On the other hand, a good amount of training is required before the error learning algorithm converges, and over-fitting is a major concern in this scenario.

Another nonlinear approach to model the human operator in-the-loop is the use of fuzzy inference to update the model parameters of an ARX model, as shown in Fig. 2.11 [15]. A discrete-time ARX model may be specified as,

$$\begin{aligned} &y(t) + a_1 y(t-1) + a_2 y(t-2) + \cdots + a_{n_a} y(t-n_a) \\ &\quad = b_1 u(t-n_k) + b_2 u(t-n_k-1) + \cdots + b_{n_b} u(t-n_k-n_b+1) + e(t), \end{aligned} \tag{2.22}$$

for time step t, where e is the modeling error, y is the output, u is the input, a_i's and b_j's are model parameters to be estimated using training data, and n_a, n_b, n_k are the orders

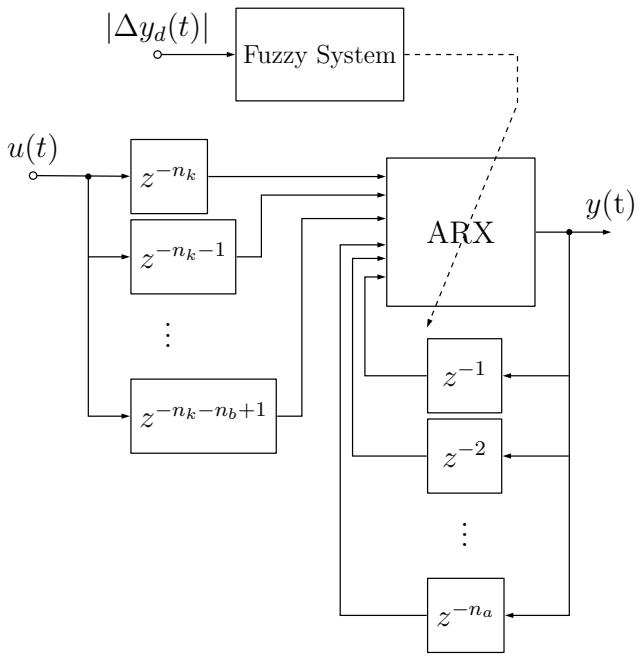


Figure 2.11: Fuzzy Auto-Regressive models with eXogenous inputs (F-ARX) model structure [15]. The fuzzy system updates the ARX model parameters based on changes to the reference input, y_d. The input to the fuzzy inference system is the absolute change in the reference signal $|y_d|$, and the output are the model parameters a_i, b_j in Eq.(2.22).

of the output, input, and the input-output delay, respectively. The fuzzy inference system provides an interpolation and smooth transition between infinite local linear ARX models, resulting in a nonlinear ARX model or a parameter varying ARX model. Thus, the model identification is a two-step process, wherein first the local linear models are identified based on each possible scenario, followed by specification of the rules that make up the fuzzy inference system to transition between these local models. The local models may be specified as a set of basis functions, for example Radial Basis Functions (RBFs) whose parameters may be estimated from training data using local linear regression schemes, for example Least Squares Estimation [15], and the fuzzy inference system may be obtained using an artificial neural

network (ANN) on training data collected from the closed-loop system. These models show promise to be used to predict human operator response, with the main advantage that the local ARX models can be modified based on the task nonlinearities, for example to account for dead zones [15]. As in the case of FEL, persistence of excitation (PE) is of concern in training the local linear models and the ANN-based fuzzy rules, implying the specification of large amount of training data for each possible of scenario.

2.2.4 Conclusion

This section described the crossover model for human-feedback response, that is useful in characterizing the successful compensatory tracking behavior of human operators. The crossover theory, on which the structural model of the human operator is based, was described in detail, with the major premise of the theory being that the human operator, when tracking successfully, has the characteristics of a good servo-mechanical feedback controller. The requirements of a good feedback controller were analyzed, and were shown to correspond to the verbal adjustment rules of the structural model of the human operator, i.e. the *analytic-verbal* model.

Further, human-in-the-loop compensatory tracking experiments were carried out to exemplify the validity of the *analytic-verbal* model. The nominal model, obtained using the verbal adjustment rules, was shown to fit the experimentally observed frequency response. In addition, the effect of time-delays in the controlled system perceived by the human operator was studied. The major effect of an increased time-delay was shown to be a deterioration in the human operator's tracking performance, as expected, and a reduction in the open-loop gain crossover frequency. Also, the nonlinear remnant was seen to increase with greater values of time-delay.

In conclusion, it is recognized that the human operator's tracking performance has limitations to the amount of acceptable time-delay in the control loop, which was also quantified

in this section. This conclusion proves to be useful in identifying the limitations of the proposed framework in the context of inverse control of nonminimum-phase systems, where the inversion of the nonminimum-phase system introduces time-delays which are dependent on the location of the nonminimum-phase zeros [76].

Finally, this chapter is concluded with a brief survey of some popular human modeling approaches in existing literature.

2.3 System model identification

Various techniques are available for fitting parametric/non-parametric models to observed input/output data, e.g., see [47] for a review with a focus on system identification. Non-parametric model estimation techniques formulate the problem of system identification as a function estimation problem in an infinite-dimensional space, in contrast to parametric prediction error such as using ARX (Auto-regressive with exogenous terms) [15]. Some of the more popular non-parametric techniques in recent years that have been used for system identification include Gaussian Process Regression (GPR), e.g., see [32], Support Vector Regression (SVR), e.g., see [55] and Locally Weighted Projection Regression (LWPR) e.g., see [62]. The work by [46] compares the above three non-parametric regression techniques in terms of modeling accuracy and computation costs, and concludes that GPR provides the most accuracy at the cost of more computation time. This additional computation cost is acceptable in this work since the human performance models are being trained offline, and then used to estimate the human intent by inversion online.

Additionally, the GPR framework provides an estimate of the modeling uncertainty of the computed model, which may be used to iteratively correct the estimated intent obtained by model-inversion, e.g., see [10]. In this work, both the nominal parametric and non-parametric model estimation methods are evaluated for modeling human-in-the-loop dynamics during teleoperation.

Chapter 3

ITERATIVE LEARNING OF HUMAN INTENT - FREQUENCY-DOMAIN

This chapter formulates the human-in-the-loop intent estimation problem as an iterative learning controller, whose convergence results in the output of the controlled system to converge to the intended goal trajectory. Further, stability and robustness of the proposed model-inversion based iterative learning controller are discussed in the presence of modeling uncertainty and/or output noise. Additionally, in situations where nominal parametric human models are no longer valid, for example with general linear controlled systems, an online inverse controller is presented that simplifies the controlled system as perceived by the human operator thereby allowing the use of the simpler nominal parametric human models for model-inversion based intent estimation. Finally, a data-based model update method is presented that enables faster intent prediction in the presence of sufficient training data.

3.1 Problem Formulation

The objective of the task is to achieve the intended goal trajectory, y_d as the output of the controlled system, G, i.e., $y = y_d$, as depicted in Fig. 3.1, where both the human operator (whose response dynamics are denoted by G_H) and the robot controller provide inputs u_h and u_c, respectively. In general, the human input, u_h is a combination of three terms ([72]), described in frequency domain by,

$$u_h(\omega) = \underbrace{G_{H,y_d}(\omega)y_d(\omega)}_{u_{h,ff}(\omega)} + \underbrace{G_{H,e}(\omega)e(\omega)}_{u_{h,fb}(\omega)} + \underbrace{G_{H,y}(\omega)y(\omega)}_{u_{h,y}(\omega)}, \tag{3.1}$$

where,

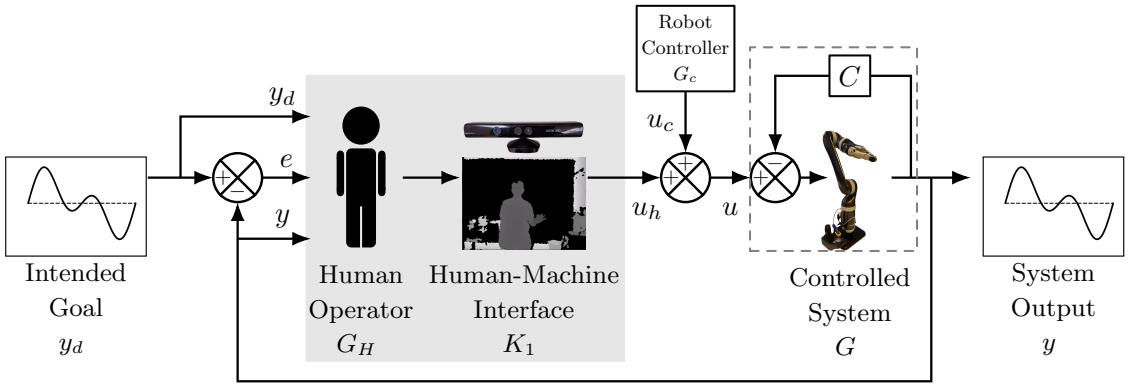


Figure 3.1: General block diagram of human-in-the-loop output tracking task; y_d is the desired output signal, y is the system output, $e = y_d - y$ is the perceived error, u_h is the human input, and u_c is the robot controller input.

$$G_{H,y_d}(\omega) = u_{h,ff}(\omega)/y_d(\omega) = K_1(\omega)H_{y_d}(\omega), \tag{3.2}$$

$$G_{H,e}(\omega) = u_{h,fb}(\omega)/e(\omega) = K_1(\omega)H_e(\omega), \tag{3.3}$$

$$G_{H,y}(\omega) = u_{h,y}(\omega)/y(\omega) = K_1(\omega)H_y(\omega), \tag{3.4}$$

where K_1 represents the human-machine interface, H_{y_d}, H_e, H_y are the feed-forward, feedback and internal response transfer functions of the human operator's dynamics, respectively. The resulting system output, y may be described in frequency domain by,

$$\begin{aligned} y(\omega) &= G(\omega)(u_c(\omega) + u_h(\omega)) \\ &= G_{fb}(\omega)y_d(\omega) + G_{ff}(\omega)u_c(\omega) \end{aligned} \tag{3.5}$$

where the feedback, G_{fb} and feed-forward, G_{ff} transfer functions are given by,

$$G_{fb}(\omega) = \frac{(G_{H,y_d}(\omega) + G_{H,e}(\omega))G(\omega)}{1 + (G_{H,e}(\omega) - G_{H,y}(\omega))G(\omega)} = \frac{G_{H,1}(\omega)G(\omega)}{1 + G_{H,2}(\omega)G(\omega)}, \tag{3.6}$$

$$G_{ff}(\omega) = \frac{G(\omega)}{1 + (G_{H,e}(\omega) - G_{H,y}(\omega))G(\omega)} = \frac{G(\omega)}{1 + G_{H,2}(\omega)G(\omega)}, \tag{3.7}$$

respectively, all expressed as frequency response functions, i.e.,

$$G(\omega) = G(s)|_{s=j\omega}, \quad \text{where } j = \sqrt{-1}. \tag{3.8}$$

Note that the feedback and feed-forward terms in Eqs. (3.6) and (3.7) are related as,

$$G_{fb}(\omega) = G_{H,1}(\omega)G_{ff}(\omega), \tag{3.9}$$

where,

$$\begin{aligned} G_{H,1}(\omega) &= G_{H,e}(\omega) + G_{H,y_d}(\omega), \\ G_{H,2}(\omega) &= G_{H,e}(\omega) - G_{H,y}(\omega). \end{aligned} \tag{3.10}$$

Using (3.10) in (3.1), the human input, u_h may be expressed as,

$$u_h(\omega) = G_{H,1}(\omega)y_d(\omega) - G_{H,2}(\omega)y(\omega). \tag{3.11}$$

Problem statement: Learn the control input, u_c^* that exactly tracks the desired output trajectory, $y = y_d$ when the robot controller is subject to the following two constraints:

Constraint 1

(access to desired output): The desired output trajectory, y_d is not available to the robot controller, but it has direct access to the control input from the human operator, u_h, and the system output, y.

Constraint 2

(modeling uncertainty): The robot controller does not have exact information of the feedback transfer function, G_{fb} but has access to a model, $\hat{G}_{fb}$.

The control input, $u = u_h + u_c = u^*$ required for exact tracking, $y = y_d$ for the controlled system G is given by (3.5) as,

$$u^*(\omega) = u_h^*(\omega) + u_c^*(\omega) = G^{-1}(\omega)y_d(\omega). \tag{3.12}$$

If the human user is an expert at the task, then the human input u_h can lead to perfect tracking, i.e., $y = y_d$. This implies, the robot controller can directly use the human input, $u_h \approx u^*$ (no longer requiring the human input) to achieve perfect tracking. This is the basis of imitation learning approaches, for example ([54]). However, if the human user is not an expert or if the system is complex, then $u_h \neq u^*$ resulting in reduced tracking accuracy. If, instead, the robot controller has access to the desired trajectory, y_d and the system model G, then it does not require the human input, but can choose the control input, $u_c = u^* = G^{-1}y_d$, to achieve perfect tracking. Moreover, Iterative Learning Control (ILC) may be used in the presence of modeling errors [7]. But, none of this is possible if the desired trajectory, y_d is not available to the robot controller, as specified by Constraint 1.

Thus, the human operator's intended goal, y_d must be determined using the available information, i.e., the system output, y and the human input u_h.

3.2 Estimating intent by model inversion

Human intent in the current setup is the desired output trajectory, $y = y_d$, that the human operator is trying to track with the system, G, as shown in Fig. 3.1. Then, the intended goal, y_d may be inferred from (3.11) as,

$$y_d(\omega) = G_{H,1}^{-1}(\omega)\left[u_h(\omega) + G_{H,2}(\omega)y(\omega)\right]. \tag{3.13}$$

forming the intent estimate, $\hat{y}_d$ given by,

$$\hat{y}_d(\omega) = \hat{G}_{H,1}^{-1}(\omega)\left[u_h(\omega) + \hat{G}_{H,2}(\omega)y(\omega)\right], \tag{3.14}$$

where,

$$\begin{aligned} \hat{G}_{H,1}(\omega) &= \hat{G}_{H,e}(\omega) + \hat{G}_{H,y_d}(\omega), \\ \hat{G}_{H,2}(\omega) &= \hat{G}_{H,e}(\omega) - \hat{G}_{H,y}(\omega), \end{aligned} \tag{3.15}$$

where $\hat{G}_{H,y_d}, \hat{G}_{H,e}$, and $\hat{G}_{H,y}$ are known models of the human response transfer functions defined in Eqs. (3.2)-(3.4). Section 2.2.1 described nominal models for $G_{H,e}$, e.g., the crossover-

based analytic-verbal model.These models may then be inverted to obtain the intent estimate as in (3.14), but when there is modeling uncertainty, or high model estimation variance, the inversion approach will result in reduced tracking precision. The following sections describe an iterative method to estimate the controller input which in combination with the human input may be used to improve the estimated intent obtained by model inversion.

3.2.1 Learning the controller input using estimated intent

Consider the system output at the k^{th} iteration, y_k, which is obtained from (3.5) as,

$$\begin{aligned} y_k(\omega) &= G(\omega)(u_{c,k}(\omega) + u_{h,k}(\omega)) \\ &= G_{fb}(\omega)y_d(\omega) + G_{ff}(\omega)u_{c,k}(\omega) \end{aligned} \tag{3.16}$$

where the feedback, G_{fb} and feed-forward, G_{ff} transfer functions are given by (3.6) and (3.7), respectively.

Thus, the tracking error at the k^{th} iteration is given by,

$$\begin{aligned} e_k(\omega) &= y_d(\omega) - y_k(\omega) \\ &= (1 - G_{fb}(\omega))\, y_d(\omega) - G_{ff}(\omega)u_{c,k}(\omega). \end{aligned} \tag{3.17}$$

Taking the difference of successive iterations, we have,

$$e_{k+1}(\omega) - e_k(\omega) = G_{ff}(\omega)\left(u_{c,k}(\omega) - u_{c,k+1}(\omega)\right). \tag{3.18}$$

Thus, to obtain perfect tracking in the $(k+1)^{th}$ iteration, i.e., for $e_{k+1}(\omega) = 0$, we must have the following,

$$\begin{aligned} u_{c,k+1}(\omega) &= u_{c,k}(\omega) + G_{ff}^{-1}(\omega)e_k(\omega), \\ &= u_{c,k}(\omega) + G_{ff}^{-1}(\omega)\left[y_d(\omega) - y_k(\omega)\right] \\ &= u_{c,k}(\omega) + \frac{G_{H,1}(\omega)}{G_{fb}(\omega)}\left[y_d(\omega) - y_k(\omega)\right]. \end{aligned} \tag{3.19}$$

Using (3.13) and (3.9) in the above yields, (suppressing ω for clarity)

$$u_{c,k+1} \;=\; u_{c,k} \;+\; G_{fb}^{-1}u_{h,k} \;-\; G_{fb}^{-1}\left(G_{H,1} - G_{H,2}\right)y_k. \tag{3.20}$$

By Constraint 2, since exact models are not available, the update law in (3.20) must be modified as,

$$u_{c,k+1} \;=\; u_{c,k} \;+\; \rho\hat{G}_{fb}^{-1}\left[u_{h,k} - \left(\hat{G}_{H,1} - \hat{G}_{H,2}\right) y_k\right], \tag{3.21}$$

where $\rho(\omega)$ is the iteration gain that needs to be chosen to ensure convergence of the iterative scheme, and the initial input is considered to be zero, i.e., $u_{c,0}(\omega) = 0$ without loss of generality, and,

$$\hat{G}_{fb}(\omega) = \frac{\hat{G}_{H,1}(\omega)\hat{G}(\omega)}{1 + \hat{G}_{H,2}(\omega)\hat{G}(\omega)}, \tag{3.22}$$

where, $\hat{G}$ is a known model of the controlled system, G, and $\hat{G}_{H,1}, \hat{G}_{H,2}$ are given by (3.15).

3.2.2 Convergence Analysis: ILC

This section presents the convergence analysis using the general 3-channel human dynamics model, discussed in the previous section. It extends the results for the compensatory tracking convergence analysis presented in [69], and the connection to those results are presented.

Robustness to modeling uncertainty

Convergence of the iterative update law in (3.21) is proved under the following three assumptions.

Assumption 1 (System and model properties)**.** *The feedback transfer function, G_{fb} and its model $\hat{G}_{fb}$ have hyperbolic zero dynamics, i.e., real part of all zeros are non-zero, and are also non-trivial, i.e., $G_{fb}(\omega) \neq 0$ and $\hat{G}_{fb}(\omega) \neq 0$. This implies, $G_{H,1}(\omega) \neq 0$ and $\hat{G}_{H,1}(\omega) \neq 0$ from Eqs.* (3.6) *and* (3.22)*.*

Assumption 2 (Stable human-in-the-loop system)**.** *The human operator's dynamics, $G_H = [G_{H,y_d}, G_{H,e}, G_{H,y}]$ acts as a stabilizing controller, e.g., [31, 43, 56], even when perfect tracking is not achieved, which ensures that the overall closed-loop system, G_{fb} and its model, $\hat{G}_{fb}$ are stable.*

Assumption 3 (Desired output)**.** *The desired output trajectory, y_d is sufficiently smooth in the time domain to ensure that an inverse input exists for exact output tracking with the controlled system, G [21].*

Lemma 1 (Convergence)**.** *Under Assumptions 1-3, the iterative update scheme in* (3.21) *results in output convergence, point-wise at each frequency, i.e.,*

$$\lim_{k\to\infty} y_k(\omega) \to y_d(\omega), \tag{3.23}$$

if and only if the magnitude of the phase error, $\Delta_\theta(\omega)$ and the iteration gain, $\rho(\omega)$ are sufficiently small, i.e.,

$$|\Delta_\theta(\omega)| < \pi/2, \tag{3.24}$$

$$0 < \rho(\omega) < \frac{2\cos\left(\Delta_\theta(\omega)\right)}{\Delta_m(\omega)} = \rho^*(\omega), \tag{3.25}$$

where,

$$\begin{aligned}\Delta(\omega) &= \Delta_m(\omega)e^{j\Delta_\theta(\omega)} \\ &= \frac{G_{fb}(\omega)}{\hat{G}_{fb}(\omega)}\left[1+\left(\frac{G_{H,2}(\omega)-\hat{G}_{H,2}(\omega)}{G_{H,1}(\omega)}\right)-\left(\frac{G_{H,1}(\omega)-\hat{G}_{H,1}(\omega)}{G_{H,1}(\omega)}\right)\right].\end{aligned} \tag{3.26}$$

Moreover, with increasing iteration steps, k, the human operator's control input, $u_{h,k}$ tends to zero, i.e.,

$$\lim_{k\to\infty} u_{h,k}(\omega) = 0, \tag{3.27}$$

and, the controller input, $u_{c,k}$ tends to the exact inverse input, i.e.,

$$\lim_{k\to\infty} u_{c,k}(\omega) = u^*(\omega) = G^{-1}(\omega)y_d(\omega). \tag{3.28}$$

Proof. Subtracting the (3.21) for the $(k+1)^{th}$ iteration step with the k^{th} iteration step, we

have,

$$\begin{aligned} u_{c,k+1}(\omega) - u_{c,k}(\omega) &= u_{c,k}(\omega) - u_{c,k-1}(\omega) \\ &\quad + \frac{\rho(\omega)}{\hat{G}_{fb}(\omega)} \left(u_{h,k}(\omega) - u_{h,k-1}(\omega)\right) \\ &\quad - \frac{\rho(\omega)\left(\hat{G}_{H,1}(\omega) - \hat{G}_{H,2}(\omega)\right)}{\hat{G}_{fb}(\omega)} \left(y_k(\omega) - y_{k-1}(\omega)\right) \end{aligned}$$

using (3.11) **we have,** (3.29)

$$\begin{aligned} &= u_{c,k}(\omega) - u_{c,k-1}(\omega) \\ &\quad - \frac{\rho(\omega) G_{H,2}(\omega)}{\hat{G}_{fb}(\omega)} \left(y_k(\omega) - y_{k-1}(\omega)\right) \\ &\quad - \frac{\rho(\omega)\left(\hat{G}_{H,1}(\omega) - \hat{G}_{H,2}(\omega)\right)}{\hat{G}_{fb}(\omega)} \left(y_k(\omega) - y_{k-1}(\omega)\right). \end{aligned}$$

Using (3.16) in the above, we have,

$$u_{c,k+1}(\omega) - u_{c,k}(\omega) = \gamma(\omega)\left(u_{c,k}(\omega) - u_{c,k-1}(\omega)\right), \tag{3.30}$$

where,

$$\gamma(\omega) = 1 - \rho(\omega)\frac{G_{ff}(\omega)}{\hat{G}_{fb}(\omega)}\left[G_{H,2}(\omega) + \hat{G}_{H,1}(\omega) - \hat{G}_{H,2}(\omega)\right], \tag{3.31}$$

which can be written in terms of the modeling error Δ defined in (3.26), using (3.9) and rearranging terms,

$$\gamma(\omega) = 1 - \rho(\omega)\Delta(\omega). \tag{3.32}$$

Applying (3.30) successively, results in

$$u_{c,k+1}(\omega) - u_{c,k}(\omega) = \gamma^k(\omega)\left(u_{c,1}(\omega) - u_{c,0}(\omega)\right). \tag{3.33}$$

Thus, the iterative update scheme in (3.21) converges, i.e.,

$$\lim_{k\to\infty}\left(u_{c,k+1}(\omega) - u_{c,k}(\omega)\right) = 0, \tag{3.34}$$

if and only if the magnitude of the mapping coefficient, γ (defined in (3.32)) is less than one, i.e.,

$$|\gamma(\omega)| < 1, \tag{3.35}$$

or, equivalently, the square of the magnitude of the mapping coefficient γ,

$$\begin{aligned} |\gamma(\omega)|^2 &= |1 - \rho(\omega)\Delta(\omega)|^2 \\ &\textbf{using } (3.26)\textbf{, we have,} \\ &= \left|1 - \rho(\omega)\Delta_m(\omega)e^{j\Delta_\theta(\omega)}\right|^2 \\ &= [1 - \rho(\omega)\Delta_m(\omega)\cos(\Delta_\theta(\omega))]^2 \\ &\quad + [\rho(\omega)\Delta_m(\omega)\sin(\Delta_\theta(\omega))]^2 \\ &= 1 + \rho^2(\omega)\Delta_m^2(\omega) \\ &\quad - 2\rho(\omega)\Delta_m(\omega)\cos(\Delta_\theta(\omega)), \end{aligned} \tag{3.36}$$

must be less than one, i.e.,

$$1 + \rho^2(\omega)\Delta_m^2(\omega) - 2\rho(\omega)\Delta_m(\omega)\cos(\Delta_\theta(\omega)) < 1, \tag{3.37}$$

or,

$$\rho(\omega)\left[\rho(\omega)\Delta_m(\omega) - 2\cos(\Delta_\theta(\omega))\right] < 0. \tag{3.38}$$

The modeling error Δ_m is nonzero,

$$\Delta_m(\omega) = \left|\frac{G_{fb}(\omega)}{\hat{G}_{fb}(\omega)}\left[1 + \left(\frac{G_{H,2}(\omega) - \hat{G}_{H,2}(\omega)}{G_{H,1}(\omega)}\right) - \left(\frac{G_{H,1}(\omega) - \hat{G}_{H,1}(\omega)}{G_{H,1}(\omega)}\right)\right]\right| > 0, \tag{3.39}$$

since the model $\hat{G}_{fb}$ is assumed to be stable by Assumption 2,

$$\left|\hat{G}_{fb}(\omega)\right| < \infty, \tag{3.40}$$

and from Assumption 1,

$$|G_{fb}(\omega)| \neq 0. \tag{3.41}$$

Therefore, (3.38) is satisfied if and only if:

1. $\cos(\Delta_\theta(\omega)) \neq 0$ i.e., $\Delta_\theta(\omega) \neq \pi/2$ or its odd multiples, and,

2. the iteration gain $\rho(\omega)$ satisfies

$$\begin{aligned} 0 < \rho(\omega) < \frac{2\cos(\Delta_\theta(\omega))}{\Delta_m(\omega)}, \quad & \text{for } \cos(\Delta_\theta(\omega)) > 0, \\ \frac{2\cos(\Delta_\theta(\omega))}{\Delta_m(\omega)} < \rho(\omega) < 0, \quad & \text{for } \cos(\Delta_\theta(\omega)) < 0. \end{aligned} \tag{3.42}$$

Now, similar to the arguments in the convergence proof in [58], since the actual system dynamics are unknown, a phase variation of magnitude greater than $\pi/2$ will result in a sign change in $\cos(\Delta_\theta(\omega))$. If this sign change is known beforehand, or it can be determined experimentally, it can be accommodated in sign changes in $\rho(\omega)$. But, if this sign change cannot be obtained *a priori*, then it becomes necessary for the magnitude of the phase variation to be less than $\pi/2$ to ensure convergence of the iterations with a fixed iteration gain $\rho(\omega)$. This results in the convergence conditions outlined in the Lemma, (3.24), (3.25).

Further, when the iterations converge, the fixed point, $u_c^*(\omega) = \lim_{k\to\infty} u_{c,k}(\omega)$ applied to (3.19), results in,

$$\begin{aligned} u_c^*(\omega) &= u_c^*(\omega) + \rho(\omega)\frac{G_{H,1}(\omega)}{\hat{G}_{fb}(\omega)}(y_d(\omega) - \lim_{k\to\infty} y_k(\omega)) \\ \text{or,} \quad \lim_{k\to\infty} y_k(\omega) &= y_d(\omega) \end{aligned} \tag{3.43}$$

If the tracking error e is zero, then the input from the human is zero, i.e., $u_h = 0$, and therefore, the exact-tracking controller input, u_c is the inverse as in Lemma 1, which completes the proof. □

Remark 1 (Avoid iterations under large uncertainty)**.** *Most models tend to have large uncertainties in some frequencies, e.g., at high frequencies, which might be difficult to reduce. In this case the iteration gain, $\rho(\omega)$ can be chosen to be small (or zero) at those frequencies to trade-off the tracking and uncertainty in an optimal manner, e.g., [17]. For example, iterations can be performed only in the region where the modeling uncertainty is sufficiently small.*

Remark 2 (Compensatory Tracking)**.** *If the human operator dynamics are dominated by the compensatory feedback channel $G_{H,e}$, defined in* (3.3)*, then the transfer functions $G_{H,1}$ and $G_{H,2}$ defined in* (3.10) *are given by,*

$$G_{H,1}(\omega) = G_{H,2}(\omega) = G_{H,e}(\omega), \tag{3.44}$$

which implies G_{fb} defined in (3.6) *is given by,*

$$G_{fb}(\omega) = \frac{G_{H,e}(\omega)G(\omega)}{1 + G_{H,e}(\omega)G(\omega)}, \tag{3.45}$$

and, the modeling uncertainty in the convergence condition in Lemma 1 is given by,

$$\Delta(\omega) = \Delta_m(\omega)e^{j\Delta_\theta(\omega)} = \frac{G_{fb}(\omega)}{\hat{G}_{fb}(\omega)}, \tag{3.46}$$

Convergence analysis under the conditions with dominant compensatory tracking was presented in [69].

Robustness to output noise

Noise can be present in the system output, y_k at every iteration k, and can potentially build up with iterations. The overall output noise can be represented as a bulk noise term N_k (that is uncorrelated with the signals y_d and $u_{c,k}$ at each iteration step k) that modifies the output at the k^{th} iteration step in (3.16) to

$$y_k(\omega) = N_k(\omega) + G_{fb}(\omega)y_d(\omega) + G_{ff}(\omega)u_{c,k}(\omega). \tag{3.47}$$

The effect of this noise can be bounded if the noise itself is bounded based on Lemma 2 in [58], and is shown below for the proposed iterative update law in (3.21).

Lemma 2 (Output noise)**.** *Under conditions for Lemma 1, if the output noise N_k in* (3.47) *is bounded for all integers $k \geq 0$,*

$$|N_k(\omega)| \leq \overline{N}(\omega), \tag{3.48}$$

where $\overline{N}(\omega)$ *is a real-valued scalar, then the error in the output with the iteration update law in* (3.21) *due to the noise is bounded as*

$$\limsup_{k\to\infty} |y_k(\omega) - y_d(\omega)| \leq \frac{2\overline{N}(\omega)}{1-\overline{W}(\omega)}, \tag{3.49}$$

where $\overline{W}(\omega)$ *is the upper bound on the magnitude of the overall iteration gain,* $\gamma(\omega)$ *in* (3.31) *given by*

$$|\gamma(\omega)| = |1 - \rho(\omega)\Delta(\omega)| \leq \overline{W}(\omega) < 1 \tag{3.50}$$

for iteration gain $\rho(\omega)$ *from Lemma 1.*

Proof. Consider the difference between consecutive outputs in (3.47),

$$\begin{aligned} y_{k+1}(\omega) = y_k(\omega) &+ G_{ff}(\omega)[u_{c,k+1}(\omega) - u_{c,k}(\omega)] \\ &+ [N_{k+1}(\omega) - N_k(\omega)], \end{aligned} \tag{3.51}$$

which can be rewritten using Eqs. (3.19), (3.26), and the definition of the overall iteration gain $\gamma(\omega)$ from (3.31), as

$$\begin{aligned} y_{k+1}(\omega) = y_k(\omega) &+ (1-\gamma(\omega))[y_d(\omega) - y_k(\omega)] \\ &+ [N_{k+1}(\omega) - N_k(\omega)]. \end{aligned} \tag{3.52}$$

Subtracting the desired output, y_d from both sides of (3.52), using the bound $\overline{W}$ from (3.50), and using the bound on the noise terms from (3.48), yields

$$\begin{aligned} |y_{k+1}(\omega) - y_d(\omega)| &\leq |\overline{W}(\omega)||y_k(\omega) - y_d(\omega)| + 2|\overline{N}(\omega)| \\ &\leq |\overline{W}(\omega)|^k |y_1(\omega) - y_d(\omega)| \\ &\quad + 2|\overline{N}(\omega)| \sum_{m=1}^{k} \left[\overline{W}(\omega)\right]^{m-1} \end{aligned} \tag{3.53}$$

and since $|\overline{W}(\omega)| < 1$ from (3.50),

$$\limsup_{k\to\infty} |y_{k+1}(\omega) - y_d(\omega)| \leq \frac{2|\overline{N}(\omega)|}{1-\overline{W}(\omega)}, \tag{3.54}$$

which completes the proof. □

Remark 3 (Initial conditions)**.** *Bounded, initial condition error (at the start of each iteration k) will result in a bounded error in the output due to stability of the systems from Assumption 1. Therefore, the effect of initial condition errors, which would be small if sufficient time is available between iterations, can be evaluated by considering it as an output noise as in Lemma 2.*

3.3 ILC with Simple Controlled Systems

In this case, the human-in-the-loop response is assumed to be dominated by the compensatory feedback channel $G_{H,e}$, i.e., $G_{H,y_d} = G_{H,y} = 0$ in (3.1) in which case the iterative update law in (3.21) is modified as,

$$u_{c,k+1}(\omega) = u_{c,k}(\omega) + \rho(\omega)\hat{G}_{fb}^{-1}(\omega)u_{h,k}(\omega), \tag{3.55}$$

where $\hat{G}_{fb}$ is the model of the feedback transfer function G_{fb} given by,

$$G_{fb}(\omega) = \frac{G(\omega)G_{H,e}(\omega)}{1 + G(\omega)G_{H,e}(\omega)} \tag{3.56}$$

The compensatory tracking iterative update law in (3.55) was validated using a human-in-the-loop output tracking experiment, as shown in Fig. 3.2

3.3.1 Apparatus

The system output, y and the reference trajectory, y_d were observed by the human through a graphical user interface (GUI) as shown in Fig. 3.2. The circular cross-hair followed the system output, y while the red cross followed the reference trajectory, y_d. The human user, estimated the error, $e = y_d - y$ in the output, and provided the input, u_h through a human-machine interface (HMI), in this case a Microsoft Kinect camera, by moving his/her right

---

This section is derived from the publication: Rahul B Warrier and Santosh Devasia. "Iterative learning from novice human demonstrations for output tracking". In: *IEEE Transactions on Human-Machine Systems* 46.4 (2016), pp. 510–521

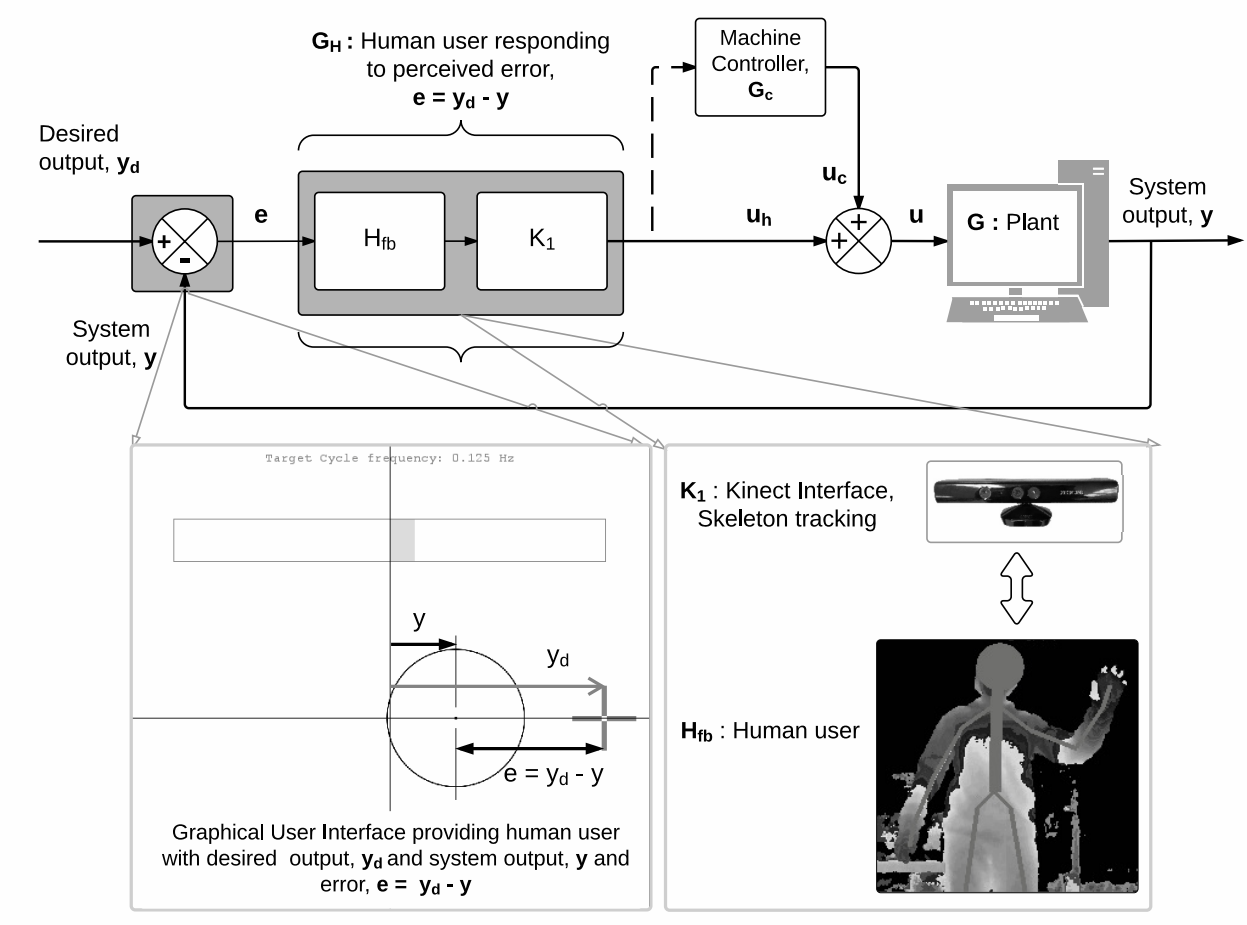


Figure 3.2: Experimental setup for human-in-the-loop experiment. Solid black line depicts continuous flow of information; dashed line depicts intermittent flow of information (cycle preceding controller input update, $u_{c,k}$).

hand from side to side. The trajectory of the right hand was then tracked using the Skeleton tracking routine provided in the Kinect SDK [73]. The HMI then computed the difference between the tracked $x-$coordinate of the right hand from the origin of the Kinect frame of reference, which was then multiplied by a gain, K_1 to obtain the human input, u_h. The gain, K_1 was selected such that maximum range of human input was limited to arm motions without the need for large whole-body movements by the human user. The human input, u_h was added to the controller input, u_c to obtain the total input,

$$u = u_h + u_c, \tag{3.57}$$

that was applied to the system, G. The system, G was simulated in a computer to enable potential experimentation with different types of systems, but, could be replaced with a physical system. Note that in this setup, the human had access to the desired output, y_d and the system output, y, while the controller G_c did not have access to y_d, as specified in Constraint 1. Also, the controller, G_c had access to the model of the closed-loop system, $\hat{G}_{fb}$, to meet Constraint 2.

3.3.2 Participants

Nine subjects, 8 male and 1 female, were recruited as volunteers for the study[1]. The subjects were healthy individuals with age ranging from 20-50 years, with no professional athletic training.

3.3.3 Apparatus

The system output, y and the reference trajectory, y_d were observed by the human through a graphical user interface (GUI) as shown in Fig. 3.2. The circular cross-hair followed the

[1]This work was reviewed and approved to conduct multiple human subject trials, under application no. 50095, by the Human Subjects Division Internal Review Board at the University of Washington

system output, y while the red cross followed the reference trajectory, y_d. The human user, estimated the error, $e = y_d - y$ in the output, and provided the input, u_h through a human-machine interface (HMI), in this case a Microsoft Kinect camera, by moving his/her right hand from side to side. The trajectory of the right hand was then tracked using the Skeleton tracking routine provided in the Kinect SDK [73]. The HMI then computed the difference between the tracked $x-$coordinate of the right hand from the origin of the Kinect frame of reference, which was then multiplied by a gain, K_1 to obtain the human input, u_h. The gain, K_1 was selected such that maximum range of human input was limited to arm motions without the need for large whole-body movements by the human user. The human input, u_h was added to the controller input, u_c to obtain the total input,

$$u = u_h + u_c, \tag{3.58}$$

that was applied to the system, G. The system, G was simulated in a computer to enable potential experimentation with different types of systems, but, could be replaced with a physical system. Note that in this setup, the human had access to the desired output, y_d and the system output, y, while the controller G_c did not have access to y_d, as specified in Constraint 1. Also, the controller, G_c had access to the model of the closed-loop system, $\hat{G}_{fb}$, to meet Constraint 2.

3.3.4 Procedure

System definitions

The analytic-verbal model [41] of the form,

$$\hat{G}_H(s) = \frac{U_H(s)}{E(s)} = K_p \frac{1}{\tau_l s + 1} e^{-\tau_d s}, \tag{3.59}$$

was used to capture the human motor response for manual control tasks. This model, based on the crossover model from [11, 39], has been used in the past with second-order systems,

Table 3.1: Parameter Values

Parameter	Value	Units	Description
Plant, $G(s)$			
K_{gain}	1.0	–	Gain
ζ	1.0	–	Damping ratio
f_n	0.25	Hz	Natural frequency
Plant Model, $\hat{G}(s)$ **(10% error)**			
$\hat{K}_{gain}$	1.1	–	Gain
$\hat{\zeta}$	1.1	–	Damping ratio
$\hat{f}_n$	0.275	Hz	Natural frequency
Fitted Human Response model, $\hat{G}_H(s)$ **(for Subject 2)**			
K_p	114.6	-	DC Gain
τ_d	0.1	s	Effective time delay
τ_l	3.8	s^{-1}	Equalization time constant
Nominal Human Response model, $\hat{G}_H(s)$ **(for Subject 2)**			
K_p	24.5	-	DC Gain
τ_d	0.2	s	Effective time delay
τ_l	0.35	s^{-1}	Equalization time constant
Desired Trajectory			
f_T	$\alpha \times f_n$	Hz	Target frequency, $\alpha = 0.5, 1.0$
A	600	m	Amplitude
Experiment parameters			
k_1	10	cycles	Human only control
k_2	10	cycles	Shared control
k_3	1	cycles	Machine only control

G of the form,

$$G(s) = \frac{y(s)}{u(s)} = \frac{K_{gain}}{s^2 + 2\zeta\omega_n s + \omega_n^2}. \tag{3.60}$$

Therefore, a second-order system, G was used in this work — the parameters used in the experiments are provided in Table 3.1. The resonance frequency, f_n in Hertz, with $\omega_n = 2\pi f_n$, of the critically damped system, G was chosen to be below the smooth-pursuit visual tracking limit of $f_T^* = 0.5$ Hz for typical healthy human users [8]. Such linear time-invariant second order systems, G have been used previously in literature to model human machine interactions, e.g., for vehicle steering angle control [25], and for modeling human arm movements when constrained by active robotic systems [23, 27].

Desired trajectory, y_d

The desired trajectory, y_d for this visuo-manual tracking task was chosen to be a reach and retract movement as shown in Fig. 3.3. The acceleration profile of the desired trajectory was defined as, (with time, t in seconds),

$$\frac{d^2 y_d}{dt^2}(t) = \begin{cases} 0, & t \in [0, 0.5) \\ A^* \sin(2\pi f_T t), & t \in [0.5, 0.5 + \frac{1}{f_T}) \\ 0, & t \in [0.5 + \frac{1}{f_T}, 1 + \frac{1}{f_T}) \\ -A^* \sin(2\pi f_T t), & t \in [1 + \frac{1}{f_T}, 1 + \frac{2}{f_T}) \\ 0, & t \in [1 + \frac{2}{f_T}, 1.5 + \frac{2}{f_T}] \end{cases} \tag{3.61}$$

with amplitude, $A^* = A(2\pi f_T^2)$, where the position amplitude, A, and frequency, f_T were selected as in Table 3.1. To allow visual display of the desired output and the actual output to the human user, the target frequency, f_T of the desired trajectory, y_d was chosen to be below the smooth-pursuit visual tracking limit of $f_T^* = 0.5$ Hz for typical healthy human

users [8], i.e.,

$$f_T \leq f_T^* = 0.5 \text{ Hz}. \tag{3.62}$$

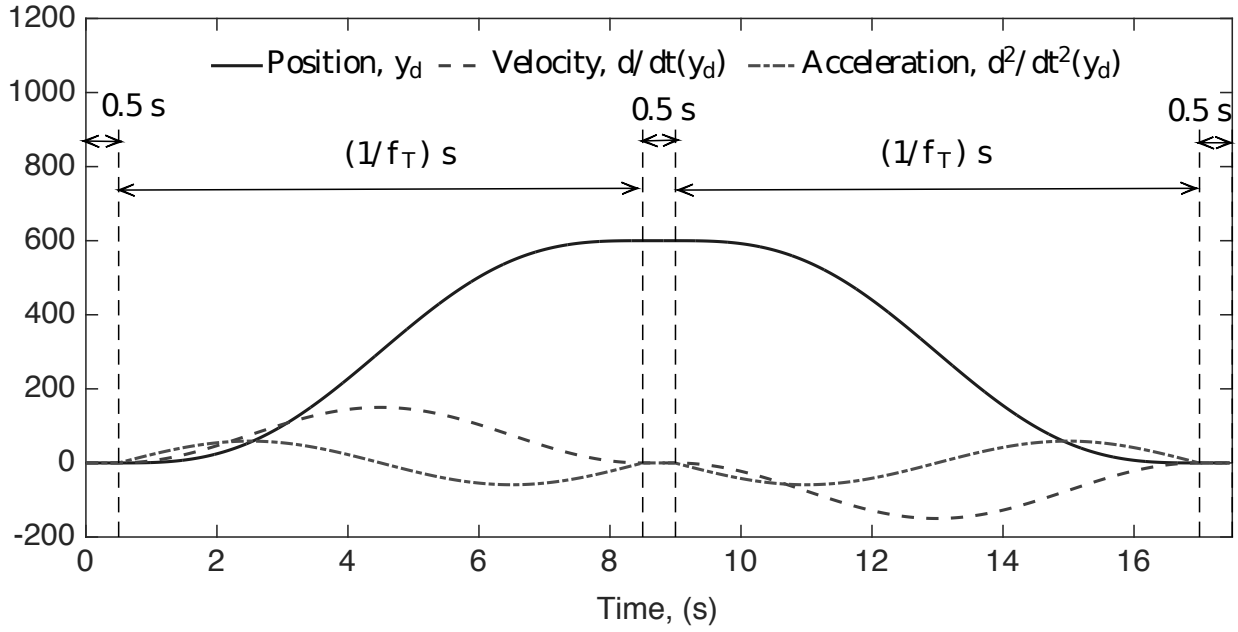


Figure 3.3: Output trajectory of the target, y_d and its first and second time derivatives.

3.3.5 Model Estimation trials - Data Analysis

The frequency response of the feedback loop, $\hat{G}_{fb}(\omega)$ was obtained from model-estimation trials. During the model estimation, the desired output, $y_d = \overline{y}_d$ was specified to be pure sinusoidal trajectories of the form,

$$\overline{y}_d(t) = A\sin(2\pi f t), \text{ for } t \geq 0, \tag{3.63}$$

with the iterative controller deactivated (manual control task). By performing a sweep through the frequencies of interest (by varying the desired-output frequency, f), the human

response $\overline{G}_H$ was computed at each frequency as, (see Fig. 3.1)

$$\overline{G}_H(\omega) = \frac{u_h(\omega)}{\overline{y}_d(\omega) - y(\omega)}, \qquad \omega = 2\pi f. \tag{3.64}$$

Feedback model, $\hat{G}_{fb}$

Ten trials were performed at each frequency, f which were in the range $[0.1, 1.0]$ Hz. The resulting frequency response was used to fit the parameters of the cross-over model, $\hat{G}_H$ in Eq. (3.59), as shown in Fig. 3.4. First, the exponential phase component from the human response $\overline{G}_H$ was removed,

$$\overline{G}^*_H(\omega) = \overline{G}_H(\omega) e^{j\tau_d \omega}, \quad j = \sqrt{-1}, \tag{3.65}$$

for different values of the time-delay parameter, $\tau_d \in [0, 1]$ s, which represents typical time-delay values found in literature [39]. Next, the best-fit transfer function, $\hat{G}^*_H$ to the delay-removed response, $\overline{G}^*_H$, in Eq. (3.65), was computed using the Matlab command `invfreqs`,

$$\hat{G}^*_H(s) = K_{p,\tau_d} \frac{1}{(\tau_{l,\tau_d})s + 1}, \tag{3.66}$$

to obtain the time-delay dependent values of the parameters, K_{p,τ_d}, τ_{l,τ_d}. The optimal values for the parameters, K_p, τ_L and the time-delay, τ_d were then obtained by minimizing over the time delay, τ_d as,

$$J = \min_{\tau_d} \left\| \overline{G}_H(\omega) - \hat{G}^*_H(s)\Big|_{s=j\omega} e^{-j\tau_d \omega} \right\|_2, \tag{3.67}$$

where $\| \cdot \|_2$ is the standard 2-norm of a vector.

The linear crossover model, $\hat{G}_H$ in Eq. (3.59) is a good representation of the human response for low frequencies, but larger deviation is seen at high frequencies in Fig. 3.4, which compares the fitted model, $\hat{G}_H$ and the frequency response data, $\overline{G}_H$ for Subject 2. (Data for Subject 2 is presented in plots in this Section because the final tracking-precision results were closest to the average value over all subjects.) Such deviation at frequencies higher than the smooth pursuit limit of 0.5 Hz is expected because visual observation of the

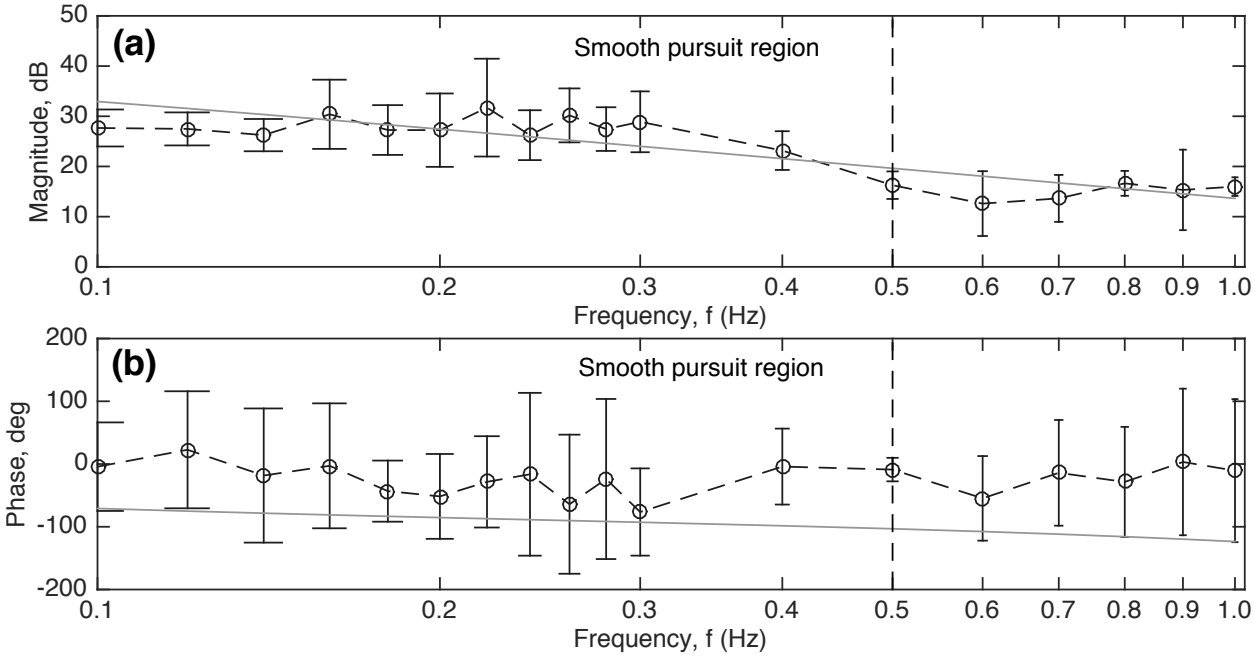


Figure 3.4: Human response model, $\hat{G}_H$, shown as a solid line, fitted to the frequency response, $\overline{G}_H$ obtained experimentally (evaluation trials), shown as a dashed line with circular markers, for Subject 2. Vertical error bars indicate one standard deviation for 10 trials at each frequency, f. The linear model, $\hat{G}_{fb}$ is expected to fit well till the smooth-pursuit tracking limit frequency, $f_T^* = 0.5$ Hz [8] indicated by the vertical dashed line.

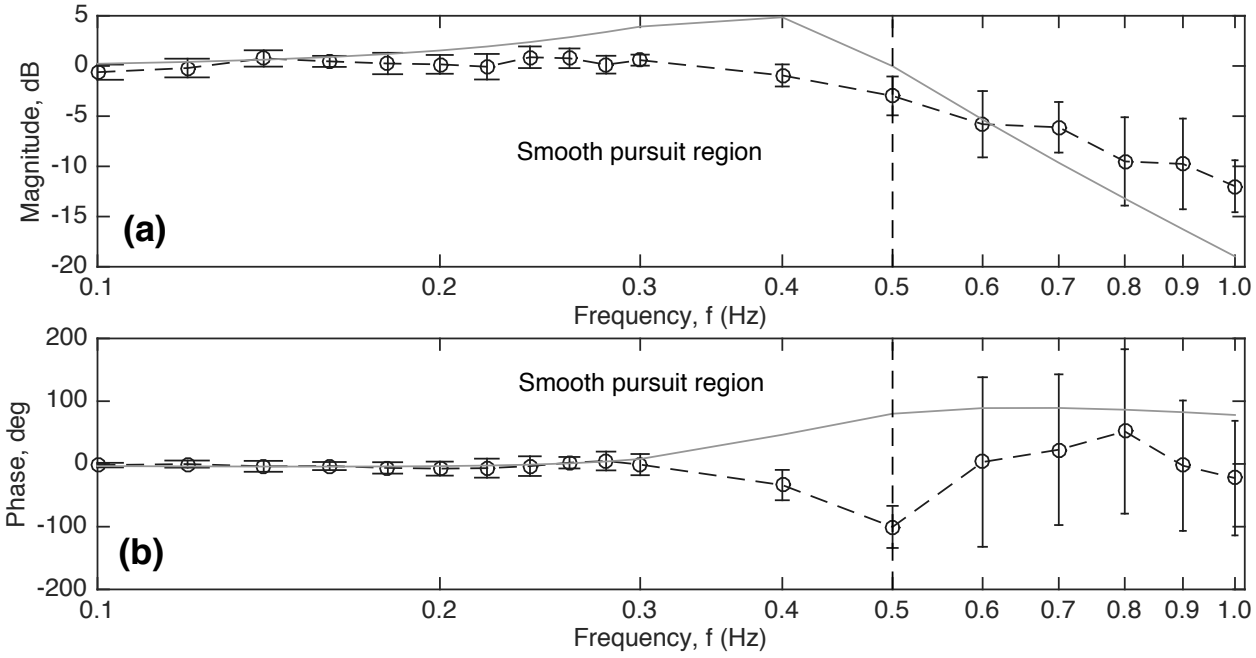


Figure 3.5: Human-in-the-loop feedback-loop model, $\hat{G}_{fb}$, shown as a solid line, for Subject 2, fitted to the frequency response, $\overline{G}_{fb}$ obtained experimentally, shown as a dashed line with circular markers, for Subject 2. Vertical error bars indicate one standard deviation for 10 trials at each frequency, f. The smooth-pursuit tracking limit frequency, $f_T^* = 0.5$ Hz [8] is indicated by the vertical dashed line.

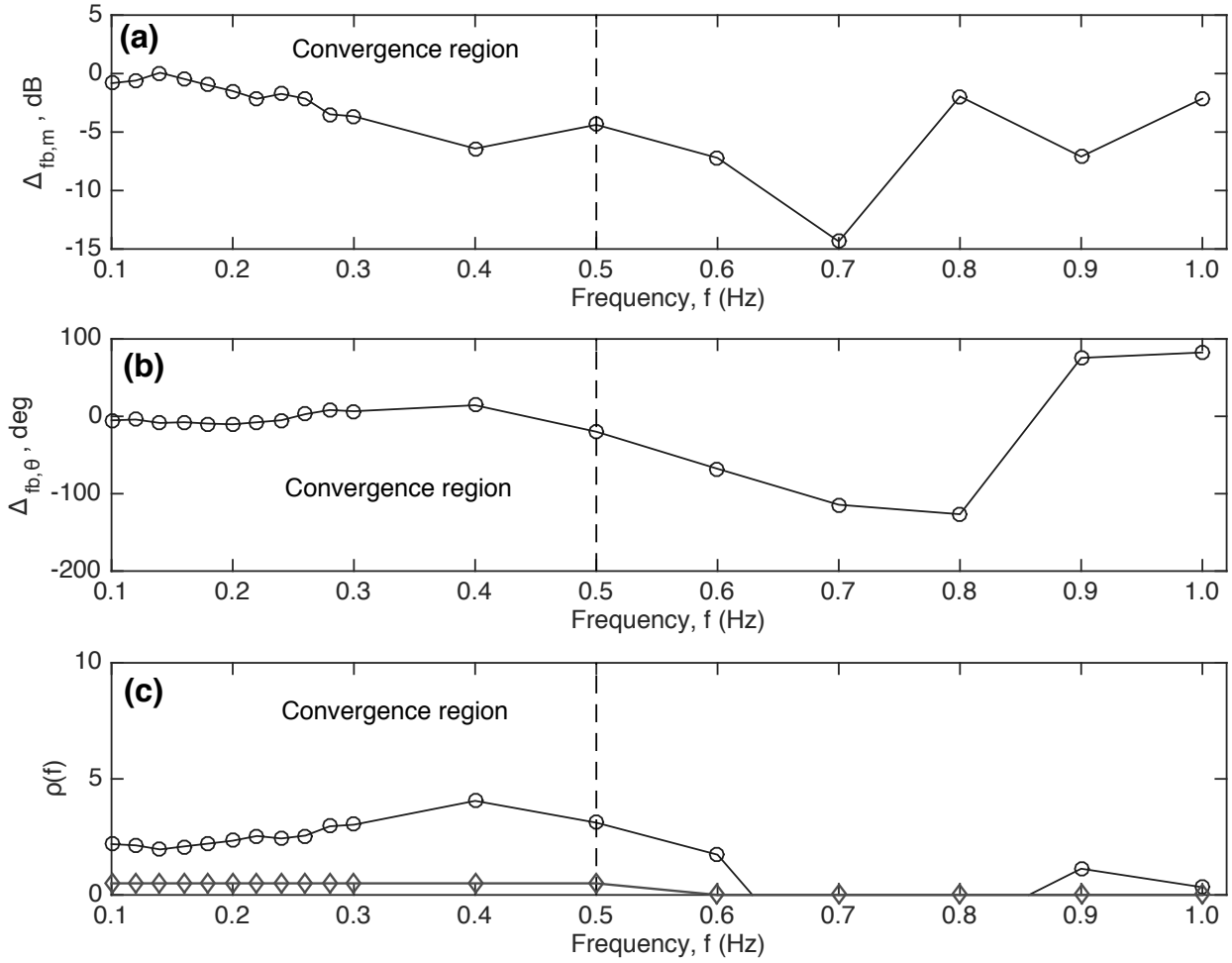


Figure 3.6: Feedback modeling error, Δ_{fb}, computed using Eq. (3.46). (a) magnitude error, $\Delta_{fb,m}$; (b) phase error, $\Delta_{fb,\theta}$; (c) maximum iteration gain, ρ^* (circular markers), computed using Eqs. (3.24) and (3.25), Lemma 1, and the chosen iteration gain, ρ (diamond markers), according to Eq. (3.71) for Subject 2. A large iteration gain, ρ is chosen in the main convergence region till frequency, $f_c = 0.5$ Hz.

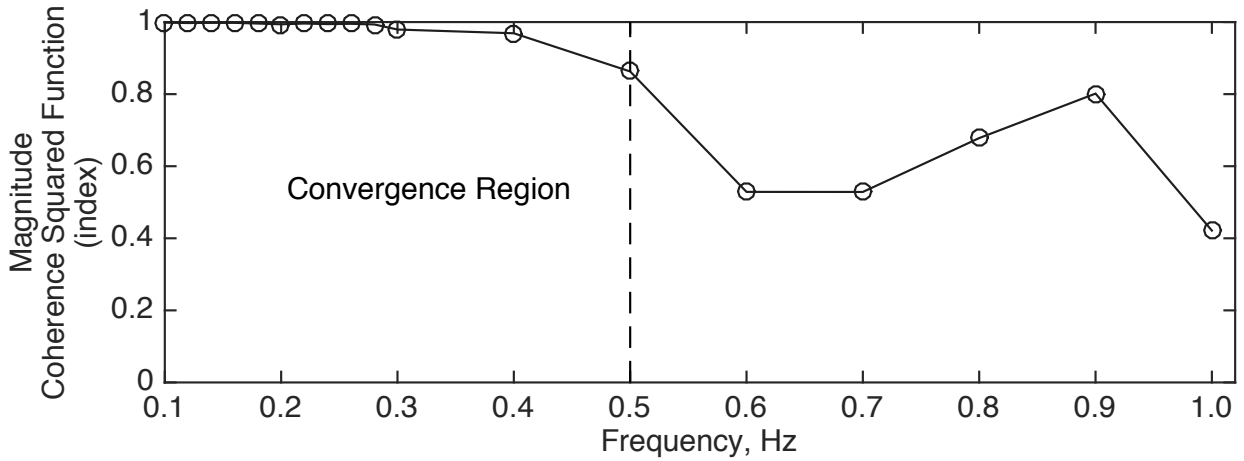


Figure 3.7: Average magnitude of the coherence squared function, C_{xy} for the feedback frequency response, G_{fb}, computed using Eq. (3.70) for Subject 2. The values are presented as the mean index at each frequency for 10 trials of the model-estimation trials. Smaller coherence values, $0 < C_{xy} << 1$, e.g., after frequency, $f_c = 0.5$ Hz (vertical dashed line) indicates potential increase in non-linearity and/or measurement noise.

error is expected to involve large nonlinear remnants at high frequencies [8], which are not captured by the linear model, see [39].

The fitted human model, $\hat{G}_H$ was then used in conjunction with the known model of the controlled plant, $\hat{G}$, i.e.,

$$\hat{G}(\omega) = \hat{G}(s)\Big|_{s=j\omega}, \quad \hat{G}(s) = \frac{\hat{K}_{gain}}{s^2 + 2\hat{\zeta}\hat{\omega}_n s + \hat{\omega}_n^2}, \quad j = \sqrt{-1}, \tag{3.68}$$

to obtain the feedback-loop model, $\hat{G}_{fb}$ as in Eq. (3.56). Note that 10% errors were present between the parameters of the model, $\hat{G}$ and actual system, G. The estimated parameter values in the above model are as given in Table 3.1. Fig. 3.5 depicts the frequency response of the human-in-the-loop feedback model, $\hat{G}_{fb}$ compared to the experimentally obtained frequency response for Subject 2,

$$G_{fb}(\omega) = y(\omega)/y_d(\omega). \tag{3.69}$$

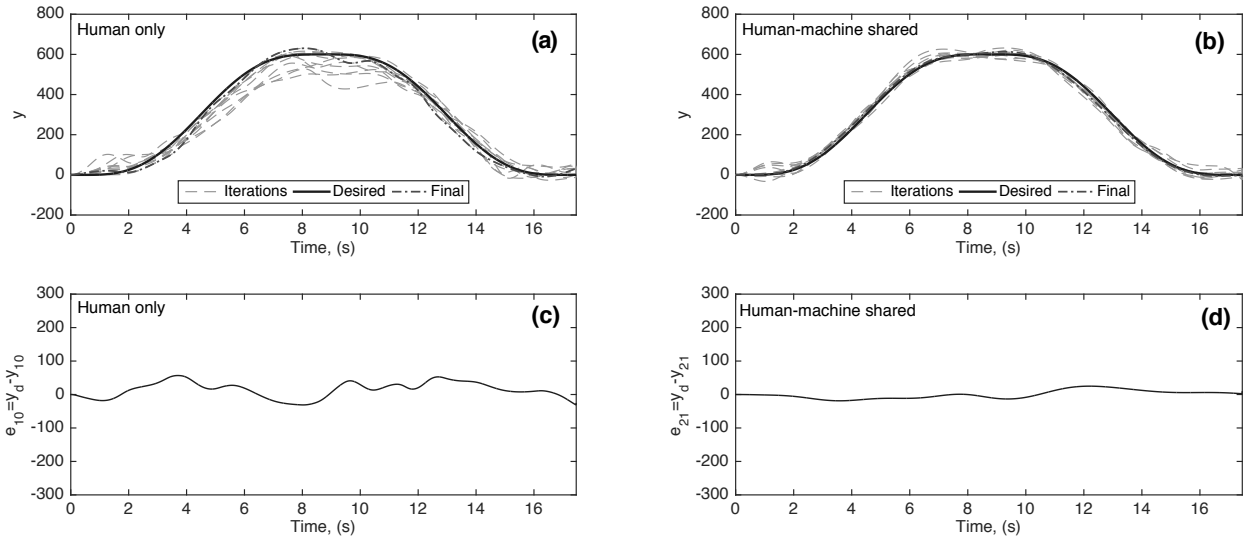


Figure 3.8: Convergence of inversion-based iterative learning control for the target frequency, $f_T = 0.125$ Hz for Subject 2: (a) system output, y_k during human-only control ($k \in [1, 10]$); (b) system output, y_k during shared human-machine iterative learning control ($k \in [11, 20]$); (c) tracking error, $e_{10} = y_d - y_{10}$ at the end of human-only control ($k = 10$); (d) tracking error, $e_{21} = y_d - y_{21}$ at the end of shared control ($k = 21$).

Iteration gain, ρ

The iteration gain, ρ should be chosen to be as large as possible while satisfying the conditions in Lemma 1 for faster convergence, i.e., $0 < \rho < \rho^*$ from Eq. (3.25). However, from Lemma 2, Eq. (3.50), larger value of iteration gain, ρ can lead to greater noise effects. Thus, a frequency dependent iteration gain, ρ needs to be used such that it has larger values at smaller frequencies (where noise is insignificant) while suppressing higher frequencies (where the modeling errors and noise effects are high and can lead to violation of the convergence conditions in Lemma 1) by using a smaller iteration gain, ρ at higher frequencies. For example, the magnitude of the coherence squared function, C_{xy} (referred to as the coherence index) for the input, y_d and output, y of the feedback response, G_{fb}, obtained using data from the model-parameter estimation trials, is shown in Fig. 3.7. This coherence index, C_{xy}

is defined, for each frequency, f as,

$$C_{xy}(f) = \frac{|P_{xy}(f)|^2}{P_{xx}(f)P_{yy}(f)}, \tag{3.70}$$

where $P_{xx}(f)$ is the power spectral density of the input, y_d at frequency, f, $P_{yy}(f)$ is the power spectral density of the output, y, and $P_{xy}(f)$ is the cross-spectral density. Note that the coherence index, $C_{xy}(f)$ is close to one ($C_{xy} \approx 1$) for frequencies, $f \leq f_c = 0.5$ Hz and becomes substantially smaller for higher frequencies. Small values of the coherence index, $C_{xy}(f)$ can indicate potential non-linearity or substantial noise. Therefore, the iteration gain, ρ was chosen as,

$$\rho(f) = \begin{cases} 0.5, & \text{for } f \leq f_c = 0.5 \text{ Hz} \\ \\ 0, & \text{for } f > 0.6 \text{ Hz} \end{cases}, \tag{3.71}$$

and linearly interpolated between 0.5 Hz and 0.6 Hz such that $\rho(f) < \rho^*(f)$ to satisfy conditions of Lemma 1 for all frequencies, f. Here, $f_c = 0.5$ Hz was the frequency above which the modeling errors are high as seen from the results of the evaluation trials shown in Fig. 3.6 and Fig. 3.7.

3.3.6 Iterations

A total of $k = 21$ iterations were performed, divided into three distinct control regimes: (i) human feedback only, without machine learning, $u_{c,k} = 0$, for iterations, $k \in [1, 10]$, (ii) human-machine shared control, using human-guided inversion-based iterative control, for iterations, $k \in [11, 20]$ with zero initial controller input, $u_{c,10} = 0$, (iii) machine open-loop control ($k = 21$) with learned control input, $u_{21} = u_{c,20}$ without human feedback, $u_{h,21} = 0$. A rest period was provided at the end of every iteration to enable resetting of the initial conditions.

3.3.7 Convergence of system output to desired output

The output tracking performance is improved by the iterative learning controller when compared to the human-alone performance, as shown in Fig. 3.8 for the target frequency, $f_T = 0.125$ Hz for Subject 2, whose performance is close to the average performance level of the nine test subjects, see Table 3.12. Note that the tracking error, e_{21} at the end of shared human-machine iterative learning is lower than the tracking error, e_{10} in Fig. 3.8 with the human-only control. To quantify the reduction, the tracking error, e is quantified in each iteration step in terms of the normalized maximum error, $e_{max,k}$ and the normalized Root Mean Square (RMS) error, $e_{rms,k}$, given by,

$$e_{max,k} = \frac{\max_{t_i} |y_d(t_i) - y_k(t_i)|}{A}, \tag{3.72}$$

for $t_i \in [kT, (k+1)T]$, where T is the time-period of each iteration, and,

$$e_{rms,k} = \frac{1}{A}\left(\frac{1}{N}\sum_{i=1}^{N}[y_d(t_i) - y(t_i)]^2\right)^{1/2}, \tag{3.73}$$

for $t_i \in [kT, (k+1)T]$, where N is the number of data samples in one iteration. The change of the normalized maximum error, $e_{max,k}$ with iteration step k is shown in Fig. 3.9 for Subject 2. In these experiments the normalized maximum error, $e_{max,k}$ decreased as the human learned to control the system and tended to reach steady state operation after about 5 iterations. With the use of shared learning, the normalized maximum error, $e_{max,k}$ reduced as the human operator adjusted to the presence of the controller input, $u_{c,k}$ for iterations, $k \in [11, 20]$, as seen in Fig. 3.9.

The tracking performance of the human alone with that of the shared human-machine learning control is compared in terms of the average normalized maximum error per iteration

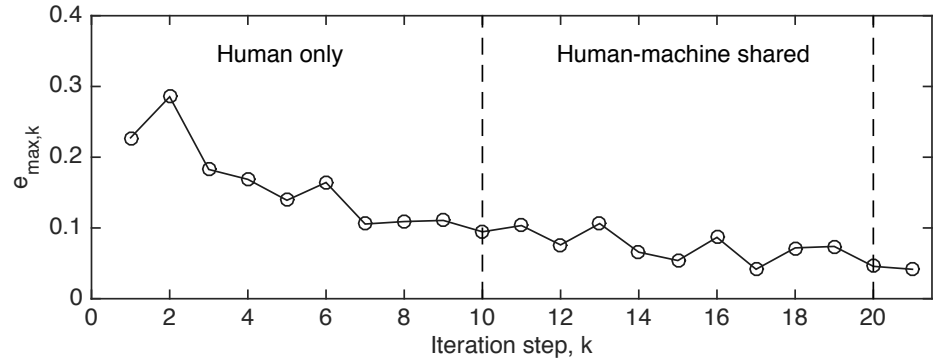


Figure 3.9: Maximum error per iteration, $e_{max,k}$, defined in Eq. (3.72) for the target frequency, $f_T = 0.125$ Hz for Subject 2.

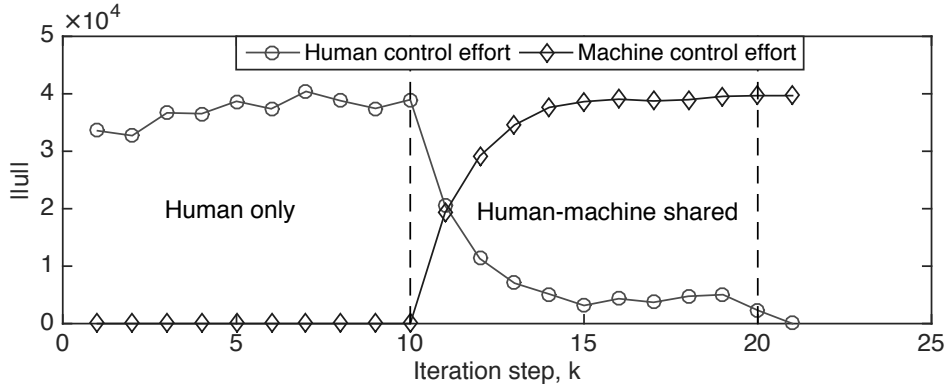


Figure 3.10: Control effort, $\|u\|$, human (circular markers), machine controller (diamond markers), as defined in Eq. (3.81) for human, and Eq. (3.82) for machine controller, for target frequency, $f_T = 0.125$ Hz for Subject 2.

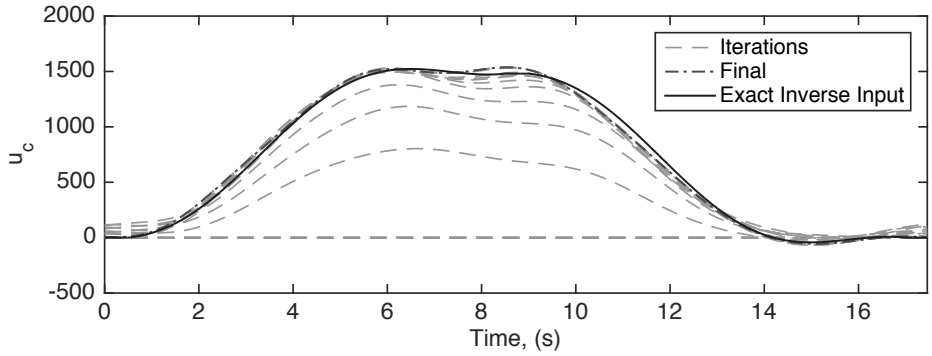


Figure 3.11: Controller input, $u_{c,k}(t)$ for iterations $k \in [10, 20]$ (dashed lines), compared to the exact inverse input, $u_c^*(t)$ (solid line), with final learned controller input in iteration $k = 21$ (dash-dotted line), for target frequency, $f_T = 0.125$ Hz for Subject 2.

(during the last five iterations of each control regime)

$$\overline{e}_{max,human} = \frac{1}{5}\sum_{k=6}^{10} e_{max,k}, \quad \text{Human-only} \tag{3.74}$$

$$\overline{e}_{max,shared} = \frac{1}{5}\sum_{k=16}^{20} e_{max,k}, \quad \text{Human-machine shared,} \tag{3.75}$$

and the average normalized root-mean-square (RMS) error (during the last five iterations of each control regime)

$$\overline{e}_{rms,human} = \frac{1}{5}\sum_{k=6}^{10} e_{rms,k}, \quad \text{Human-only} \tag{3.76}$$

$$\overline{e}_{rms,shared} = \frac{1}{5}\sum_{k=16}^{20} e_{rms,k}, \quad \text{Human-machine shared.} \tag{3.77}$$

For the nine test subjects, the average normalized maximum error per iteration with the human alone was $\overline{e}_{max,human} = 0.10 \pm 0.03$, while the average normalized maximum error per iteration at the end of the shared learning was $e_{max,21} = 0.03 \pm 0.03$. This represents a percentage reduction

$$\%E_{max,red} = \frac{\overline{e}_{max,human} - e_{max,21}}{\overline{e}_{max,human}} \times 100, \tag{3.78}$$

of $\%E_{max,red} = 67 \pm 24\%$. Similarly, the average normalized RMS error per iteration with human alone was $\overline{e}_{rms,human} = 0.04 \pm 0.01$, while the average normalized RMS error per iteration at the end of the shared learning was $e_{rms,21} = 0.02 \pm 0.01$, a percentage reduction

$$\%E_{rms,red} = \frac{\overline{e}_{rms,human} - e_{rms,21}}{\overline{e}_{rms,human}} \times 100, \tag{3.79}$$

of $\%E_{rms,red} = 64 \pm 26\%$. Thus, the human and the machine were able to learn simultaneously and reduce the tracking error, e.

3.3.8 Learning by machine controller

The machine controller, G_c learns from the human input, u_h to control the system, G. For the case of Subject 2 shown in Figs. 3.8 and 3.9, the tracking error, e_{21} for the machine open-loop control using the learned input, $u = u_{c,21}$ (without human feedback) is similar to the error, e_{20} at the end of shared human-machine iterative learning — the normalized maximum error, $e_{max,21} = 0.04$ and the normalized root-mean-square (RMS) error, $e_{rms,21} = 0.02$, which are both small. This indicates that the machine input, u_{21} is close to the exact inverse input, u_c^*, given by,

$$u_c^*(\omega) = G^{-1}(\omega) y_d(\omega), \tag{3.80}$$

needed for exactly tracking the desired output, y_d as illustrated in Fig. 3.11.

Due to the machine learning the input needed for tracking, the human control effort, $\|u_{h,k}\|$, given by,

$$\|u_{h,k}\| = \left(\sum_i^N u_{h,k}^2(t_i) \right)^{1/2}, \quad \text{for } t_i \in [kT, (k+1)T], \tag{3.81}$$

decreases with iteration number k as seen in Fig. 3.10, while the machine controller effort, $\|u_{c,k}\|$, given by,

$$\|u_{c,k}\| = \left(\sum_i^N u_{c,k}^2(t_i) \right)^{1/2}, \quad \text{for } t_i \in [kT, (k+1)T], \tag{3.82}$$

Figure 3.12: Experimental results: comparison of average normalized tracking performance for all nine human subjects[1].

Target Frequency f_T	$\frac{f_T}{f_n}$	**Maximum tracking error**				**RMS tracking error**				**Human Effort Reduction**
		Human only $\overline{e}_{max,human}$ Eq. (3.74)	**Shared** $\overline{e}_{max,shared}$ Eq. (3.75)	**Machine open-loop** $e_{max,21}$ Eq. (3.72)	**% reduction** $\%E_{max,red}$ Eq. (3.78)	**Human only** $\overline{e}_{rms,human}$ Eq. (3.76)	**Shared** $\overline{e}_{rms,shared}$ Eq. (3.77)	**Machine open-loop** $e_{rms,21}$ Eq. (3.73)	**% reduction** $\%E_{rms,red}$ Eq. (3.79)	$\%UH_{red}$ Eq. (3.83)
Using fitted parameters in human feedback model, $\hat{G}_H$										
0.125 Hz	0.5	0.10 ± 0.03	0.05 ± 0.03	0.03 ± 0.03	67 ± 24	0.04 ± 0.01	0.02 ± 0.01	0.02 ± 0.01	64 ± 26	94 ± 2
0.25 Hz	1.0	0.18 ± 0.05	0.10 ± 0.03	0.07 ± 0.03	59 ± 22	0.09 ± 0.02	0.05 ± 0.02	0.04 ± 0.02	58 ± 23	86 ± 6
Using nominal parameters in human feedback model, $\hat{G}_H$										
0.125 Hz	0.5	0.10 ± 0.03	0.05 ± 0.02	0.03 ± 0.02	72 ± 12	0.04 ± 0.01	0.02 ± 0.01	0.01 ± 0.01	67 ± 13	92 ± 3
0.25 Hz	1.0	0.18 ± 0.04	0.08 ± 0.03	0.05 ± 0.03	71 ± 12	0.09 ± 0.02	0.03 ± 0.02	0.03 ± 0.01	70 ± 14	89 ± 8

[1] The values, for all human subjects, are reported as $\mu_x \pm \sigma_x$ where $\mu_x = \sum_{i=1}^{n} (x_i/n)$ is the sample mean, $\sigma_x = \sqrt{\sum_{i=1}^{n} \frac{(x_i-\mu_x)^2}{n-1}}$ is the standard deviation, x_i is the value for each human subject i, where $i = 1, 2, \ldots, n$, and $n = 9$.

increases with iteration number k as seen in Fig. 3.10. The human control effort (without the machine control) is $\|u_{h,10}\| = 3.9 \times 10^4$ at iteration step $k = 10$, which is substantially reduced to $\|u_{h,20}\| = 0.2 \times 10^4$ at the end of the shared learning at iteration step $k = 20$. This represents a percentage reduction

$$\%UH_{red} = \frac{\|u_{h,10}\| - \|u_{h,20}\|}{\|u_{h,10}\|} \times 100, \tag{3.83}$$

of $\%UH_{red} = 94\%$ in the human effort for Subject 2 with the use of the proposed iterative learning. Such reduction in the human effort was seen for each subject, as seen in Table 3.2. Overall, an average percentage reduction of $\%UH_{red} = (94 \pm 2)\%$ in the human effort was observed over the nine test subjects. Thus, the experimental results show that the iterative learning control leads to improved tracking performance with simultaneous reduction in human effort.

3.3.9 Effect of target frequency, f_T

Although the human tracking performance tends to degrade with increased target frequency, f_T, the shared learning yields significantly better tracking than the human-alone case. The tracking performance of the iterative controller was studied for two categories of target frequencies, f_T: (a) $f_T/f_n < 1$ and (b) $f_T/f_n \approx 1$, where f_n is the natural frequency of the plant, G, as given in Table 3.1. The tracking performance by the human-alone is expected to deteriorate from category (a) to (b), due to increased phase differences between the exact-tracking inverse input, u_c^* in Eq. (3.80) and the desired output, y_d at higher target frequencies, f_T.

The maximum and RMS error with human alone and at the end of the shared learning are compared for the different categories of target frequencies and the average values over all human subjects are reported in Table 3.12. Note that the average normalized maximum error per iteration with the human alone, $e_{max,human}$ in Eq. (3.74), increases with increasing target frequency, f_T, with $e_{max,human} = 0.10 \pm 0.03$ for target frequency $f_T = 0.125$ Hz which

increases to $e_{max,human} = 0.18 \pm 0.05$ for target frequency $f_T = 0.25$ Hz. Nevertheless, the normalized maximum error at the end of the shared human-machine learning, $e_{max,21}$ is reduced in each case, resulting in the relative percentage reduction, $\%E_{max,red}$ in Eq. (3.78) to decrease from $\%E_{max,red} = (67 \pm 24)\%$ for target frequency $f_T = 0.125$ Hz to $\%E_{max,red} = (59 \pm 22)\%$ for target frequency $f_T = 0.25$ Hz. A similar trend is seen for the normalized RMS error, as seen in Table 3.12.

Thus, the results show that the inversion-based iterative controller is able to learn from the human demonstrations and capture most of the human effort (about 90%). Moreover, it achieves greater tracking precision (about 60% reduction in the average error) than what was achieved by the human alone.

3.3.10 Using nominal human feedback model

Nominal parameter values from existing literature for the human feedback model, $\hat{G}_H(s)$ in Eq. (3.59) can be used instead of performing model estimation trials to obtain these parameters from the experimental frequency response data, $\overline{G}_H$ in Eq. (3.64). For example, the time-delay parameter, τ_d has typical values in the range 0.1-0.2 s for healthy individuals [41], while the equalization time-constant, τ_L depends on the crossover frequency, ω_c of the open-loop response, $|G_H(\omega)G(\omega)|$ as,

$$\frac{1}{\tau_L} \ll \omega_c, \tag{3.84}$$

where $\omega_c \approx 5$ rad/s for second order systems with high natural frequency, $\omega_n > 1/\tau_d$ [39]. Table 3.1 lists the nominal values of these parameters, τ_d and τ_L. The overall gain, K_p in Eq. (3.59) is characteristic of the user preferences such as range of hand motion, and can be determined from one initial set of the model estimation trials without the need for additional trials (at various frequencies) to obtain the frequency response data for determining the human feedback response model, $\hat{G}_H$.

Table 3.2: Tracking performance using the individually fitted model at target frequency, $f_T = 0.125$ Hz.

Subject	$\%E_{max,red}$ Eq. (3.78)	$\%E_{rms,red}$ Eq. (3.79)	$\%UH_{red}$ Eq. (3.83)
1	95	94	98
2	64	62	94
3	82	76	93
4	88	86	97
5	56	53	96
6	83	79	92
7	71	81	93
8	23	16	93
9	41	31	91
Overall	$\mathbf{67 \pm 24}$	$\mathbf{64 \pm 26}$	$\mathbf{94 \pm 2}$

Table 3.3: Tracking performance using the nominal human feedback model at target frequency, $f_T = 0.125$ Hz.

Subject	$\%E_{max,red}$ Eq. (3.78)	$\%E_{rms,red}$ Eq. (3.79)	$\%UH_{red}$ Eq. (3.83)
1	82	74	98
2	69	66	93
3	84	84	93
4	47	41	90
5	63	52	93
6	82	76	88
7	71	64	92
8	78	76	92
9	70	64	95
Overall	$\mathbf{72 \pm 12}$	$\mathbf{67 \pm 13}$	$\mathbf{92 \pm 3}$

The experimental results indicate that the use of nominal parameters of the human feedback model, $\hat{G}_H(s)$ leads to output-tracking performance improvements for each subject as shown in Table 3.3. For example, the inversion-based iterative controller is able to learn from the human demonstrations and capture most of the human effort (overall 92%). Moreover, it achieves greater tracking precision (overall 72% reduction in the average error) than what was achieved by the human alone. These improvements with the nominal parameters for the human feedback model are comparable to the case when using an individually-fitted human feedback model in Table 3.2, where the observed differences are found to be statistically not significant. For example, a repeated-measures t-test [64] was conducted to compare the relative percentage reduction in maximum error, $\%E_{max,red}$, and RMS error, $\%E_{rms,red}$ for individually-fitted versus the nominal-parameters case for the target frequency, $f_T = 0.125$ Hz. There was no significant difference in the relative percentage reduction, $\%E_{max,red}$ for individually-fitted model and nominal-parameter model conditions; $p = 0.6 > \alpha = 0.05$. Similarly, there was no significant difference in the relative percentage reduction, $\%E_{rms,red}$ for individually-fitted model and nominal-parameter model conditions; $p = 0.84 > \alpha = 0.05$. These results suggest that it is sufficient to use the nominal parameters for the human feedback model with the proposed approach, since it requires less trials for parameter estimation compared to individual-fitting.

3.4 Pushing the limits of nominal models

Human-response models, $\hat{H}_{fb}$ (in Eq. (3.59)) are only available for limited types of controlled-systems, G (e.g., for first and second order linear systems [39, 31]), which restricts the applicability of the above intent estimation approach. To extend the technique to general linear systems, for which such human-feedback response models are not known, the apparent

---

This section is derived from the publication: Rahul B Warrier and Santosh Devasia. "Inferring Intent for Novice Human-in-the-Loop Iterative Learning Control". In: *IEEE Transactions on Control Systems Technology* (2016)

system perceived by the user, G_A is modified by using a controller G_r, as shown in Fig. 3.13 to be,

$$G_A(s) = G_r(s)G(s) = \hat{G}^{-1}(s)G(s) \approx Ke^{-\tau_d s}, \tag{3.85}$$

where, τ_d is the time-delay in seconds introduced by preview-based inversion of nonminimum-phase systems, covered in Section 3.4.1. Thus, the *inversion* of the controlled system, G using the controller, $G_r = \hat{G}^{-1}$ (where $\hat{G}$ is a stable model of the stable controlled-system, G), results in a simplified apparent system, G_A, for which the human-feedback response model, $\hat{H}_{fb}$ can be obtained as [39],

$$\hat{H}_{fb}(s) = \frac{K_p e^{-\tau_e s}}{(\tau_I s + 1)}, \tag{3.86}$$

provided the time-delay, τ_d of the apparent system perceived by the user, G_A in Eq. (3.85) is not too large [26].

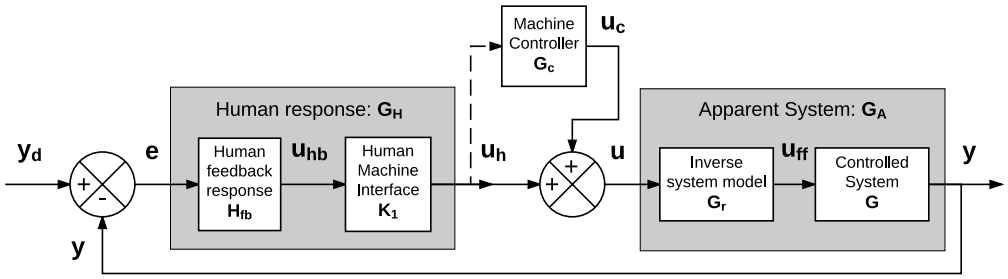


Figure 3.13: *With controlled-system inverse.* Block diagram of the proposed inversion-based learning approach. Solid line depicts continuous flow of information; dashed line depicts intermittent flow of information; y_d is the desired output signal, y is the output of the controlled-system, G, $e = y_d - y$ is the perceived error, u_h is the human input, u_c is the machine controller input.

3.4.1 Preview-based inversion of controlled-system

The procedure for preview-based inversion from [76] of the controlled-system model, $\hat{G}$ is described below.

Exact feed-forward inversion

Let the state-space form of the controlled-system model, $\hat{G}$ be

$$\begin{aligned} \dot{x}(t) &= \hat{A}x(t) + \hat{B}u(t) \\ y(t) &= \hat{C}x(t). \end{aligned} \tag{3.87}$$

Given a desired output trajectory $\tilde{y}_d(\cdot)$, the inversion-based approach finds bounded input-state trajectories $[u_{inv}(\cdot), x_{ref}(\cdot)]$ that satisfy the model dynamics in Eq. (3.87) and achieve the desired output, $y = \tilde{y}_d$,

$$\begin{aligned} \dot{x}_{ref}(t) &= \hat{A}x_{ref}(t) + \hat{B}u_{inv}(t) \\ \tilde{y}_d(t) &= \hat{C}x_{ref}(t). \end{aligned} \tag{3.88}$$

Since the model system, $\hat{G}$ is stable, applying the inverse input, u_{inv} to the system, G results in output tracking, i.e., $x(t) \to x_{ref}(t)$ and $y(t) \to \tilde{y}_d(t)$ as $t \to \infty$.

Let the relative degree (difference between number of poles and zeros) of the LTI system in Eq. (3.87) be r, which implies that the r^{th} time derivative of the output will be given by,

$$\frac{d^r}{dt^r}y(t) \;=\; \hat{C}\hat{A}^r x(t) + \hat{C}\hat{A}^{r-1}\hat{B}u(t) \;=\; A_y x(t) + B_y u(t), \tag{3.89}$$

where B_y is invertible. Then, the inverse system is given by [17],

$$\begin{aligned} \dot{\eta}(t) &= A_{inv}\eta(t) + B_{inv}\mathbb{Y}_d(t) \\ u_{inv}(t) &= C_{inv}\eta(t) + D_{inv}\mathbb{Y}_d(t), \end{aligned} \tag{3.90}$$

where,

$$\begin{aligned} A_{inv} &:= T_\eta \left[\hat{A} - \left(\hat{B}B_y^{-1}A_y\right)\right] T_r^{-1} \\ B_{inv} &:= \left[T_\eta \left[\hat{A} - \left(\hat{B}B_y^{-1}A_y\right)\right] T_l^{-1} \;\;\vdots\;\; \left(T_\eta \hat{B}B_y^{-1}\right)\right] \\ C_{inv} &:= -B_y^{-1}A_y T_r^{-1} \\ D_{inv} &:= \left[\left(-B_y^{-1}A_y T_l^{-1}\right) \;\;\vdots\;\; \left(B_y^{-1}\right)\right] \\ \mathbb{Y}_d(t) &:= \begin{bmatrix} \zeta_d(t) \\ \\ \dfrac{d^r}{dt^r}\tilde{y}_d(t) \end{bmatrix}, \end{aligned} \tag{3.91}$$

with T representing a coordinate transformation,

$$\begin{bmatrix} \zeta_d(t) \\ \hdots\hdots \\ \eta(t) \end{bmatrix} = \begin{bmatrix} \tilde{y}_d(t) \\ \frac{d}{dt}\tilde{y}_d(t) \\ \vdots \\ \frac{d^{(r-1)}}{dt^{(r-1)}}\tilde{y}_d(t) \\ \hdots\hdots\hdots\hdots \\ \eta(t) \end{bmatrix} = \begin{bmatrix} \hat{C} \\ \hat{C}\hat{A} \\ \vdots \\ \hat{C}\hat{A}^{r-1} \\ \hdots\hdots\hdots \\ T_\eta \end{bmatrix} x(t) = \begin{bmatrix} T_\zeta \\ \hdots. \\ T_\eta \end{bmatrix} x(t) = Tx(t) \tag{3.92}$$

$$x(t) = T^{-1} \begin{bmatrix} \zeta_d(t) \\ \eta(t) \end{bmatrix} = \begin{bmatrix} T_l^{-1} & \vdots & T_r^{-1} \end{bmatrix} \begin{bmatrix} \zeta_d(t) \\ \eta(t) \end{bmatrix}, \tag{3.93}$$

and, T_η is chosen such that T is invertible.

Note that the eigenvalues of A_{inv} are the zeros of the original system in Eq. (3.87). Thus, in general, for a system with hyperbolic zero dynamics, i.e., none of the zeros of the system in Eq. (3.87) lie on the imaginary axis of the complex plane (the case for non-hyperbolic zero dynamics is discussed in [20]), a transformation U can be found such that the internal dynamics in Eq. (3.90) can be decoupled into a stable subsystem (σ_s) and an unstable subsystem (σ_u):

$$\dot{\sigma}_s(t) = \tilde{A}_s\sigma_s(t) + \tilde{B}_s\mathbb{Y}_d(t) \tag{3.94}$$

$$\dot{\sigma}_u(t) = \tilde{A}_u\sigma_u(t) + \tilde{B}_u\mathbb{Y}_d(t), \tag{3.95}$$

where,

$$\sigma(t) := \begin{bmatrix} \sigma_s(t) \\ \sigma_u(t) \end{bmatrix} = U\eta(t) \tag{3.96}$$

$$\begin{bmatrix} \tilde{A}_s & 0 \\ 0 & \tilde{A}_u \end{bmatrix} := UA_{inv}U^{-1}, \quad \text{and} \quad \begin{bmatrix} \tilde{B}_s \\ \tilde{B}_u \end{bmatrix} := UB_{inv}. \tag{3.97}$$

The individual subsystems in Eqs. (3.94) and (3.95) have bounded solutions given by,

$$\sigma_{s,ref}(t) = \int_{-\infty}^{t} e^{\tilde{A}_s(t-\tau)} \tilde{B}_s \mathbb{Y}_d(\tau)\, d\tau \tag{3.98}$$

$$\sigma_{u,ref}(t) = -\int_{t}^{\infty} e^{-\tilde{A}_u(t-\tau)} \tilde{B}_u \mathbb{Y}_d(\tau)\, d\tau. \tag{3.99}$$

The reference internal state trajectory, η_{ref} is then obtained as, (from Eq. (3.96))

$$\eta_{ref}(t) = U^{-1} \begin{bmatrix} \sigma_{s,ref}(t) \\ \sigma_{u,ref}(t) \end{bmatrix}, \tag{3.100}$$

and the inverse input, u_{inv} can be found from Eq. (3.90) as,

$$u_{inv}(t) = C_{inv}\eta_{ref}(t) + D_{inv}\mathbb{Y}_d(t). \tag{3.101}$$

Online inversion using finite preview information

Complete future information of the desired output and its derivatives, $\mathbb{Y}_d$ is required to obtain the bounded solution, $\sigma_{u,ref}$ to the unstable subsystem, as seen in Eq. (3.99).

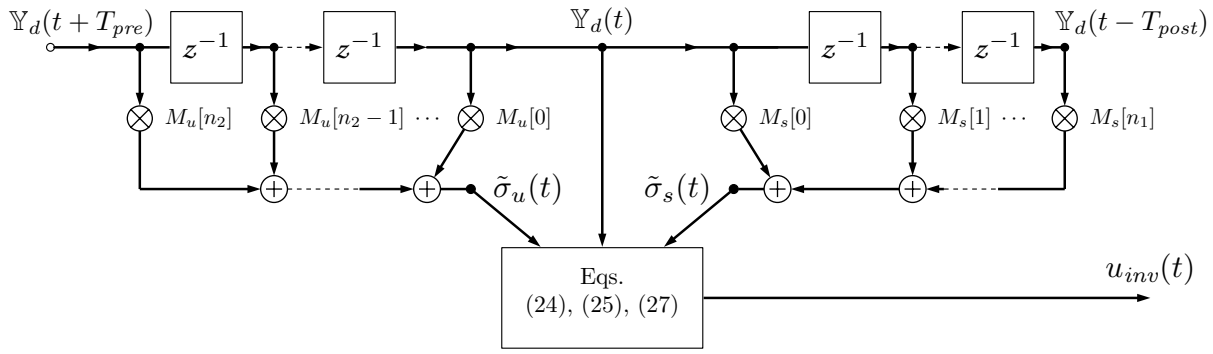


Figure 3.14: Online inversion of controlled-system, G using the preview-based online inversion scheme outlined in Section 3.4.1.

Thus, the desired output needs to be completely specified *a priori*, which restricts the use of inversion to off-line scenarios. The preview-based inversion scheme in [76] overcomes this

limitation by computing an approximated solution to the unstable subsystem, σ_u by using a finite amount of preview information of the desired output and its trajectories, $\mathbb{Y}_d$, given by,

$$\begin{aligned}\tilde{\sigma}_s(t) &= \int_{t-T_{post}}^{t} e^{\tilde{A}_s(t-\tau)}\tilde{B}_s\mathbb{Y}_d(\tau)\,d\tau \\ \tilde{\sigma}_u(t) &= -\int_{t}^{t+T_{pre}} e^{-\tilde{A}_u(t-\tau)}\tilde{B}_u\mathbb{Y}_d(\tau)\,d\tau,\end{aligned} \tag{3.102}$$

where T_{post} is the post-view time and T_{pre} is the preview time during which $\mathbb{Y}_d(\cdot)$ is known. A suitable numerical integration scheme, such as the zero-order hold (ZOH) integration [16] can be used with sampling time, T_s for on-line implementation such that the integration in Eq. (3.102) is evaluated as,

$$\begin{aligned}\tilde{\sigma}_s(t) &\approx \sum_{i=0}^{n_1}\mathcal{M}_s(i)\mathbb{Y}_d(t, i-n_1) \\ \tilde{\sigma}_u(t) &\approx \sum_{i=0}^{n_2}\mathcal{M}_u(i)\mathbb{Y}_d(t, n_2-i),\end{aligned} \tag{3.103}$$

where, $\mathbb{Y}_d(t,i) = \mathbb{Y}_d(t+iT_s)$ and, $n_1 = T_{post}/T_s$, $n_2 = T_{pre}/T_s$ are both assumed to be even integers. The weighting matrices for the above numerical integration scheme are given by,

$$\mathcal{M}_s(i) = [M_{s,1}]^{n_1-i}\cdot[M_{s,2}], \quad \mathcal{M}_u(i) = [M_{u,1}]^{n_2-i}\cdot[M_{u,2}], \tag{3.104}$$

where,

$$\begin{aligned}&= e^{\tilde{A}_sT_s}, \quad [M_{s,2}] = \tilde{A}_s^{-1}\left(e^{\tilde{A}_sT_s}-I\right)\tilde{B}_s \\ [M_{u,1}] &= e^{-\tilde{A}_uT_s}, \quad [M_{u,2}] = \tilde{A}_u^{-1}\left(e^{-\tilde{A}_uT_s}-I\right)\tilde{B}_u.\end{aligned} \tag{3.105}$$

The error in computing the inverse input, u_{inv} and therefore, the tracking error, $e_y(t) := \tilde{y}_d(t) - y(t)$ can be made arbitrarily small by choosing large enough preview time, T_{pre} and post-view time, T_{post}, and a sufficiently small sampling time, T_s.

Filtering the input signal

Since the input, $u(t) = u_h(t) + u_c(t)$ (see Fig. 3.13) tends to be noisy, a cascade of first-order low-pass filters is used to obtain a filtered input, $\overline{u}$, given by,

$$\dot{z}(t) = -az(t) + au(t), \tag{3.106}$$

$$\dot{\overline{u}}(t) = -a\overline{u}(t) + az(t), \tag{3.107}$$

as illustrated in Fig. 3.15. The combined signals, $\overline{u}, \dot{\overline{u}}$, as well as $\ddot{\overline{u}}$ given by,

$$\ddot{\overline{u}}(t) = -a\dot{\overline{u}}(t) + a\dot{z}(t), \tag{3.108}$$

i.e., the $r = 2$ time derivatives, $\mathbb{Y}_d$ of the input, $\tilde{y}_d(t) = \overline{u}$,

$$\mathbb{Y}_d(t) = \begin{bmatrix} \overline{u}(t) & \dot{\overline{u}}(t) & \ddot{\overline{u}}(t) \end{bmatrix}^T, \tag{3.109}$$

are fed into the inversion scheme depicted in Fig. 3.14, to obtain the inverse input, u_{inv}.

3.4.2 Experimental validation with multiple human subjects

The compensatory ILC update law in (3.55) was applied to a general linear controlled system (both minimum-phase and nonminimum-phase) with the online inverse control approach discussed in Section 3.4.

Participants

The output tracking experiment was conducted for nine ($n = 9$) human subjects, 7 male and 2 female with age ranging from 20-30 years, with the experimental conditions, i.e., system type (minimum-phase/nonminimum-phase) and target frequency bandwidth, $f_{BW} = 0.1, 0.2$ Hz, randomized for the nine subjects.

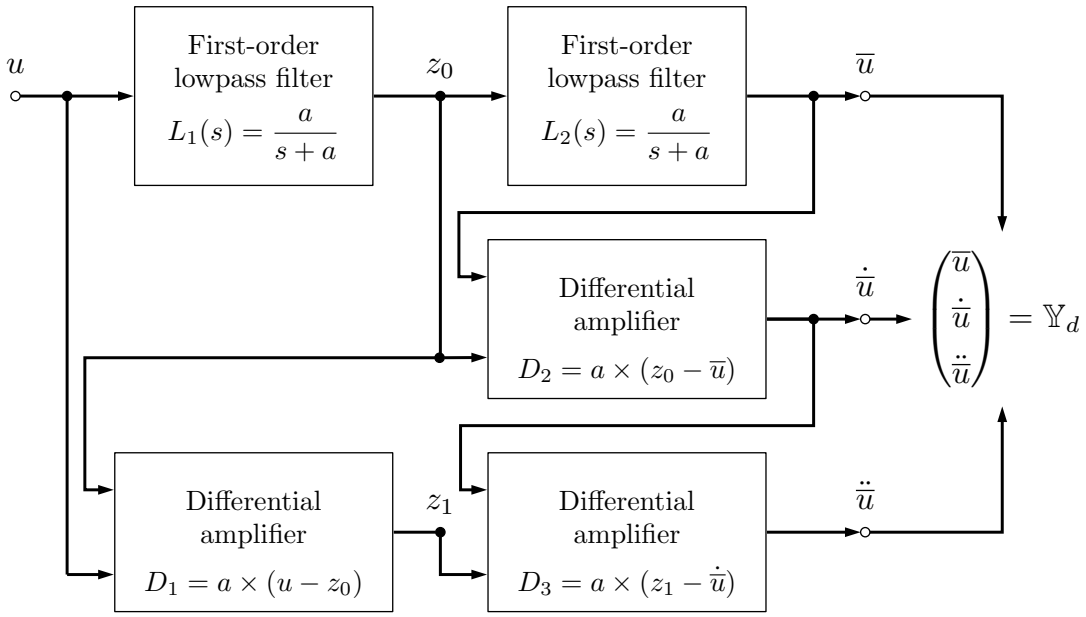


Figure 3.15: Cascade of two first-order low-pass filters to find the filtered signal, $\overline{u}$, and its time derivatives, $\dot{\overline{u}}$ and $\ddot{\overline{u}}$ according to Eqs. (3.107) and (3.108).

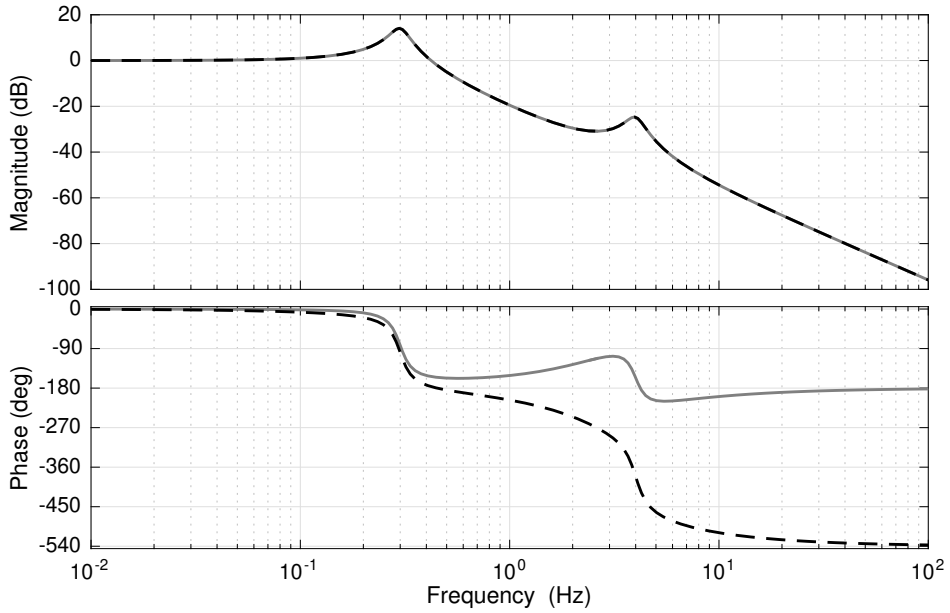


Figure 3.16: Frequency response function for the controlled-system, $G(\omega)$: (i) for $m = 0$ in Eq. (3.110) and G is minimum-phase (shown in gray solid lines), (ii) for $m = 1$ in Eq. (3.110) and G is nonminimum-phase (shown in black dashed lines). Note that the difference between the minimum-phase and nonminimum-phase systems is in the phase response.

Controlled-system dynamics

The controlled-system, G was chosen as a fourth-order linear time-invariant system of the form,

$$G(s) = K_{gain}\frac{s^2 \ + \ (-1)^m[2\xi_z\omega_{nz}s] \ + \ \omega_{nz}^2}{(s^2+2\xi_1\omega_{n1}s+\omega_{n1}^2)(s^2+2\xi_2\omega_{n2}s+\omega_{n2}^2)}, \tag{3.110}$$

where the parameters are as given in Table 3.4, with $\omega = 2\pi f$ rad/s, where f is the frequency in Hz. Note that the system, G is minimum-phase for $m = 0$, and nonminimum-phase for $m = 1$, as shown in Fig. 3.16. The form of the human-feedback response model, $\hat{H}_{fb}$ is unknown for such a higher-order controlled-system, G in Eq. (3.110), and therefore, this fourth-order controlled-system is used to illustrate the proposed approach.

The desired output trajectory, y_d for this pursuit control task was chosen to be a smooth reach and retract movement as shown in Fig. 3.17, similar to (3.61). Briefly, a preliminary acceleration profile is given by,

$$\frac{d^2\hat{y}_d}{dt^2}(t) = \begin{cases} 0, & t \in \left[0, \frac{T}{12}\right) \\ A^*\sin(\frac{6\pi}{T}t), & t \in \left[\frac{T}{12}, \frac{5T}{12}\right) \\ 0, & t \in \left[\frac{5T}{12}, \frac{7T}{12}\right) \\ -A^*\sin(\frac{6\pi}{T}t), & t \in \left[\frac{7T}{12}, \frac{11T}{12}\right) \\ 0, & t \in \left[\frac{11T}{12}, T\right] \end{cases} \tag{3.111}$$

with amplitude, $A^* = A(\frac{6\pi}{T})^2$, where A is the position amplitude, T is the time period of the trajectory in seconds, and the main frequency of the sinusoidal acceleration terms is

$$f_{BW} = 3/T. \tag{3.112}$$

In order to allow smooth pursuit tracking by the human user, the main frequencies in the desired trajectories need to be below the smooth-pursuit visual tracking limit of $f_T^* = 0.5$ Hz for typical healthy human users [8]. This was achieved by filtering the trajectory, $\hat{y}_d$ using

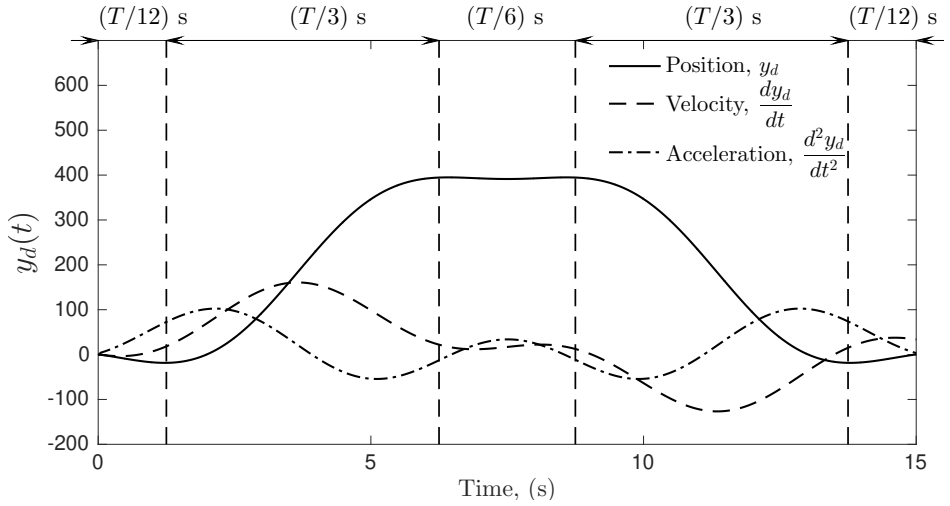


Figure 3.17: Desired output trajectory, y_d and its first and second time derivatives, Eq. (3.111), for time period, $T = 15$ seconds.

a first-order, low-pass filter, G_{fy} with a cutoff frequency at $f_{BW} = 3/T$ Hz, i.e., to keep the first three harmonics, where,

$$f_{BW} \leq f_T^* = 0.5 \text{ Hz}. \tag{3.113}$$

The output of the first-order filter is selected as the desired output trajectory, $y_d = [G_{fy}]\hat{y}_d$.

3.4.3 Experimental validation with single subject

Iterations

A total of $k = 26$ iterations were performed for each subject and experimental condition, divided into four control regimes: (I) human-only feedback with apparent system, $G_A = G$ and no machine learning, $u_{c,k} = 0$ for iterations $k \in [1, 5]$, (II) human-only feedback with apparent system, $G_A = G_r G$ and no machine learning, $u_{c,k} = 0$ for iterations $k \in [6, 15]$, (III) human-machine shared control with apparent system, $G_A = G_r G$, using the human-guided iterative impedance matching algorithm, for iterations $k \in [16, 25]$ with zero initial

Table 3.4: Parameter Values - Inversion-based ILC

Parameter	Value	Units	Description
Controlled-System, $G(s)$			
$\zeta_1, \zeta_2, \zeta_z$	$0.1, 0.1, 0.707$	–	Damping ratio
f_{n1}, f_{n2}, f_{nz}	$0.3, 5.0, 3.0$	Hz	Natural frequency
K_{gain}	$(-1)^m \left[\frac{(2\pi f_{n1})(2\pi f_{n2})}{(2\pi f_{nz})}\right]^2$	–	Gain
m	$\begin{bmatrix}0\\1\end{bmatrix}$	–	Minimum phase
		–	Nonminimum phase
Controlled-System Model, $\hat{G}(s)$ **(10% error)**			
$\hat{\zeta}_1, \hat{\zeta}_2, \hat{\zeta}_z$	$0.11, 0.11, 0.77$	–	Damping ratio
$\hat{f}_{n1}, \hat{f}_{n2}, \hat{f}_{nz}$	$0.33, 5.5, 3.3$	Hz	Natural frequency
$\hat{K}_{gain}$	$(-1)^m \left[\frac{(2\pi \hat{f}_{n1})(2\pi \hat{f}_{n2})}{(2\pi \hat{f}_{nz})}\right]^2$	–	Gain
Nominal Human Response model, $\hat{H}_{fb}(s)$			
K_p	1	-	Human Static Gain
τ_e	0.55	s	Effective time delay
f_c	0.8 to 1.0	Hz	Open-loop gain crossover frequency
τ_I	$1/2\pi f_c \approx 0.2$	s^{-1}	Equalization time constant
System Inversion and ILC			
T_{pre}	0.25	s	Preview time
T_{post}	0.4	s	Post-view time
ρ	0.5	-	Iteration gain
Desired Trajectory			
A	400	pixels	Amplitude
T	15 (fast), 30 (slow)	s	Time period
f_{BW}	$3/T$	Hz	Effective bandwidth (first 3 harmonics)

controller input, $u_{c,15} = 0$, and (IV) machine-only open loop control with apparent system, $G_A = G_r G$ ($k = 26$) with the learned control input, $u_{26} = u_{c,26}$ without human feedback, $u_{h,26} = 0$. A rest period was provided at the end of each iteration to enable resetting of the initial conditions. In addition to statistical data from these experiments, for illustrative purposes, detailed results with one person (Subject 8 whose performance is closest to the average performance, see Table 3.18) is presented with the following experimental conditions: (a) nonminimum-phase controlled-system, i.e., $m = 1$ in Eq. (3.110), and (b) reference trajectory bandwidth, $f_{BW} = 0.2$ Hz.

Convergence of system output to desired output

The output tracking performance improved significantly with use of the proposed inversion-based iterative learning algorithm, as shown in Fig. 3.19 for the target bandwidth frequency, $f_{BW} = 0.2$ Hz for the nine subjects. Note that the tracking error, $e_{26} = y_d - y_{26}$ at the end of human-machine shared control (with the inverse system included) is lower than the tracking error, $e_{15} = y_d - y_{15}$ for the human-only control with the inverse included, which in turn is significantly lower than the tracking error, $e_5 = y_d - y_5$ for human-only control without the inverse system.

To quantify this reduction, the tracking error, $e_k = y_d - y_k$ is quantified in terms of the normalized maximum error, $e_{max,k}$, and the normalized Root Mean Square (RMS) error, $e_{rms,k}$, respectively as,

$$e_{max,k} = \frac{\max_{t_i}[e_k(t_i)]}{A}, \quad e_{rms,k} = \frac{1}{A}\left(\frac{1}{N}\sum_{i=1}^{N} e_k^2(t_i)\right)^{1/2}, \tag{3.114}$$

for time, $t_i \in [kT, (k+1)T]$, where T is the time-period of each iteration k, N is the number of data samples in one iteration, and A is the amplitude of the desired output, given in Table 3.4. Fig. 3.20 depicts the evolution of the maximum error per iteration, $e_{max,k}$ for the four control regimes, $I - IV$ for Subject 8, for target frequency bandwidth, $f_{BW} = 0.2$ Hz.

Figure 3.18: Experimental results of ILC with general linear controlled systems: comparison of average normalized tracking performance and human-intent estimation (inside parentheses) at target bandwidth frequencies, $f_{BW} = 0.1, 0.2$ Hz.

Controlled Sys[1] G	**Bandwidth Frequency** f_{BW}	**Normalized Maximum error[2,3]**					**Normalized RMS error[2,3]**					**Human effort[2]**
		Human only	Human only (w/ inverse)	Shared	Machine open-loop	% reduction	Human only	Human only (w/ inverse)	Shared	Machine open-loop	% reduction	% reduction
		$\overline{e}_{max,I}$ (3.115) $\overline{\epsilon}_{max,I}$ (3.118)	$\overline{e}_{max,II}$ (3.115) $\overline{\epsilon}_{max,II}$ (3.118)	$\overline{e}_{max,III}$ (3.115) $\overline{\epsilon}_{max,III}$ (3.118)	$e_{max,26}$ (3.114) $\epsilon_{max,26}$ (3.116)	$\%E_{max,red}$ (3.125) $\%\mathcal{E}_{max,red}$ (3.119)	$\overline{e}_{rms,I}$ (3.115) $\overline{\epsilon}_{rms,I}$ (3.118)	$\overline{e}_{rms,II}$ (3.115) $\overline{\epsilon}_{rms,II}$ (3.118)	$\overline{e}_{rms,III}$ (3.115) $\overline{\epsilon}_{rms,III}$ (3.118)	$e_{rms,26}$ (3.114) $\epsilon_{rms,26}$ (3.116)	$\%E_{rms,red}$ (3.125) $\%\mathcal{E}_{rms,red}$ (3.119)	$\%UH_{red}$ (3.124)
MP	0.1 Hz	32 ± 18	15 ± 11	12 ± 21	2.2 ± 0.8	92 ± 3	12 ± 8	4 ± 3	3 ± 6	1.1 ± 0.3	88 ± 7	95 ± 9
		(16 ± 8)	(7 ± 4)	(9 ± 17)	(1.8 ± 0.7)	(85 ± 11)	(7 ± 3)	(3 ± 2)	(3 ± 6)	(1.0 ± 0.3)	(82 ± 14)	
	0.2 Hz	32 ± 16	12 ± 4	4 ± 2	2 ± 1	91 ± 5	13 ± 6	5 ± 2	1.4 ± 0.5	1.2 ± 0.3	89 ± 7	98 ± 1
		(16 ± 6)	(8 ± 3)	(2 ± 1)	(1.9 ± 0.9)	(85 ± 10)	(8 ± 3)	(4 ± 2)	(1.3 ± 0.5)	(1.0 ± 0.3)	(84 ± 10)	
NMP	0.1 Hz	52 ± 20	22 ± 14	7 ± 5	2 ± 1	95 ± 2	19 ± 8	7 ± 4	2 ± 1	1.2 ± 0.6	93 ± 3	97 ± 2
		(23 ± 7)	(11 ± 8)	(4 ± 3)	(2 ± 1)	(90 ± 3)	(10 ± 3)	(4 ± 3)	(2 ± 1)	(1.2 ± 0.5)	(88 ± 3)	
	0.2 Hz	56 ± 14	19 ± 4	6 ± 4	4 ± 2	93 ± 4	24 ± 6	8 ± 2	2 ± 1	1.7 ± 0.9	93 ± 4	96 ± 2
		(26 ± 6)	(8 ± 3)	(4 ± 2)	(3 ± 2)	(87 ± 8)	(12 ± 3)	(4 ± 1)	(2 ± 1)	(1.5 ± 0.8)	(87 ± 6)	

[1] MP – Minimum-phase system, G for $m = 0$ in (3.110); NMP - Nonminimum-phase system, G for $m = 1$ in (3.110).

[2] Values reported as $(\mu_x \pm \sigma_x) \times 100\%$, where $\mu_x = \sum_{i=1}^{n} x_i/n$ is the mean, and $\sigma_x = \sqrt{\sum_{i=1}^{n} x_i^2/(n-1)}$ is one standard deviation for $n = 9$ human subjects, for each performance metric, x.

[3] For each entry, top row: tracking error $e_k = y_d - y_k$, bottom row in parentheses: intent estimation error $\epsilon_k = y_d - \hat{y}_{d,k}$.

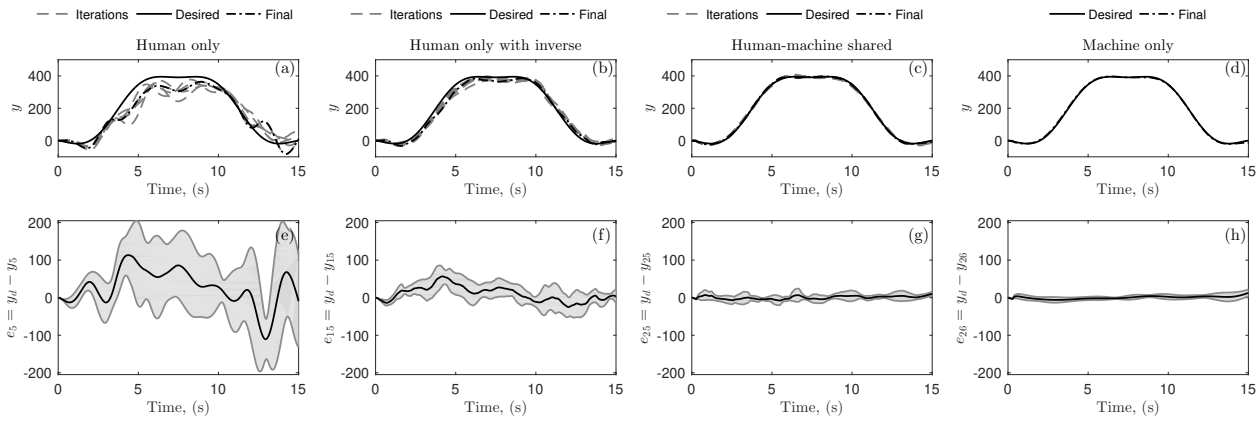


Figure 3.19: Convergence of iterative inversion-based impedance matching control for target bandwidth frequency, $f_{BW} = 0.2$ Hz and nonminimum-phase controlled-system, G with multiple human subjects; **(a)** system output, y_k during human only control with apparent system, $G_A = G$ ($k \in [1, 5]$), **(b)** system output, y_k during human only control with apparent system, $G_A = G_r G$ ($k \in [6, 15]$), **(c)** system output, y_k during shared human-machine iterative learning control with apparent system, $G_A = G_r G$ ($k \in [16, 25]$), **(d)** system output, y_k during machine-only control with apparent system, $G_A = G_r G$ ($k = 26$), **(e)** tracking error, $e_5 = y_d - y_5$ at the end of human-only control ($k = 5$), **(f)** tracking error, $e_{15} = y_d - y_{15}$ at the end of human-only control with inverse system included ($k = 15$), **(g)** tracking error, $e_{25} = y_d - y_{25}$ at the end of shared control ($k = 25$), and **(h)** tracking error, $e_{26} = y_d - y_{26}$ during machine-only control ($k = 26$). Solid lines depict mean trajectories over 9 subjects, and shaded region depicts one standard deviation over 9 subjects.

Note that the maximum error per iteration, $e_{max,k}$ decreases, with some fluctuations, over each successive control regime, $I - IV$. In Table 3.18, the tracking performance of the human-alone (with and without the inverse system included) is compared to the shared human-machine learning control in terms of the average normalized maximum error per iteration, $\overline{e}_{max,i}$, and the average normalized RMS error per iteration, $\overline{e}_{rms,i}$ (considering the last five iterations of each control regime),

$$\overline{e}_{max,i} = \frac{1}{5}\sum_{k=n_i-4}^{n_i} e_{max,k}, \quad \overline{e}_{rms,i} = \frac{1}{5}\sum_{k=n_i-4}^{n_i} e_{rms,k}, \tag{3.115}$$

where, $n_i = \{5, 15, 25\}$ is the last iteration of each control regime, $i = \{I, II, III\}$, respectively.

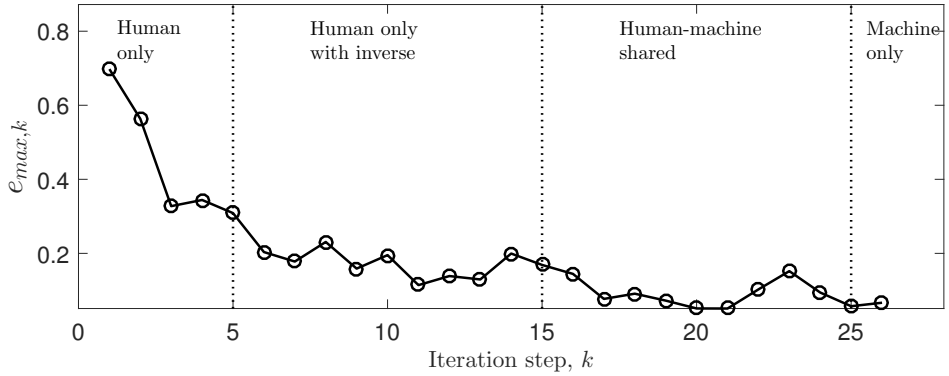


Figure 3.20: Maximum error per iteration, $e_{max,k}$, defined in Eq. (3.114), for target bandwidth frequency, $f_{BW} = 0.2$ Hz, nonminimum-phase controlled-system, G, for Subject 8.

Effect of using the inverse system model. The output tracking performance by the human user significantly improves by using the inverse system model, even without the assistance by the machine controller, as can be seen in Fig. 3.19 and Fig. 3.20. For example, for the target bandwidth frequency, $f_{BW} = 0.2$ Hz, and nonminium-phase controlled-system, the average normalized maximum error for human-only control without the inverse system model was $\overline{e}_{max,I} = 0.45$, which was reduced to $\overline{e}_{max,II} = 0.15$ with the inverse system model

included. This represents a percentage reduction of 66%. Similarly, the average normalized RMS error, $\overline{e}_{rms,I} = 0.20$ for human-only control without the inverse system model was reduced to $\overline{e}_{rms,II} = 0.07$ with the inverse system model included, which represents a percentage reduction of 67%. Thus, using the inverse system model, $G_r = \hat{G}^{-1}$ improved the manual tracking performance of the human user.

Effect of iterative learning by machine controller. The machine controller, G_c learned from the human input, u_h to control the apparent system, $G_A = G_r G$. From Fig. 3.20, note the further reduction of the normalized maximum error, $e_{max,k}$ during the application of the human-guided iterative impedance matching control, during iterations, $k \in [16, 25]$ for target bandwidth frequency, $f_{BW} = 0.2$ Hz, and nonminimum-phase controlled-system, for Subject 8. Note that the normalized maximum error, $e_{max,26} = 0.067$ for the machine open-loop control using the learned input, $u_{c,26}$ is similar to the error, $e_{max,25} = 0.055$ at the end of the shared human-machine iterative learning, which are both small.

With increasing iterations, the desired output trajectory, y_d, is closely approximated by the estimated human intent, $\hat{y}_{d,k}$ defined in Eq. (3.14), as seen in Fig. 3.21. The overall intent estimation error, $\epsilon_k = y_d - \hat{y}_{d,k}$, and the controller's intent-estimation error, $\tilde{\epsilon}_k = y_d - \tilde{y}_{d,k}$ are quantified in terms of the normalized maximum error, $\epsilon_{max,k}, \tilde{\epsilon}_{max,k}$, and the normalized Root Mean Square (RMS) error, $\epsilon_{rms,k}, \tilde{\epsilon}_{rms}$, respectively as,

$$\epsilon_{max,k} = \frac{\max_{t_i}[\epsilon_k(t_i)]}{A}, \quad \epsilon_{rms,k} = \frac{1}{A}\left(\frac{1}{N}\sum_{i=1}^{N}\epsilon_k^2(t_i)\right)^{1/2}, \tag{3.116}$$

$$\tilde{\epsilon}_{max,k} = \frac{\max_{t_i}[\tilde{\epsilon}_k(t_i)]}{A}, \quad \tilde{\epsilon}_{rms,k} = \frac{1}{A}\left(\frac{1}{N}\sum_{i=1}^{N}\tilde{\epsilon}_k^2(t_i)\right)^{1/2}, \tag{3.117}$$

for time, $t_i \in [kT, (k+1)T]$, where T is the time-period of each iteration k, N is the number of data samples in one iteration, and A is the amplitude of the desired output, given in Table 3.4. Also, the averages across each control regime are computed similar to the tracking error as,

$$\overline{\epsilon}_{max,i} = \frac{1}{5}\sum_{k=n_i-4}^{n_i}\epsilon_{max,k}, \quad \overline{\epsilon}_{rms,i} = \frac{1}{5}\sum_{k=n_i-4}^{n_i}\epsilon_{rms,k}, \tag{3.118}$$

where, $n_i = \{5, 15, 25\}$ is the last iteration of each control regime, $i = \{I, II, III\}$, respectively. Further, the reduction in the normalized maximum intent estimation error is computed as,

$$\%\mathcal{E}_{max,red} = \frac{\overline{\epsilon}_{max,I} - \epsilon_{max,26}}{\overline{\epsilon}_{max,I}} \times 100, \tag{3.119}$$

and, similarly for the normalized RMS intent estimation error,

$$\%\mathcal{E}_{rms,red} = \frac{\overline{\epsilon}_{rms,I} - \epsilon_{rms,26}}{\overline{\epsilon}_{rms,I}} \times 100. \tag{3.120}$$

On average, the proposed approach estimates the human intent to within 97% accuracy, i.e., intent estimation error is within 3% as seen from Table 3.18, which lists the overall intent estimation performance across the nine test subjects (shown inside parentheses). Similar results are seen for the normalized RMS intent estimation error. For example, across the nine subjects, with the nonminimum-phase controlled-system and input bandwidth frequency, $f_{BW} = 0.2$ Hz, the average normalized maximum intent estimation error with the human alone is $\overline{\epsilon}_{max,I} = 26 \pm 6\%$ which is reduced to $\epsilon_{max,26} = 3 \pm 2\%$ at the end of the shared learning at iteration $k = 26$. This represents a reduction in the estimation error of $\%\mathcal{E}_{max,red} = 87 \pm 8\%$, with similar performance seen across each of the tested experimental conditions for all nine subjects.

The reduction in estimated-intent error is reflected in the controller input u_c. Fig. 3.21 compares the maximum intent estimation error, $\epsilon_{max,k}$ (defined in Eq. (3.116)) with the the maximum controller's learned intent estimation error, $\tilde{\epsilon}_{max,k}$ (defined in Eq. (3.117)) for Subject 8. Since the controller input u_c is zero before iteration step $k = 15$, the reduction in estimation error, $\epsilon_{max,k}$ during these iteration steps ($k < 15$) reflects improvements due to human learning. The use of the inverse control reduces the intent estimation error during iterations $k \in [6, 15]$, where $\overline{\epsilon}_{max,II} = 5\%$, when compared to the first five iterations with the human alone where $\overline{\epsilon}_{max,I} = 31\%$. Further reduction is obtained with the iterative learning controller in iterations $k \in [16, 25]$ with the final overall intent estimation error at iteration

$k = 26$ given by, $\epsilon_{max,26} = 5\%$, representing a total reduction compared to the human only control of $\%\mathcal{E}_{max,red} = 83\%$. Moreover, for iterations $k \in [16, 25]$ the iterative controller's learned intent estimation error, $\tilde{\epsilon}_{max,k}$ shows how the proposed approach averages out (and thereby reduces) the impact of human-induced variations, e.g., seen at at iterations $k = 22, 23$ in the intent estimation error, $\epsilon_{max,k}$.

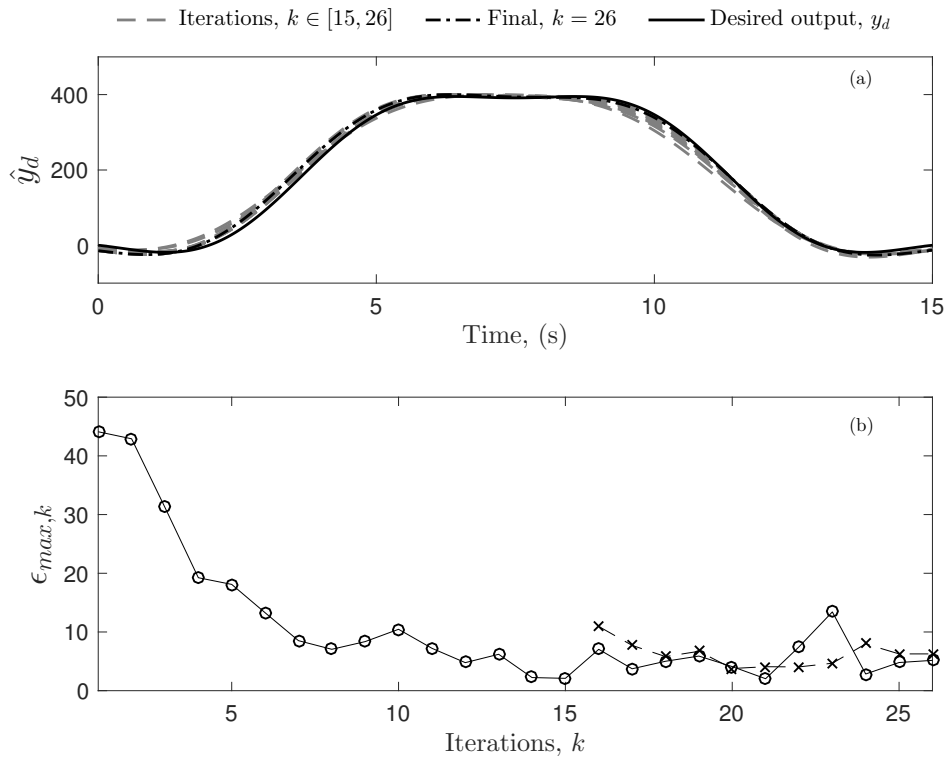


Figure 3.21: Estimation of human intent, **(a)** $\hat{y}_{d,k}(t)$ for iterations $k \in [1, 25]$, shown as dashed lines, compared to the desired output, $y_d(t)$, shown as a solid line, with the final learned human intent, $\hat{y}_{d,26}(t)$ at iteration $k = 26$, shown as a dash-dotted line, **(b)** maximum intent estimation error per iteration, $\epsilon_{max,k}$ for iterations $k \in [1, 26]$, defined in Eq. (3.116) shown with circular markers, and the controller's maximum intent-estimation error, $\tilde{\epsilon}_{max,k}$ for iterations $k \in [16, 26]$, defined in Eq. (3.117) shown with cross markers; for target bandwidth frequency, $f_{BW} = 0.2$ Hz, nonminimum-phase controlled-system, G, for Subject 8.

Additionally, the manual control effort by the human user, $\|u_{h,k}\|$ decreased with it-

erations $k \in [16, 25]$, while the machine controller effort, $\|u_{c,k}\|$ increased with iterations $k \in [16, 25]$, as shown in Fig. 3.22, where

$$\|u\| = \left(\frac{1}{N} \sum_{i}^{N} u^2(t_i) \right)^{1/2}, \quad \text{for } t_i \in [kT, (k+1)T], \tag{3.121}$$

for $u = \{u_{h,k}, u_{c,k}\}$, respectively, and N is the number of samples in each iteration, k.

For example, for Subject 8, the average human control effort in control regime I was

$$\|u_{h,I}\| = \frac{1}{5} \sum_{k=1}^{5} \|u_h, k\| = 230, \tag{3.122}$$

which was substantially reduced after the shared control regime III, i.e.,

$$\|u_{h,III}\| = \frac{1}{5} \sum_{k=21}^{25} \|u_h, k\| = 15, \tag{3.123}$$

for the target bandwidth frequency, $f_{BW} = 0.2$ Hz with the nonminimum-phase controlled-system, G. This represents a percentage reduction in the human effort,

$$\%UH_{red} = \frac{\|u_{h,I}\| - \|u_{h,III}\|}{\|u_{h,I}\|} \times 100, \tag{3.124}$$

of $\%UH_{red} \approx 94\%$.

The average normalized maximum error per iteration with the human alone (with the inverse controller) was $\overline{e}_{max,II} = 0.15$, which reduced to $e_{max,26} = 0.07$ at the end of shared human-machine learning, representing a percentage reduction of 56% (or, 19% further reduction from unassisted human-only control, $e_{max,I} = 0.45$). Similarly, the average normalized RMS error, $\overline{e}_{rms,II} = 0.07$ for human-only control (with the inverse controller) reduced to $e_{rms,26} = 0.03$ at the end of shared human-machine learning — a percentage reduction of 61% (or, 20% further reduction from unassisted human-only control, $e_{rms,I} = 0.20$). In total, at the end of the shared learning, there was a net percentage reduction in the average normalized maximum error, compared to human-only control (without any assistance),

$$\%E_{max,red} = \frac{\overline{e}_{max,I} - e_{max,26}}{\overline{e}_{max,I}} \times 100, \tag{3.125}$$

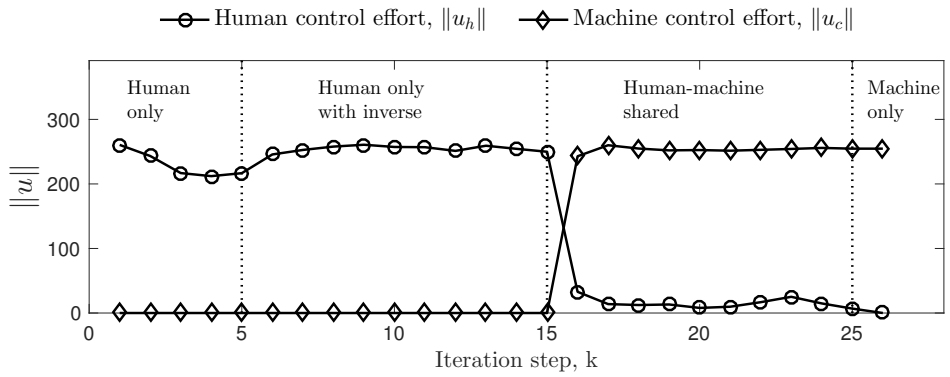


Figure 3.22: Control effort, $\|u\|$ for iterations $k \in [1, 26]$, defined in Eq. (3.121), with the human control effort, $\|u_{h,k}\|$ (circular markers), and the machine controller effort, $\|u_{c,k}\|$ (diamond markers), for target bandwidth frequency, $f_{BW} = 0.2$ Hz, nonminimum-phase controlled-system, G, for Subject 8.

of $\%E_{max,red} = 85\%$, and similarly, in the average normalized RMS error

$$\%E_{rms,red} = \frac{\overline{e}_{rms,I} - e_{rms,26}}{\overline{e}_{rms,I}} \times 100, \tag{3.126}$$

of $\%E_{rms,red} = 87\%$. Thus, the human user and the machine learned simultaneously to further reduce the tracking error, e with simultaneous reduction in human effort.

Effect of tracking frequency, f_{BW}. The normalized tracking performance of the human user was compared for tracking bandwidth frequencies, $f_{BW} = 0.1, 0.2$ Hz, and are presented in Table 3.18. The data suggests that there was no significant change in the tracking performance with varying target bandwidth frequency, f_{BW} over the selected frequency range. Consequently, there was a total percentage reduction in the normalized maximum error, $\%E_{max,red} \approx 94\%$ at the end of shared learning for each target bandwidth frequency, $f_{BW} = 0.1, 0.2$ Hz. Similar reduction was also seen in the average normalized RMS error, $\%E_{rms,red} \approx 93\%$ for each target bandwidth frequency, $f_{BW} = 0.1, 0.2$ Hz.

Thus, the results indicate that the proposed inversion-based iterative learning controller is able to learn from the human demonstrations to significantly improve the tracking performance (approximately 93% reduction), with simultaneous reduction in the human control

effort (about 96%), as compared to manual control by the human alone.

3.5 Improving tracking bandwidth with data

The proposed method is primarily a frequency response model identification method that uses available data to estimate the required frequency response functions used in the iterative robot-learning scheme in (3.21).

First, the human operator is allowed to perform the tracking task without any assistance from the robot controller, i.e., $u_c = 0$. Next, iterative robot-learning (using the update law in (3.21)) is carried out using nominal models, like in the previous subsections. The desired output trajectory, y_d during this step is chosen such that it has frequency harmonics only in the range of convergence of the nominal models (according to (3.25)), which is usually restricted to low frequencies. This ensures that output convergence is successfully achieved, i.e., $y_{k_f} \approx y_d$ for some iteration $k = k_f$. Then, the human input, u_h is given by (3.11) as,

$$u_{h,k_f}(\omega) \;=\; (G_{H,1}(\omega) - G_{H,2}(\omega))\, y_{k_f}(\omega), \tag{3.127}$$

which implies, the human models can be computed for frequencies ω as,

$$\hat{G}_{H,1}(\omega) - \hat{G}_{H,2}(\omega) = u_{h,k_f}(\omega)/y_{k_f}(\omega) \approx 0, \tag{3.128}$$

since, at convergence, $u_{h,k_f}(\omega) \approx \lim_{k\to\infty} u_{h,k}(\omega) = 0$, as mentioned in Lemma 1 in Section 3.2.2. Using data from the initial human-only trials, and the estimated human intent at the end of convergence of robot-learning, compute

$$\hat{G}_{fb}(\omega) \;=\; \frac{1}{N_1}\sum_{k=1}^{N_1} y_k(\omega)/y_{k_f}(\omega). \tag{3.129}$$

i.e., the mean of the ratio of the output at each iteration, y_k, to the converged output, y_{k_f}, for iterations $k \in [1, N_1]$, where N_1 is the number of human-only iterations.

This section is derived from the publication: Rahul B Warrier and Santosh Devasia. "Data-based Iterative Human-in-the-loop Robot-Learning for Output Tracking". In: *IFAC-PapersOnLine* 50.1 (2017), pp. 12113–12118

Also, modeling error, Δ defined in (3.26) can be computed with the assumptions, $\hat{G}_{H,1}(\omega) = G_{H,1}(\omega)$ and $\hat{G}_{H,2}(\omega) = G_{H,2}(\omega)$, i.e.,

$$\begin{aligned} \Delta(\omega) &= \Delta_m(\omega)e^{j\Delta_\theta(\omega)} \approx G_{fb}(\omega)/\hat{G}_{fb}(\omega) \\ &\approx \frac{1}{N_1}\sqrt{\sum_{k=1}^{N_1}\left(\frac{y_k(\omega)}{y_{k_f}(\omega)} - \hat{G}_{fb}(\omega)\right)^2}, \end{aligned} \tag{3.130}$$

i.e., the standard deviation (from the mean) of the ratio of the output at each iteration, y_k, to the converged output, y_{k_f}, for iterations, $k \in [1, N_1]$. Finally, the iteration gain, ρ can be updated using (3.25) as,

$$\rho(\omega) = \rho^*(\omega)/2 = \cos(\Delta_\theta(\omega))/\Delta_m(\omega). \tag{3.131}$$

3.5.1 Experimental Setup

A MICO2 robot arm, manufactured by Kinova Robotics was used in the human-robot experiments, where the objective was to follow a specified output trajectory, known to the human operator but not known to the robot. Specifying the desired output $\vec{q}_d$ to the human operator allowed the evaluation of the convergence with the proposed iterative approach and quantification of the error in output tracking. The experimental setup is shown in Fig. 3.23.

A flexible structure consisting of a coil spring was attached rigidly to the end-effector of the MICO2 robot arm. A laser attached to the end of the flexible structure projected a red dot on to the screen — the position of the red dot on the screen was the output vector,

$$\vec{q} = [q_x, q_y]^T. \tag{3.132}$$

In what follows, vectors will be denoted in bold; non-bold symbols, such as y, will denote scalars in either the x or y axes, with the direction specifically clarified with a subscript, e.g., q_x, when necessary. An LCD projector displayed the desired trajectory $\vec{q}_d$, represented by the center of a green circle. A camera was used to measure the real-time positions of

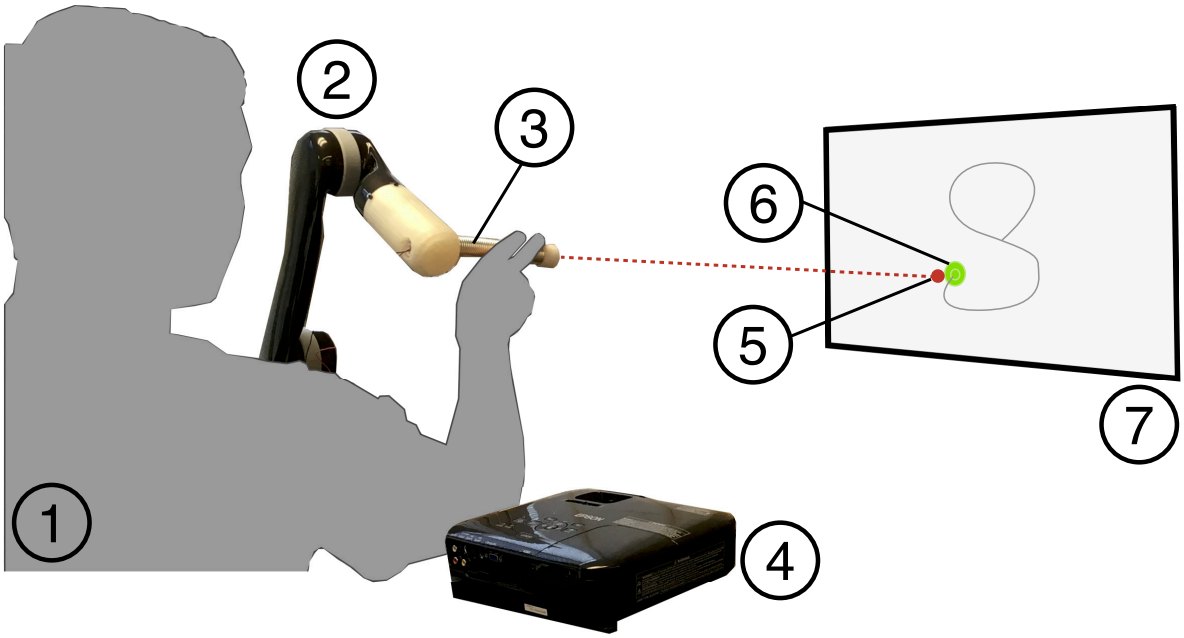


Figure 3.23: Experimental setup for the human-robot output-tracking experiment: (1) human operator; (2) Kinova Mico 4-dof robot arm; (3) flexible end-effector with laser pointer embedded in the tip; (4) LCD projector used to display the desired output $\mathbf{q_d}$ (green circle) (6) and Logitech C920 web-camera (positioned adjacent to the projector (not shown)) used to sense the output $\mathbf{q}$ (laser) (5) and the desired output $\mathbf{q_d}$ (green circle) (6); and (7) screen where projected image is displayed.

the laser dot $\vec{q}$ and the green circle $\vec{q}_d$ at a frame rate of 30Hz and the signals were filtered with a zero-phase second-order Butterworth filter with cutoff frequency of 5 Hz [13]. Image processing tools used, e.g., to find the center of the green circle for the desired output $\vec{q}_d$, were from the OpenCV library [12]. The objective was to track the center of the green circle $\vec{q}_d$, whose trajectory was specified but not known a priori to the human operator (and was not available to the robot controller), using the red laser dot $\vec{q}$.

3.5.2 Flexible human-robot interface

The proposed collaborative human-robot control scheme was applied through a flexible-spring system shown in Fig. 3.24. The flexibility of the spring human-robot-interface allowed both the human and robot to simultaneously control the total output, $\vec{q}$, namely the position of the red dot on the screen This facilitated demonstration of the task (such as tracking the desired output) by the human operator even when the robot was operating independently, e.g., during the initial trial $k = 0$ when the robot had no information about the desired output.

The robot end-effector was attached to one end of the flexible structure, i.e., end (A) in Fig. 3.24, while the human operator deflected the structure relative to end (A) by applying a displacement, δ_h at the other end, i.e., end (B) in Fig. 3.24. Thus, relative to the end (A), the flexible structure can be considered as a cantilever beam with a concentrated load, P acting at end (B) due to the human operator. Then, the displacement of end (B) relative to end (A), i.e., δ_h can be related to the slope, θ as, (using the Timoshenko beam theory [59]) $\theta \approx \frac{3\delta_h}{2l}$, where, l is the length of the flexible structure, and $\delta_h = (\delta_{h,x}, \delta_{h,y})$ are deflections of end (B) along each coordinate axis (x, y). The position y of the red dot on the screen can then be obtained (in either the x or y axes) as,

$$q = q_r + q_h, \quad \text{where } q_h = f(\delta_h). \tag{3.133}$$

When the distance L to the screen is large compared to the length l of the flexible structure,

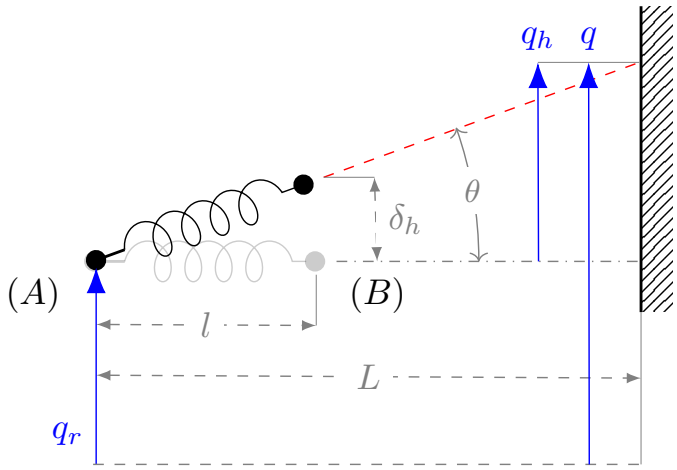


Figure 3.24: Collaborative control of the flexible structure. (a) Robot end-effector controlled position, $q_r = (x_r, y_r)$ of end (A), (b) Human operator's deflection with respect to end (A), $\delta_h = (\delta_{x,h}, \delta_{y,h})$, (c) Total position, $q = q_r + q_h$, where $q_h = f(\delta_h)$, as in (3.134).

i.e., $L \gg l$, the relationship between the deflection, δ_h and the screen position, q_h can be simplified to,

$$q_h = f(\delta_h) \approx L \tan\theta = L \tan\left(\frac{3\delta_h}{2l}\right) \approx L\left(\frac{3\delta_h}{2l}\right), \tag{3.134}$$

where the angle θ is as illustrated in Fig. 3.24.

3.5.3 Screen To Robot Transformation

The screen where the position of the robot is measured was not fully parallel to the x, y axes of the robot end effector motion. Hence a transformation was used to map the observed positions on the screen $\vec{q}^{\mathcal{S}}$ to the robot x, y coordinates $\vec{q}^{\mathcal{R}}$:

$$\vec{q}^{\mathcal{R}} = \vec{A}\vec{q}^{\mathcal{S}} + \vec{b} \tag{3.135}$$

where the elements of the 2×2 matrix $\mathbf{A}$ and 2×1 vector $\mathbf{b}$ were found using a least squares fit with experimentally measured $\mathbf{q^R}$ and $\mathbf{q^S}$. Note that this map is invertible, i.e.,

$\vec{A}$ is invertible. In the following, most data are presented in the robot coordinate frame. Therefore, the superscript R in (3.135) for the robot frame is not added for notational ease. However, the superscript S for the screen frame is explicitly denoted.

3.5.4 Robot position control

The position of the robot arm's end-effector is directly controlled in Cartesian space using the built-in Application Programming Interface (API). The API, written in C++, provides basic functions to control the robot arm, and recommends velocity control to achieve smooth trajectories in Cartesian end-effector space. Moreover, the linear speed of the robot is restricted to 0.2 m/s, which aids a human operator to safely interact with the robot.

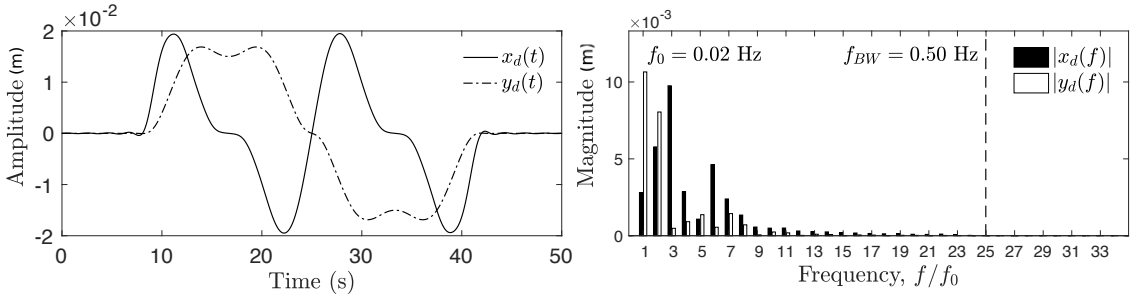
An outer proportional feedback loop was designed around the commanded position, u to operate the robot with the Kinova velocity controller, resulting in the closed-loop system, decoupled along each dimension of the Cartesian end-effector space (x, y),

$$\hat{G}_x(s) = \frac{y_x(s)}{u_x(s)} = \frac{P \cdot \frac{\tau}{s(s+\tau)}}{1 + P \cdot \frac{\tau}{s(s+\tau)}} = \frac{y_y(s)}{u_y(s)} = \hat{G}_y(s), \tag{3.136}$$

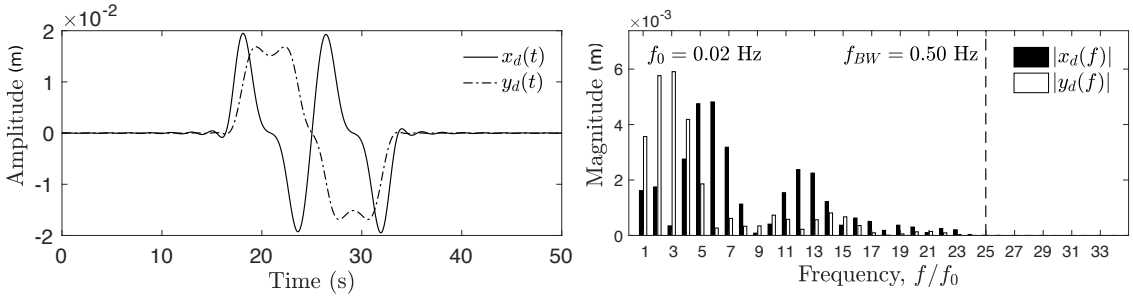
for $s = j\omega$ and $j = \sqrt{-1}$. The proportional controller was selected as $P = 2.0$, and a nominal parameter value of $\tau \approx 8.0$ was obtained by characterizing the step response of the Kinova API velocity controller. More details about the C++ implementation of the robot end-effector Cartesian controller are provided in the Appendix Section A.

Iterations:

A total of $k = 22$ iterations were performed. Two reference trajectories were considered: (1) $\vec{y}_{d,1} = x_{d,1}\hat{e}_x + y_{d,1}\hat{e}_y$ with significant harmonics up to ≈ 0.2 Hz, see Fig. 3.25(a) and (2) $\vec{y}_{d,2} = x_{d,2}\hat{e}_x + y_{d,2}\hat{e}_y$ with significant harmonics up to ≈ 0.5 Hz, as shown in Fig. 3.25(b). For iterations, $k \in [1, 11]$, the desired output trajectory to be tracked was chosen as, $\vec{y}_d = \vec{y}_{d,1}$, divided into 3 control regimes: (I) human-only control with no assistance from the robot



(a) Slow Trajectory, $\vec{y}_{d,1}$



(b) Fast Trajectory, $\vec{y}_{d,2}$

Figure 3.25: Reference trajectories to be tracked: (a) $\vec{y}_{d,1}$ for iterations, $k \in [1, 11]$, and (b) $\vec{y}_{d,2}$ for iterations, $k \in [12, 22]$, shown in time-domain (left) and in frequency-domain (right), i.e., magnitude of harmonics at multiples of fundamental frequency, $f_0 = 1/T = 0.02$ Hz.

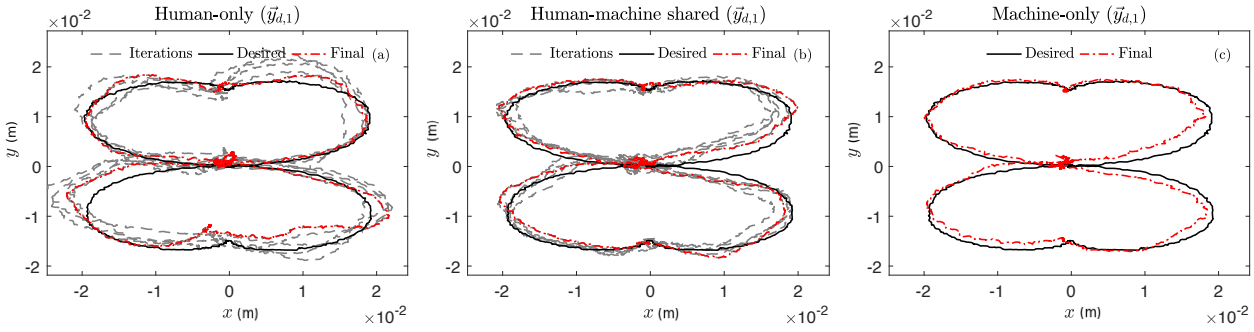


Figure 3.26: Output tracking convergence with desired output trajectory, $\vec{y}_{d,1}$ shown in black, and output, $\vec{y}_k$ (a) during human-only control for iterations, $k \in [1, 5]$ shown as gray dashed lines with final iteration shown as red dash-dotted lines, (b) during human-machine shared control for iterations, $k \in [6, 10]$ shown as gray dashed lines with final iteration shown as red dash-dotted lines, (c) during machine-only control using learned input for iteration $k = 11$ shown as red dash-dotted line.

controller, i.e., $u_{c,k} = 0$ for iterations, $k \in [1, 5]$, (II) human-machine shared control with iterative robot-learning using nominal models, for iterations, $k \in [6, 10]$, with zero initial controller input, $u_{c,5} = 0$, (III) machine-only open-loop control for $k = 11$, with learned control input, $u_{11} = u_{c,11}$, without human feedback, i.e., $u_{h,11} = 0$. Similarly, for iterations $k \in [12, 22]$, the desired output trajectory was chosen as $\vec{y}_d = \vec{y}_{d,2}$, and control regimes (I), (II) and (III) were used for iterations, $k \in [12, 16]$, $k \in [17, 21]$, and $k = 22$, respectively.

Further, the following two cases were considered for iterations, $k \in [17, 21]$: (a) Case 1: iterative robot-learning using nominal models, (b) Case 2: iterative robot-learning using models obtained by the proposed model update algorithm.

Output convergence with nominal models:

First, the iterative robot-learning scheme in (3.21), was applied to track the desired output trajectory, $\vec{y}_{d,1}$ using the nominal models described in Sections 3.5.4 and 3.3.10. Output convergence was achieved at the end of the human-robot shared learning as shown in

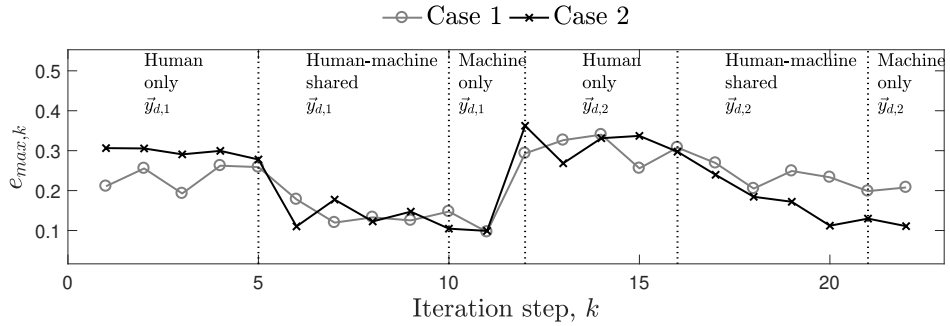


Figure 3.27: Normalized maximum absolute error per iteration, $e_{max,k}$ as defined in (3.137) for Case 1 (shown in gray with circular markers) and Case 2 (shown in black with x-markers).

Fig. 3.26. To quantify the performance of the iterative robot-learning scheme, the tracking error, $\vec{e}_k = \vec{y}_d - \vec{y}_k$, is quantified in terms of the normalized maximum absolute error per iteration, $e_{max,k}$, and the normalized Root Mean Square (RMS) error per iteration, $e_{rms,k}$ given by,

$$e_{max,k} = \frac{\max_{t_i} |\vec{e}_k(t_i)|}{A}, \quad e_{rms,k} = \frac{1}{A}\left(\frac{1}{N}\sum_{i=1}^{N} |\vec{e}_k(t_i)|^2\right)^{1/2}, \tag{3.137}$$

for $t_i \in [kT, (k+1)T]$, where T is the time-duration of one iteration, N is the number of samples per iteration, and A is the amplitude of the desired output trajectory, $\vec{y}_d$. Table 3.5 tabulates the normalized tracking performance for each case.

Fig. 3.27 depicts the evolution of the normalized maximum absolute error, $e_{max,k}$ for each control regime (I)-(III), and for desired output trajectories, $\vec{y}_{d,1}$ for iterations, $k \in [1, 11]$, and $\vec{y}_{d,2}$ for iteration, $k \in [12, 22]$. Note that tracking performance for Case 1 and 2 are similar at the end of convergence with desired output trajectory, $\vec{y}_{d,1}$ at iteration $k = 11$, e.g., $e_{max,11} = 0.097$ for Case 1, compared to $e_{max,11} = 0.099$ for Case 2. This is expected since the desired output trajectory, $\vec{y}_{d,1}$ has significant harmonics up to ≈ 0.2 Hz, as shown in Fig. 3.25(a), which is within the convergence criteria for the iterative robot-learning scheme with nominal models.

Finally, at the end of convergence at iteration $k = 11$, the proposed model-update method described in the beginning of Section 3.5 was applied to obtain the updated closed-loop feedback model, $\hat{G}_{fb}$ as shown in Fig. 3.28, and associated modeling error, Δ and iteration gain, ρ as shown in Fig. 3.29.

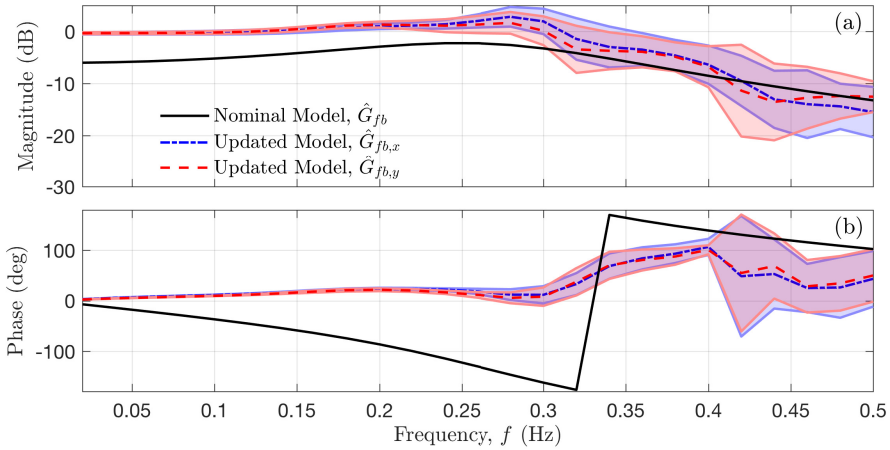


Figure 3.28: (a) Magnitude and (b) phase of the frequency response of the updated feedback transfer function model, $\hat{G}_{fb}$ (shown in blue along the x-direction, and in red along the y-direction, mean as dashed line and ± 1 std. dev. as patched area) obtained by the proposed model-update method compared to the nominal model, $\hat{G}_{fb}^{nom}$ (shown in black).

Output Convergence at higher frequencies:

In the case of learning to track $\vec{y}_{d,2}$ in iterations, $k \in [12, 22]$, the proposed model-update method (Case 2) showed significant improvement in tracking performance compared to using nominal models (Case 1) in the iterative robot-learning scheme, as shown in Fig. 3.27 and Table 3.5. For example, at the end of convergence in iteration $k = 22$, $e_{max,22} = 0.208$ for Case 1, compared to $e_{max,22} = 0.083$ for Case 2, which represents a percentage reduction compared to human-only control in control regime I (with trajectory, $\vec{y}_{d,2}$) of 32% for Case 1,

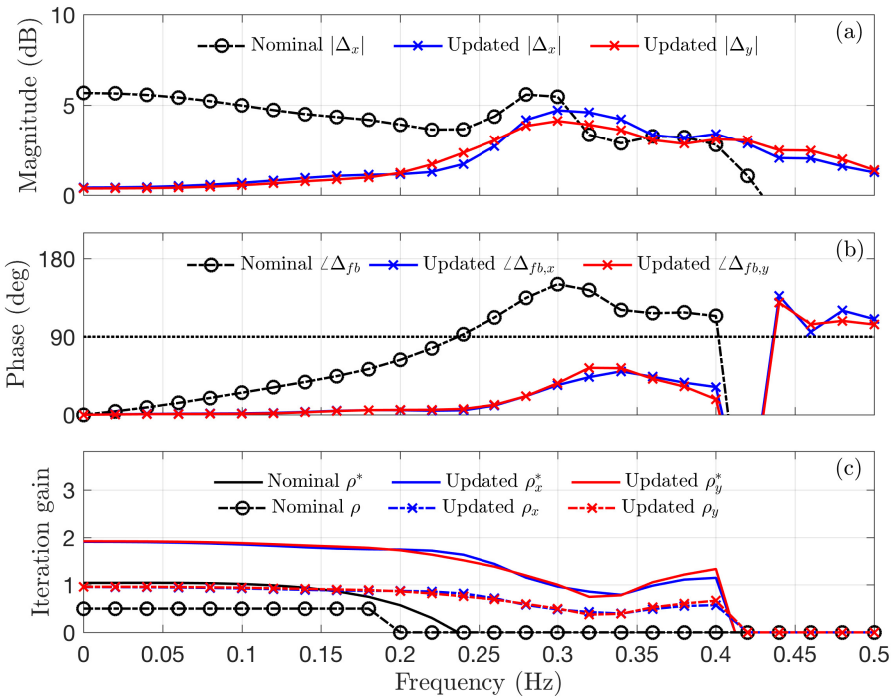


Figure 3.29: (a) Magnitude and (b) phase of the modeling error, Δ computed according to (3.130), (c) maximum possible iteration gain, ρ^* computed according to (3.25) (shown in solid lines) and chosen iteration gain, ρ computed according to (3.131) (shown as dash-dotted line with x-markers). Results along the x-direction are shown in blue, and along the y-direction are shown in red.

and 65% for Case 2, respectively. This is $\approx 103\%$ increase in tracking error reduction. This is because the nominal models result in modeling error, Δ that restricts convergence up to only 0.2 Hz (i.e., $\angle\Delta_\theta(f) > \pi/2$ for $f > 0.2$ Hz, as shown in Fig. 3.29), but the desired output trajectory, $\vec{y}_{d,2}$ has significant harmonics up to ≈ 0.5 Hz, as shown in Fig. 3.25(b). This implies, with the nominal models, the iterative robot-learning scheme does not converge at frequencies higher than 0.2 Hz In contrast, the updated models have reduced modeling error, increasing the range of convergence up to $f_c = 0.5$ Hz, resulting in improved tracking performance for $\vec{y}_{d,2}$, as shown in Fig. 3.29.

Thus, the proposed data-based model update algorithm improves the convergence of the iterative robot-learning approach by increasing the frequency range of convergence (from 0.2 Hz to 0.5 Hz) while simultaneously doubling the overall tracking performance ($\approx 103\%$ increase in tracking error reduction).

Table 3.5: Comparison of average normalized tracking performance for Case 1 (nominal models) and Case 2 (updated models).

Trajectory	Case	Max error				RMS error			
		Control Regime				Control Regime			
		I[1]	II[1]	III[1]	% red.[3]	I[2]	II[2]	III[2]	% red.[3]
$\vec{y}_{d,1}$	1	0.24	0.14	0.10	59	0.07	0.05	0.04	41
	2	0.30	0.13	0.10	67	0.09	0.05	0.04	60
$\vec{y}_{d,2}$	1	0.30	0.23	0.21	32	0.09	0.08	0.08	11
	2	0.32	0.17	0.11	65	0.10	0.06	0.05	52

[1] Mean value of normalized maximum absolute tracking error, $e_{max,k}$, defined in (3.137), in each control regime, (I)-(III).

[2] Mean value of normalized RMS tracking error, $e_{rms,k}$, defined in (3.137), in each control regime, (I)-(III).

[3] Percentage reduction in tracking error ($e_{max,k}$ and $e_{rms,k}$) at the end of convergence in III compared to mean value in I, i.e., $100 \times (I - III)/I$.

Chapter 4

ONE-SHOT INTENT ESTIMATION - FREQUENCY-DOMAIN

4.1 Introduction

In this chapter, we consider the primary research problem of a robotic controller inferring an output trajectory from the demonstrations of a human-in-the-loop (HIL) operator, but considering only a single human demonstration as opposed to an iterative learning framework as discussed in Chapter 3.

In previous chapters, the proposed methods required nominal human models to estimate intent, which assumes that the behavior of the individual human operator adheres to population averaged models. For example, in previous work [69], human intent estimation was addressed by inverting some known nominal human response models which are available for specific types of controlled systems, but not for general task conditions. An extension of this approach to more general controlled systems was discussed in [67] where an inverse controller was used to modify the controlled system, thereby simplifying the apparent controlled system perceived by the human operator whose response could still be described by the available nominal models. But, in the absence of a priori information about the controlled system, or if the controlled system is difficult to model, then the introduction of the inverse controller will adversely affect the online tracking performance of the human operator, and the iterative controllers to estimate intent, described in [67] will be restricted to smaller learning rates, i.e., more number of training trials will be required to converge to a solution.

In contrast, this chapter studies the development of a kernel-based model estimation method which learns personalized feedback and feed-forward response models of an individual human operator, first introduced in [70]. The novelty of this method is that it does not require

multiple iterations of expert input/output data to estimate the system models, in addition to providing an estimate of the modeling uncertainty at each frequency. This article provides the following main contributions in contrast to the work in [70]:

1. Bounds on the modeling uncertainty are developed in this article which guarantee that the estimated intent using the computed models yields reduced intent estimation error compared to the human-demonstrated output.

2. Multiple human-subjects experiments are used to validate this conservative model-inversion approach with an output-tracking experiment that models a tele-operated surgical task in tracking an intended output trajectory.

The proposed intent estimation approach was validated using human-in-the-loop output tracking experiments with multiple human subjects. Results indicate that the computed human models are able to successfully predict the intended goal-trajectory y_d, on an average reducing the intent estimation error by 30% compared to considering the human-demonstrated output y as the intended goal-trajectory y_d. More importantly, it was shown that the proposed method ensured that the intent estimation error was greater than or equal to the human-demonstrated tracking error even under significant modeling uncertainty.

The rest of the chapter is organized as follows. Section 4.2 formalizes the problem and explains the rationale for the proposed method. Section 4.4 presents the robustness conditions on the modeling uncertainty in terms of guaranteeing better intent estimation accuracy compared to the human demonstrations. Also, a non-parametric regression technique in the complex domain called complex-valued Gaussian process regression (CGPR) is discussed in Section 4.4.4 to obtain mean and variance estimates of the human-in-the-loop dynamics models from training data. Finally, the proposed method is validated using multiple human subject experiments and the experimental setup and results are discussed in Section 4.5. Conclusions and future work are summarized in Section 4.6.

4.2 Problem Formulation

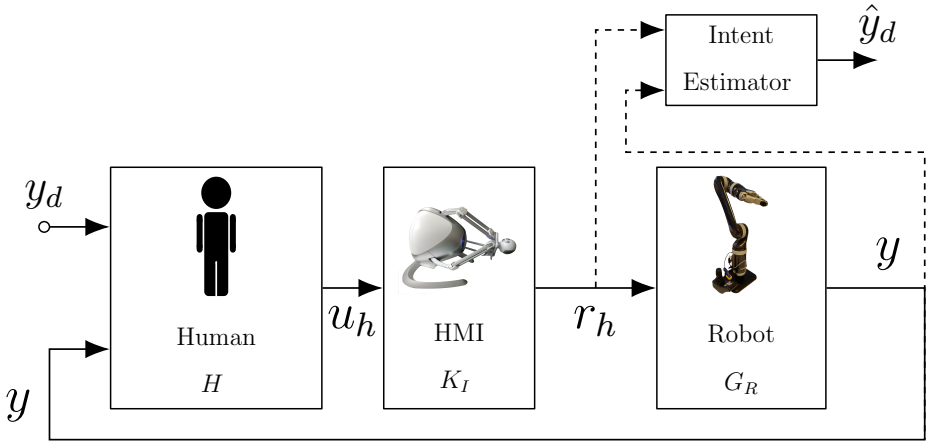


Figure 4.1: Block diagram schematic of human-in-the-loop output tracking: human operator H applies an input u_h to the human-machine interface K_I which applies input r_h to the robot system G_R with the aim of setting the system output $y \approx y_d$ the intended goal trajectory.

The human intent in the current setup is the desired output y_d, that the human operator is trying to track with the output y of the system P, as shown in Figure 4.1, which may be described in frequency domain as,

$$y(\omega) = P(\omega)r_h(\omega) = G(\omega)y_d(\omega) \tag{4.1}$$

where P is the controlled system (plant) frequency response and G is the closed-loop frequency response function.

Specifically, the following are the main research questions addressed in this section:

(RQ-1) Can we infer the intended goal y_d from the human demonstration y by inverting non-parametric transfer function models?

(RQ-2) When is model-inversion better for intent estimation compared to the average human demonstration?

Problem statement: Estimate the intended goal y_d of the human-in-the-loop operator using the available human input r_h and the system output y.

4.3 Solution Approach

4.3.1 Intent Estimation by model-inversion

The intended goal y_d may be inferred from the human input r_h and the system output y as,

$$y_d(\omega) = G_{inv,1}(\omega) r_h(\omega) + G_{inv,2}(\omega) y(\omega), \tag{4.2}$$

where, $G_{inv,1}, G_{inv,2}$ are transfer functions that relate the specific signals at frequency ω. Further, the human input r_h may be inferred from (4.1) as,

$$r_h(\omega) = P^{-1}(\omega) y(\omega), \tag{4.3}$$

which when substituted into (4.2) results in,

$$\begin{aligned} y_d(\omega) &= \left[G_{inv,1}(\omega) P^{-1}(\omega) + G_{inv,2}(\omega) \right] y(\omega), \\ &= G^{-1} y(\omega), \end{aligned} \tag{4.4}$$

where G is the transfer function between the intended goal y_d and the system output y.

Thus, an estimate $\hat{y}_d$ of the intended goal y_d may be computed as,

$$\begin{aligned} \hat{y}_d(\omega) &= \hat{G}_{inv,1}(\omega) r_h(\omega) + \hat{G}_{inv,2}(\omega) y(\omega), \\ &= \hat{G}^{-1}(\omega) y(\omega), \end{aligned} \tag{4.5}$$

where, $\hat{G}^{-1}, \hat{G}_{inv,1}, \hat{G}_{inv,2}$ are known/estimated models of $G^{-1}, G_{inv,1}, G_{inv,2}$, defined in (4.2) and (4.4), respectively.

4.3.2 Connection to human-in-the-loop dynamics models

In general, the human input r_h is modeled as a combination of three terms [72], as shown in Fig. 4.2, described in frequency domain as,

$$r_h(\omega) = G_{H,ff}(\omega) y_d(\omega) + G_{H,fb}(\omega) e(\omega) + G_{H,in}(\omega) y(\omega), \tag{4.6}$$

or, by rearranging the terms,

$$r_h(\omega) = G_{H,1}(\omega)y_d(\omega) - G_{H,2}(\omega)y(\omega), \tag{4.7}$$

where the frequency response functions $G_{H,1}, G_{H,2}$ are given by,

$$G_{H,1}(\omega) = G_{H,\mathit{ff}}(\omega) + G_{H,\mathit{fb}}(\omega), \tag{4.8}$$

$$G_{H,2}(\omega) = G_{H,\mathit{fb}}(\omega) - G_{H,in}(\omega), \tag{4.9}$$

where,

$$G_{H,\mathit{ff}} = K_1(\omega)u_{h,\mathit{ff}}(\omega)/y_d(\omega) = K_1(\omega)H_{\mathit{ff}}(\omega), \tag{4.10}$$

$$G_{H,\mathit{fb}} = K_1(\omega)u_{h,\mathit{fb}}(\omega)/e(\omega) = K_1(\omega)H_{\mathit{fb}}(\omega), \tag{4.11}$$

$$G_{H,in} = K_1(\omega)u_{h,in}(\omega)/y(\omega) = K_1(\omega)H_{in}(\omega), \tag{4.12}$$

where K_1 represents the human-machine interface, H_{ff}, H_{fb}, and H_{in} are the feed-forward, feedback, and inner-loop response transfer functions of the human operator's dynamics, respectively. It has been shown previously that with expertise, the human feed-forward response $G_{H,\mathit{ff}} \to P^{-1}$ [74] while the human feedback response $G_{H,\mathit{fb}}$ can be described by some empirical models for simple types of controlled systems P, e.g., the crossover pilot model [11].

Then, the closed-loop frequency response function G may be expressed in terms of the human-in-the-loop models in (4.8)-(4.12) as,

$$\begin{aligned} G(\omega) = \frac{y(\omega)}{y_d(\omega)} &= \frac{(G_{H,\mathit{ff}}(\omega) + G_{H,\mathit{fb}}(\omega))P(\omega)}{1 + (G_{H,\mathit{fb}} - G_{H,in}(\omega))P(\omega)} \\ &= \frac{G_{H,1}(\omega)P(\omega)}{1 + G_{H,2}(\omega)P(\omega)} \end{aligned} \tag{4.13}$$

and, the inverse is given by,

$$G^{-1}(\omega) = \frac{1 + G_{H,2}(\omega)P(\omega)}{G_{H,1}(\omega)P(\omega)} \tag{4.14}$$

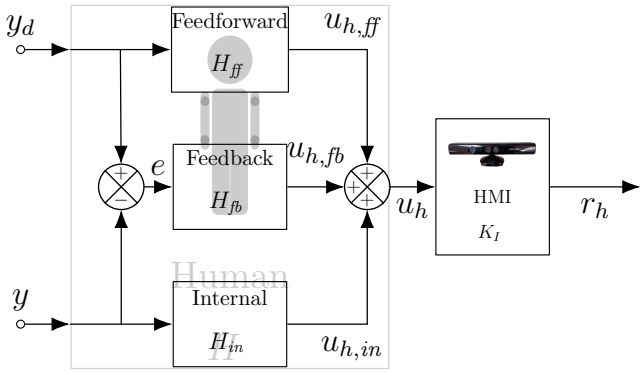


Figure 4.2: Human response model H as a combination of three channels (i) feed-forward H_{ff} with input y_d, (ii) feedback H_{fb} with input $e = y_d - y$, and (iii) internal H_{in} with input y.

or, from (4.7) as,

$$y_d(\omega) = G_{inv,1}(\omega) r_h(\omega) + G_{inv,2}(\omega) y(\omega), \tag{4.15}$$

where,

$$G_{inv,1}(\omega) = G_{H,1}^{-1}(\omega), \quad G_{inv,2}(\omega) = \frac{G_{H,2}(\omega)}{G_{H,1}(\omega)} \tag{4.16}$$

where, $G_{H,1}, G_{H,2}$ are defined in (4.8) and (4.9), respectively. Next, an estimate $\hat{y}_d$ of the intended goal y_d may be computed as,

$$\hat{y}_d(\omega) = \hat{G}^{-1}(\omega) y(\omega) = \hat{G}_{inv,1}(\omega) r_h(\omega) + \hat{G}_{inv,2}(\omega) y(\omega), \tag{4.17}$$

where, $\hat{G}^{-1}, \hat{G}_{inv,1}, \hat{G}_{inv,2}$ are known/estimated models of the frequency response functions defined in (4.14), and (4.16), respectively.

4.3.3 Research Problem

If there is no error in the model $\hat{G}$, then the inversion-based approach in (4.5) will lead to perfect estimation of the intent y_d. However, errors in the model $\hat{G}$ will lead to error in the estimated intent $\hat{y}_d$. The research problem is to identify conditions under which the inversion-based intent estimate $\hat{y}_d$ is guaranteed to yield smaller error $\hat{y}_d - y_d$ when compared

to the error $y - y_d$ from using the demonstration y instead of the intent estimate $\hat{y}_d$. Formally, let the human-in-the-loop response G at frequency ω be

$$G(\omega) = a(\omega) + jb(\omega), \tag{4.18}$$

and, the model $\hat{G}$ for each frequency ω be

$$\hat{G}(\omega) = \hat{a}(\omega) + j\hat{b}(\omega), \tag{4.19}$$

where $j = \sqrt{-1}$ and the error in the real and imaginary parts of the model $\hat{G}$,

$$\delta_a(\omega) = \hat{a}(\omega) - a(\omega), \quad \delta_b(\omega) = \hat{b}(\omega) - b(\omega), \tag{4.20}$$

are bounded by $\Delta_a(\omega)$ and $\Delta_b(\omega)$ as,

$$|\delta_a(\omega)| = |\hat{a}(\omega) - a(\omega)| \leq \Delta_a(\omega) \tag{4.21}$$

$$|\delta_b(\omega)| = |\hat{b}(\omega) - b(\omega)| \leq \Delta_b(\omega). \tag{4.22}$$

The research problem is to find conditions on the uncertainty bounds $(\Delta_a(\omega), \Delta_b(\omega))$ such that the estimated intent $\hat{y}_d(\omega)$ better approximates the intent $y_d(\omega)$ compared to the human-demonstrated output $y(\omega)$, i.e., for any nonzero intent $y_d(\omega)$,

$$\left| \frac{y_d(\omega) - \hat{y}_d(\omega)}{y_d(\omega)} \right| < \left| \frac{y_d(\omega) - y(\omega)}{y_d(\omega)} \right|, \qquad \forall \;\; y_d(\omega) \neq 0, \tag{4.23}$$

where the normalization by the intent y_d facilitates the development of conditions, for the inequality to be satisfied, that are independent of the unknown intent y_d. In particular, substituting for the terms $\hat{y}_d = \hat{G}^{-1}y = [\hat{G}^{-1}G]y_d$ and $y = Gy_d$ from (4.1) and (4.5) into the inequality (4.23) yields the condition

$$|1 - \hat{G}^{-1}(\omega)G(\omega)| < |1 - G(\omega)|, \tag{4.24}$$

which is independent of the (unknown) intent y_d.

4.4 Method

This section develops an algorithm that only uses model-inversion-based intent estimation when the model uncertainty is sufficiently small. The section begins with quantification of the acceptable uncertainty, followed by the CGPR approach to estimate the model and its uncertainty, and their use to estimate the intent.

4.4.1 Acceptable model uncertainty

Differences between the human-response dynamics G and the estimated model $\hat{G}$ leads to errors in the estimated intent $\hat{y}_d$. The following lemma identifies the acceptable error in the model, i.e., the set of acceptable (unknown) human-response dynamics G around the (known) nominal model $\hat{G}$ wherein the estimated intent $\hat{y}_d$ is at least as good as the demonstration y, i.e., satisfies the inequality (4.24).

Lemma 1 (Acceptable model error)**.** *The estimated intent $\hat{y}_d$ is at least as good as the demonstration y for any desired intent y_d, i.e., satisfies the inequality* (4.24) *at frequency ω, if and only if*

$$G \in \mathcal{M} = \begin{cases} a(\hat{a}-1) + b(\hat{b}) < 2(1-\hat{a}) \\ \qquad\qquad for\ |\hat{G}| = 1, \hat{G} \neq 1 \\ [a-(\hat{a}+\hat{c}_a)]^2 + \left[b-(\hat{b}+\hat{c}_b)\right]^2 < \frac{\hat{c}_a^2+\hat{c}_b^2}{\hat{a}^2+\hat{b}^2} \\ \qquad\qquad for\ 0 < |\hat{G}| < 1, \\ [a-(\hat{a}+\hat{c}_a)]^2 + \left[b-(\hat{b}+\hat{c}_b)\right]^2 > \frac{\hat{c}_a^2+\hat{c}_b^2}{\hat{a}^2+\hat{b}^2} \\ \qquad\qquad for\ |\hat{G}| > 1, \end{cases} \tag{4.25}$$

where,

$$\hat{c}_a(\omega) = \frac{1-\hat{a}(\omega)}{|\hat{G}^{-1}(\omega)|^2 - 1}, \quad \hat{c}_b(\omega) = \frac{-\hat{b}(\omega)}{|\hat{G}^{-1}(\omega)|^2 - 1}. \tag{4.26}$$

Proof. Using (4.20) in the inequality in (4.24), and suppressing ω for notational simplicity results in,

$$\left|1-\hat{G}^{-1}G\right| < \left|1-\hat{G}-\delta\right|, \tag{4.27}$$

where $\delta(\omega)=\delta_a(\omega)+j\delta_b(\omega)$, and using (4.18) and (4.19) results in,

$$\left|\frac{(\hat{a}-a)+j(\hat{b}-b)}{\hat{a}+j\hat{b}}\right| < \left|(1-\hat{a}-\delta_a)-j(\hat{b}+\delta_b)\right|, \tag{4.28}$$

or, equivalently by squaring both sides, and using (4.20) results in,

$$\left|\frac{\delta_a+j\delta_b}{\hat{a}+j\hat{b}}\right|^2 < \left|(1-\hat{a}-\delta_a)-j(\hat{b}+\delta_b)\right|^2, \tag{4.29}$$

or,

$$\frac{\delta_a^2+\delta_b^2}{\hat{a}^2+\hat{b}^2} < (1-\hat{a}-\delta_a)^2+(\hat{b}+\delta_b)^2, \tag{4.30}$$

and, expanding both sides and rearranging the terms results in,

$$(\delta_a^2+\delta_b^2)\left(\frac{1}{\hat{a}^2+\hat{b}^2}-1\right)+2\delta_a(1-\hat{a})-2\delta_b\hat{b} < |1-\hat{G}|^2 \tag{4.31}$$

or

$$(\delta_a^2+\delta_b^2)\left(|\hat{G}^{-1}|^2-1\right)+2\delta_a(1-\hat{a})-2\delta_b\hat{b} < |1-\hat{G}|^2. \tag{4.32}$$

Case 1: Suppose $|\hat{G}(\omega)|=1$ *and* $\hat{G}(\omega)\neq 1$

Substituting for $|\hat{G}^{-1}|=1$ in (4.32) and noting that $|\hat{G}|^2=\hat{a}^2+\hat{b}^2=1$ results in,

$$2\delta_a(1-\hat{a})-2\delta_b\hat{b} < |1-\hat{G}|^2 = 2-2\hat{a}, \tag{4.33}$$

which can be rewritten using (4.20) as

$$(\hat{a}-a)(1-\hat{a})-(\hat{b}-b)\hat{b} < 1-\hat{a}, \tag{4.34}$$

and simplified to

$$a(\hat{a}-1)+b\hat{b} < 1-\hat{a}+(\hat{a})(\hat{a}-1)+\hat{b}^2, \tag{4.35}$$

that results in the expression for this case in (4.25) of the lemma. Thus, the set $\mathcal{M}$ of dynamics G, with acceptable model error, is a half-plane as illustrated in Fig. 4.3a.

Case 2: Suppose $0 < |\hat{G}(\omega)| < 1$

Then, for each frequency ω, $|\hat{G}^{-1}(\omega)| > 1$, or equivalently, $|\hat{G}^{-1}(\omega)|^2 > 1$, i.e., (suppressing ω for notational simplicity) and, dividing (4.32) throughout by $(|\hat{G}^{-1}|^2 - 1) > 0$ and completing the squares for δ_a, δ_b results in,

$$\left(\delta_a + \frac{1-\hat{a}}{|\hat{G}^{-1}|^2 - 1}\right)^2 + \left(\delta_b - \frac{\hat{b}}{|\hat{G}^{-1}|^2 - 1}\right)^2 < \frac{|1-\hat{G}|^2|\hat{G}^{-1}|^2}{\left(|\hat{G}^{-1}|^2 - 1\right)^2} \tag{4.36}$$

which can be rewritten as

$$\left(\delta_a + \frac{1-\hat{a}}{|\hat{G}^{-1}|^2 - 1}\right)^2 + \left(\delta_b - \frac{\hat{b}}{|\hat{G}^{-1}|^2 - 1}\right)^2 < \frac{((1-\hat{a})^2 + \hat{b}^2)|\hat{G}^{-1}|^2}{\left(|\hat{G}^{-1}|^2 - 1\right)^2} \tag{4.37}$$

and using (4.20) and (4.26) leads to

$$(\hat{a} - a + \hat{c}_a)^2 + \left(\hat{b} - b + \hat{c}_b\right)^2 < \frac{\hat{c}_a^2 + \hat{c}_b^2}{\hat{a}^2 + \hat{b}^2} \tag{4.38}$$

resulting in the expression for this case in (4.25) of the lemma. Thus, the set $\mathcal{M}$ of dynamics G, with acceptable model error, lies inside a disc in the complex plane with the center at $(\hat{c}_a + \hat{a}, \hat{c}_b + \hat{b})$ and radius $\rho = \sqrt{\frac{\hat{c}_a^2 + \hat{c}_b^2}{\hat{a}^2 + \hat{b}^2}}$, as shown in Fig. 4.3b.

Case 3: Suppose $|\hat{G}(\omega)| > 1$

Then, $|\hat{G}^{-1}| < 1$, or equivalently, $1 - |\hat{G}^{-1}|^2 > 0$, and so (4.32) may be rewritten as,

$$-(\delta_a^2 + \delta_b^2)(1 - |\hat{G}^{-1}|^2) + 2\delta_a(1-\hat{a}) - 2\delta_b\hat{b} - |1-\hat{G}|^2 < 0 \tag{4.39}$$

and, multiplying throughout by -1 results in,

$$(\delta_a^2 + \delta_b^2)(1 - |\hat{G}^{-1}|^2) - 2\delta_a(1-\hat{a}) + 2\delta_b\hat{b} + |1-\hat{G}|^2 > 0 \tag{4.40}$$

and rearranging the terms by completion of squares on the left-hand side we have,

$$\left(\delta_a - \frac{1-\hat{a}}{1-|\hat{G}^{-1}|^2}\right)^2 + \left(\delta_b + \frac{\hat{b}}{1-|\hat{G}^{-1}|^2}\right)^2 - \frac{|1-\hat{G}|^2|\hat{G}^{-1}|^2}{\left(1-|\hat{G}^{-1}|^2\right)^2} > 0, \tag{4.41}$$

and using (4.20) and (4.26) leads to

$$(\hat{a}-a+\hat{c}_a)^2 + \left(\hat{b}-b+\hat{c}_b\right)^2 > \frac{\hat{c}_a^2+\hat{c}_b^2}{\hat{a}^2+\hat{b}^2} \tag{4.42}$$

resulting in the expression for this case in (4.25) of the lemma. Thus, the set $\mathcal{M}$ of dynamics G, with acceptable model error, lies outside an open disc in the complex plane with the center at $(\hat{c}_a+\hat{a}, \hat{c}_b+\hat{b})$ and radius $\rho = \sqrt{\frac{\hat{c}_a^2+\hat{c}_b^2}{\hat{a}^2+\hat{b}^2}}$, as shown in Figs. 4.3c-4.3e. □

Remark 4 (Robustness of the intent estimate)**.** *The robustness of the intent estimation approach (i.e., the allowable size of acceptable error δ_a, δ_b in the model to ensure improvement with the approach) decreases as the size of the model $\hat{G}$ decreases. In particular when $\hat{a}$ and $\hat{b}$ are small, the radius*

$$\rho = \sqrt{\frac{\hat{c}_a^2+\hat{c}_b^2}{\hat{a}^2+\hat{b}^2}} = \frac{1}{|\hat{G}^{-1}|-\hat{G}}\sqrt{(1-\hat{a})^2+\hat{b}^2} \tag{4.43}$$

of the acceptance region tends to zero since the numerator of the radius ρ expression in (4.43) *becomes one and its denominator becomes large. Moreover, when the model estimate is zero, i.e., $\hat{G}(\omega)=0$, then the model has a zero at ω and the response $y(\omega)$ predicted by the model is zero even when the intent $y_d(\omega)$ is non-zero. In this case the model-inversion approach cannot be used to estimate the intent.*

Remark 5 (Expert human demonstration)**.** *Let the model be one, i.e., $\hat{G}(\omega)=1$. Then, when there is no modeling error, the human-in-the-loop response is one, i.e., $G=1$. Therefore, the human is an expert at tracking at frequency ω and tracks the intended trajectory well. Therefore, the estimated intent $\hat{y}_d$ cannot be guaranteed to be more precise than the human demonstrated output y and the set $\mathcal{M}$ of acceptable dynamics G is the empty set.*

The set $\mathcal{M}$ of dynamics G defined in (4.25) yields a geometric interpretation of the regions where the use of the model $\hat{G}$ improves the intent-estimation. Nevertheless, the actual error (δ_a, δ_b) in the model is unknown. Therefore, the following lemma develops necessary and sufficient conditions on uncertainty bounds (Δ_a, Δ_b) (defined in (4.21) and (4.22)) on the modeling error (δ_a, δ_b) to ensure improvement with the use of the model-based intent estimation.

Lemma 2. *For a given model $\hat{G}$ of the unknown human-in-the-loop dynamics G, at each frequency ω, the condition for intent estimation improvement in* (4.23) *is satisfied if and only if the uncertainty bounds $\Delta_a(\omega), \Delta_b(\omega)$ defined in* (4.21) *and* (4.22) *satisfy the following conditions at frequency ω (dependence on ω suppressed for notational simplicity).*

1. *Case 1: If $|\hat{G}| = 1, \hat{G} \neq 1$*

$$\Delta_a|1 - \hat{a}| + \Delta_b|\hat{b}| < 1 - \hat{a}, \tag{4.44}$$

2. *Case 2: If $0 < |\hat{G}(\omega)| < 1$*

$$(\Delta_a + |\hat{c}_a|)^2 + (\Delta_b + |\hat{c}_b|)^2 < \frac{\hat{c}_a^2 + \hat{c}_b^2}{\hat{a}^2 + \hat{b}^2}, \tag{4.45}$$

3. *Case 3: If $|\hat{G}(\omega)| > 1$ and,*

   - *Case 3a: if $\Delta_a(\omega) \leq |\hat{c}_a(\omega)|$ and $\Delta_b(\omega) \leq |\hat{c}_b(\omega)|$*

$$(\Delta_a - |\hat{c}_a|)^2 + (\Delta_b - |\hat{c}_b|)^2 > \frac{\hat{c}_a^2 + \hat{c}_b^2}{\hat{a}^2 + \hat{b}^2}, \tag{4.46}$$

   - *Case 3b: if $\Delta_a(\omega) > |\hat{c}_a(\omega)|$ and $\Delta_b(\omega) \leq |\hat{c}_b(\omega)|$*

$$\Delta_b < |\hat{c}_b| - \sqrt{\frac{\hat{c}_a^2 + \hat{c}_b^2}{\hat{a}^2 + \hat{b}^2}}, \tag{4.47}$$

- *Case 3c: if* $\Delta_a(\omega) \leq |\hat{c}_a(\omega)|$ *and* $\Delta_b(\omega) > |\hat{c}_b(\omega)|$

$$\Delta_a < |\hat{c}_a| - \sqrt{\frac{\hat{c}_a^2 + \hat{c}_b^2}{\hat{a}^2 + \hat{b}^2}}, \tag{4.48}$$

where $\hat{c}_a, \hat{c}_b$ *are defined in* (4.26).

Proof. Case 1: Suppose $|\hat{G}(\omega)| = 1, \hat{G}(\omega) \neq 1$

Then, according to Lemma 1, the acceptable dynamics $G(\omega) = a(\omega) + jb(\omega)$ satisfies the relevant condition in (4.25) for each frequency ω, which is equivalent to, from (4.33)

$$\delta_a(\omega)(1 - \hat{a}(\omega)) - \delta_b(\omega)\hat{b}(\omega) < 1 - \hat{a}. \tag{4.49}$$

Since $\delta_a(\omega) \in [-\Delta_a(\omega), \Delta_a(\omega)]$ and $\delta_b(\omega) \in [-\Delta_b(\omega), \Delta_b(\omega)]$, which implies the maximal value of the LHS of (4.49) is obtained when

$$\delta_a(\omega) = \Delta_a(\omega)\frac{1 - \hat{a}(\omega)}{|1 - \hat{a}(\omega)|}, \quad \delta_b(\omega) = -\Delta_b(\omega)\frac{\hat{b}(\omega)}{|\hat{b}(\omega)|}, \tag{4.50}$$

which when substituted into (4.49) results in the required condition in (4.44).

Case 2: Suppose $0 < |\hat{G}(\omega)| < 1$

Then, according to Lemma 1, the acceptable dynamics $G(\omega) = a(\omega) + jb(\omega)$ satisfies the relevant condition in (4.25) for each frequency ω, which is equivalent to, from (4.37)

$$(\delta_a(\omega) + \hat{c}_a(\omega))^2 + (\delta_b(\omega) + \hat{c}_b(\omega))^2 < \frac{\hat{c}_a^2 + \hat{c}_b^2}{\hat{a}^2 + \hat{b}^2} \tag{4.51}$$

where $\hat{c}_a, \hat{c}_b$ are defined in (4.26), respectively, and since $\delta_a(\omega) \in [-\Delta_a(\omega), \Delta_a(\omega)]$ and $\delta_b(\omega) \in [-\Delta_b(\omega), \Delta_b(\omega)]$, the inequality in (4.51) must be satisfied for the maximum value of the left hand side, which is obtained when $(\delta_a(\omega), \delta_b(\omega))$ are chosen as,

$$\delta_a(\omega) = \Delta_a(\omega)\frac{\hat{c}_a(\omega)}{|\hat{c}_a(\omega)|}, \quad \delta_b(\omega) = \Delta_b(\omega)\frac{\hat{c}_b(\omega)}{|\hat{c}_b(\omega)|}, \tag{4.52}$$

which when substituted in (4.51) results in the required condition in (4.45).

Case 3: Suppose $|\hat{G}(\omega)| > 1$

Lemma 1 states that the dynamics $G(\omega) = a(\omega) + jb(\omega)$ satisfies the relevant condition in (4.25) for each frequency ω, which is equivalent to, from (4.41)

$$(\delta_a(\omega) + \hat{c}_a(\omega))^2 + (\delta_b(\omega) + \hat{c}_b(\omega))^2 \geq \frac{\hat{c}_a^2 + \hat{c}_b^2}{\hat{a}^2 + \hat{b}^2}, \tag{4.53}$$

where $\hat{c}_a, \hat{c}_b$ are defined in (4.26), respectively, and since $\delta_a(\omega) \in [-\Delta_a(\omega), \Delta_a(\omega)]$ and $\delta_b(\omega) \in [-\Delta_b(\omega), \Delta_b(\omega)]$ for each frequency ω, the minimum of the left hand side of the inequality in (4.53) may be found considering the relative size of $\Delta_a(\omega)$ and $\Delta_b(\omega)$ as follows:

(a) If $\Delta_a(\omega) \leq |\hat{c}_a(\omega)|$ and $\Delta_b(\omega) \leq |\hat{c}_b(\omega)|$: In this case, the minimum of the LHS of (4.53) is obtained for (δ_a, δ_b) chosen as,

$$\delta_a(\omega) = -\Delta_a(\omega)\frac{\hat{c}_a(\omega)}{|\hat{c}_a(\omega)|}, \quad \delta_b(\omega) = -\Delta_b(\omega)\frac{\hat{c}_b(\omega)}{|\hat{c}_b(\omega)|}, \tag{4.54}$$

which when substituted into (4.53) results in the required condition in (4.46).

(b) If $\Delta_a(\omega) > |\hat{c}_a(\omega)|$ $\Delta_b(\omega) \leq |\hat{c}_b(\omega)|$: In this case, the minimum of the LHS of (4.53) is obtained for (δ_a, δ_b) chosen as,

$$\delta_a(\omega) = -\hat{c}_a(\omega), \quad \delta_b(\omega) = -\Delta_b(\omega)\frac{\hat{c}_b(\omega)}{|\hat{c}_b(\omega)|}, \tag{4.55}$$

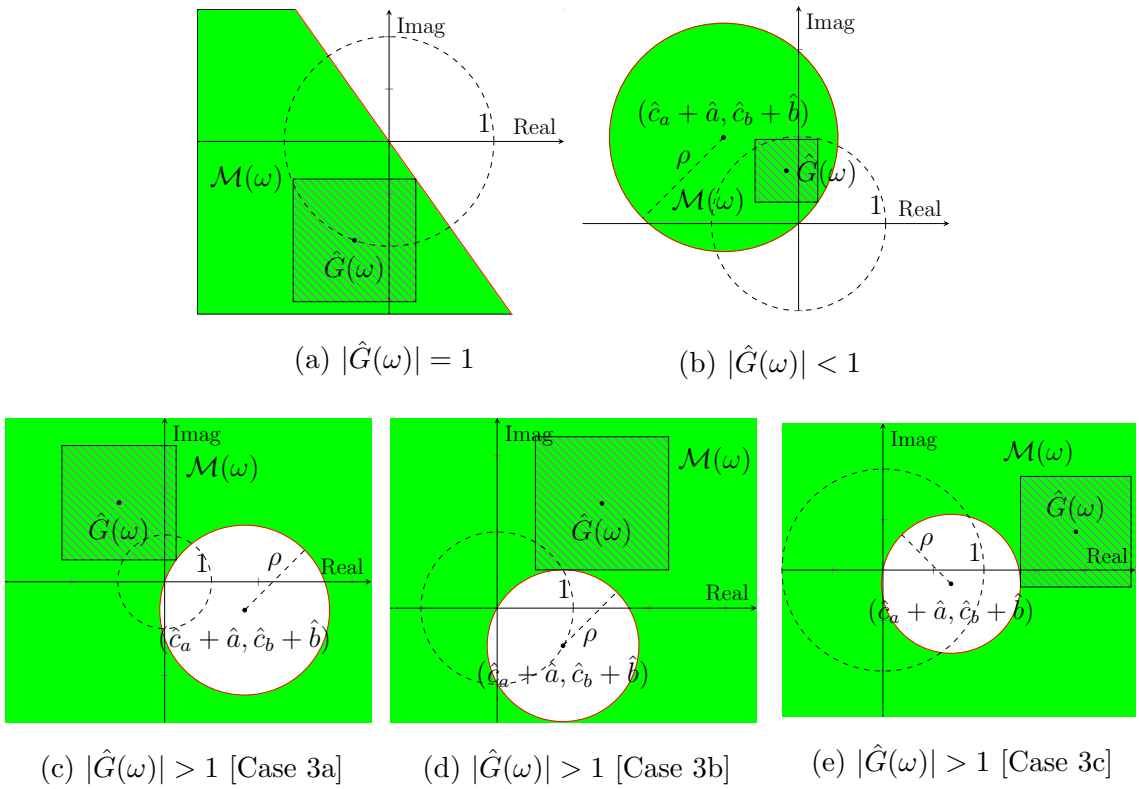
which when substituted into (4.53) results in the required condition in (4.47).

(c) If $\Delta_a(\omega) \leq |\hat{c}_a(\omega)|$ and $\Delta_b(\omega) > |\hat{c}_b(\omega)|$: In this case, the minimum of the LHS of (4.53) is obtained for (δ_a, δ_b) chosen as,

$$\delta_a(\omega) = -\Delta_a(\omega)\frac{\hat{c}_a(\omega)}{|\hat{c}_a(\omega)|}, \quad \delta_b(\omega) = -\hat{c}_b(\omega), \tag{4.56}$$

which when substituted into (4.53) results in the required condition in (4.48).

□



(a) $|\hat{G}(\omega)| = 1$ (b) $|\hat{G}(\omega)| < 1$

(c) $|\hat{G}(\omega)| > 1$ [Case 3a] (d) $|\hat{G}(\omega)| > 1$ [Case 3b] (e) $|\hat{G}(\omega)| > 1$ [Case 3c]

Figure 4.3: Graphical representation of the acceptable region $\mathcal{M}(\omega) \subset \mathcal{M}(\omega)$ when $\Delta_a(\omega) = \Delta_b(\omega) = \Delta(\omega)$ according to Remark 8 (shown as the hatched square in the green shaded region) for (a) Case 1: $|\hat{G}(\omega)| = 1, \hat{G}(\omega) \neq 1$, (b) Case 2: $0 < |\hat{G}(\omega)| < 1$, (c) Case 3a: $|\hat{G}(\omega)| > 1$ with $\Delta(\omega) \leq |\hat{c}_a(\omega)|$ and $\Delta(\omega) \leq |\hat{c}_b(\omega)|$, (d) Case 3b: $|\hat{G}(\omega)| > 1$ with $\Delta(\omega) > |\hat{c}_a(\omega)|$, and (e) Case 3c: $|\hat{G}(\omega)| > 1$ with $\Delta(\omega) > |\hat{c}_b(\omega)|$, for each frequency ω. The red circle in each case denotes the decision boundary as specified in the inequalities in the definition of the set $\mathcal{M}$ in (4.25).

Corollary 1. *If there exists* $\overline{\Delta}(\omega) \geq 0$ *at each frequency* ω *such that*

$$\Delta_a(\omega) < \overline{\Delta}(\omega), \quad \Delta_b(\omega) < \overline{\Delta}(\omega), \tag{4.57}$$

where Δ_a, Δ_b *are defined in* (4.21) *and* (4.22)*, respectively, then the intent estimation improvement in* (4.23) *is satisfied if and only if the maximum bound* $\overline{\Delta}(\omega)$ *for each frequency* ω *(dependence on* ω *suppressed for notational simplicity) is given by,*

1. *Case 1: If* $|\hat{G}| = 1, \hat{G} \neq 1$

$$\overline{\Delta} = \frac{1-\hat{a}}{|1-\hat{a}| + |\hat{b}|}, \tag{4.58}$$

2. *Case 2: If* $0 < |\hat{G}(\omega)| < 1$

$$\overline{\Delta} = \frac{1}{2}\left(\sqrt{\begin{array}{l}2|\hat{c}_a||\hat{c}_b| \\ +(|\hat{c}_a|^2+|\hat{c}_b|^2)\left(2|\hat{G}^{-1}|^2-1\right)\end{array}} - (|\hat{c}_a|+|\hat{c}_b|)\right) \tag{4.59}$$

3. *Case 3: If* $|\hat{G}(\omega)| > 1$ *and,*

   - *Case 3a: if* $\overline{\Delta}(\omega) \leq |\hat{c}_a(\omega)|$ *and* $\overline{\Delta}(\omega) \leq |\hat{c}_b(\omega)|$

$$0 \leq \overline{\Delta} = \frac{1}{2}\left(|\hat{c}_a| + |\hat{c}_b| - \sqrt{\begin{array}{l}2|\hat{c}_a||\hat{c}_b| \\ +(|\hat{c}_a|^2+|\hat{c}_b|^2)\left(2|\hat{G}^{-1}|^2-1\right)\end{array}}\right) \tag{4.60}$$

   - *Case 3b: if* $|\hat{c}_a(\omega)| < \overline{\Delta}(\omega) \leq |\hat{c}_b(\omega)|$

$$\overline{\Delta} = |\hat{c}_b| - \sqrt{\frac{\hat{c}_a^2+\hat{c}_b^2}{\hat{a}^2+\hat{b}^2}}, \tag{4.61}$$

   - *Case 3c: if* $|\hat{c}_b(\omega)| < \overline{\Delta}(\omega) \leq |\hat{c}_a(\omega)|$

$$\overline{\Delta} = |\hat{c}_a| - \sqrt{\frac{\hat{c}_a^2+\hat{c}_b^2}{\hat{a}^2+\hat{b}^2}}, \tag{4.62}$$

where $\hat{c}_a, \hat{c}_b$ *are defined in* (4.26).

Proof. The conditions are obtained by setting the model error bounds $\Delta_a = \Delta_b = \overline{\Delta}$ (maximum upper bound) for the corresponding conditions in Lemma 2 for each case. □

Remark 6 (Maximum Allowable Uncertainty)**.** *The Corollary specifies the conditions on the maximum allowable uncertainty* $\overline{\Delta}$ *such that the intent estimation improvement in* (4.23) *is satisfied for a given model* $\hat{G}$ *at each frequency* ω*. This is equivalent to finding the largest square centered on the model* $\hat{G}$ *in the complex plane that is completely contained inside the acceptable set* $\mathcal{M}$ *of dynamics* G*, as shown in Fig. 4.3 for each case as a hatched square whose side-length is given by* $\overline{\Delta}$ *computed according to the conditions in the Corollary.*

4.4.2 Inverse-model form of the Robustness conditions

Consider the inverse dynamics G^{-1}

$$G^{-1} = \tilde{a} + j\tilde{b} \tag{4.63}$$

and the corresponding model $\hat{G}^{-1}$, specified as,

$$\hat{G}^{-1} = \hat{\tilde{a}} + j\hat{\tilde{b}} \tag{4.64}$$

with associated model uncertainties $\tilde{\Delta}_a, \tilde{\Delta}_b$ defined for each frequency ω as,

$$|\tilde{\delta}_a(\omega)| = |\hat{\tilde{a}}(\omega) - \tilde{a}(\omega)| \leq \tilde{\Delta}_a, \tag{4.65}$$

$$|\tilde{\delta}_b(\omega)| = |\hat{\tilde{b}}(\omega) - \tilde{b}(\omega)| \leq \tilde{\Delta}_b, \tag{4.66}$$

Then, the intent estimation improvement in (4.24) may be rewritten in terms of the inverse dynamics G^{-1} by dividing throughout by $G \neq 0$,

$$\left|G^{-1} - \hat{G}^{-1}\right| \leq \left|G^{-1} - 1\right|, \tag{4.67}$$

and, the conditions in Lemmas 1-2 may be rewritten in terms of the inverse model $\hat{G}^{-1}$ defined in (4.64) as follows.

Lemma 3 (Acceptable model error)**.** *The estimated intent $\hat{y}_d$ is at least as good as the demonstration y for any desired intent y_d, i.e., satisfies the inequality* (4.67) *at frequency ω, if and only if the dynamics G^{-1} lies in the set $\tilde{\mathcal{M}}$ in the complex plane $\mathbb{C}$ defined by*

$$G^{-1} \in \tilde{\mathcal{M}} = \left\{ (\tilde{a} + j\tilde{b} \in \mathbb{C}) \ni \tilde{a}(\hat{\tilde{a}} - 1) + \tilde{b}\hat{\tilde{b}} > \frac{\hat{\tilde{a}}^2 + \hat{\tilde{b}}^2 - 1}{2} \right\}. \tag{4.68}$$

Additionally, the set $\hat{\tilde{\mathcal{M}}}$ of the dynamics estimated by the model $\hat{G}^{-1}$, given by,

$$G^{-1} \in \hat{\tilde{\mathcal{M}}} = \left\{ (\hat{G}^{-1} - \delta \in \mathbb{C}) \ni |\tilde{\delta}_a| \leq \tilde{\Delta}_a, |\tilde{\delta}_b| \leq \tilde{\Delta}_b \right\} \tag{4.69}$$

is completely contained in the set $\tilde{\mathcal{M}}$, i.e., $\hat{\tilde{\mathcal{M}}} \subset \tilde{\mathcal{M}}$ if and only if the bounds on the model uncertainties $\tilde{\Delta}_a, \tilde{\Delta}_b$ satisfy

$$\tilde{\Delta}_a |\hat{\tilde{a}} - 1| + \tilde{\Delta}_b |\hat{\tilde{b}}| < \frac{(\hat{\tilde{a}} - 1)^2 + \hat{\tilde{b}}^2}{2}, \tag{4.70}$$

and, the supremum $\overline{\tilde{\Delta}}$ on the model uncertainties is given by,

$$\tilde{\delta}_a, \tilde{\delta}_b < \overline{\tilde{\Delta}} = \frac{(\hat{\tilde{a}} - 1)^2 + \hat{\tilde{b}}^2}{2(|\hat{\tilde{a}} - 1| + |\hat{\tilde{b}}|)}. \tag{4.71}$$

Proof. Using (4.18) and (4.19) in the inequality in (4.67), and suppressing ω for notational simplicity results in,

$$\left| (\tilde{a} - \hat{\tilde{a}}) + j(\tilde{b} - \hat{\tilde{b}}) \right| < \left| (\tilde{a} - 1) + j\tilde{b} \right|, \tag{4.72}$$

or, equivalently by squaring both sides and expanding the squared magnitudes results in,

$$(\tilde{a} - \hat{\tilde{a}})^2 + (\tilde{b} - \hat{\tilde{b}})^2 < (\tilde{a} - 1)^2 + \tilde{b}^2, \tag{4.73}$$

which results in the required inequality in (4.68) upon simplification. Next, using (4.20) in (4.73) results in,

$$\tilde{\delta}_a^2 + \tilde{\delta}_b^2 < (\hat{\tilde{a}} - \tilde{\delta}_a - 1)^2 + (\hat{\tilde{b}} - \tilde{\delta}_b)^2, \tag{4.74}$$

which upon simplification results in,

$$\tilde{\delta}_a(\hat{\tilde{a}}-1)+\tilde{\delta}_b\hat{\tilde{b}}<\frac{(\hat{\tilde{a}}-1)^2+\hat{\tilde{b}}^2}{2}. \tag{4.75}$$

From (4.21), we have $\tilde{\delta}_a(\omega)\in[-\tilde{\Delta}_a(\omega),\tilde{\Delta}_a(\omega)]$ and from (4.22) we have $\tilde{\delta}_b(\omega)\in[-\tilde{\Delta}_b(\omega),\tilde{\Delta}_b(\omega)]$, then the maximal value of the LHS of (4.75) is obtained when

$$\tilde{\delta}_a(\omega)=\tilde{\Delta}_a(\omega)\frac{\hat{\tilde{a}}(\omega)-1}{|\hat{\tilde{a}}(\omega)-1|},\quad \tilde{\delta}_b(\omega)=\tilde{\Delta}_b(\omega)\frac{\hat{\tilde{b}}(\omega)}{|\hat{\tilde{b}}(\omega)|}, \tag{4.76}$$

which when substituted into (4.74) results in the required condition in (4.70). Finally, the supremeum $\overline{\tilde{\Delta}}$ in (4.71) is obtained by setting the model error bounds $\tilde{\Delta}_a=\tilde{\Delta}_b=\overline{\tilde{\Delta}}$ (maximum upper bound) and replacing the inequality with equality in the corresponding condition in (4.70).

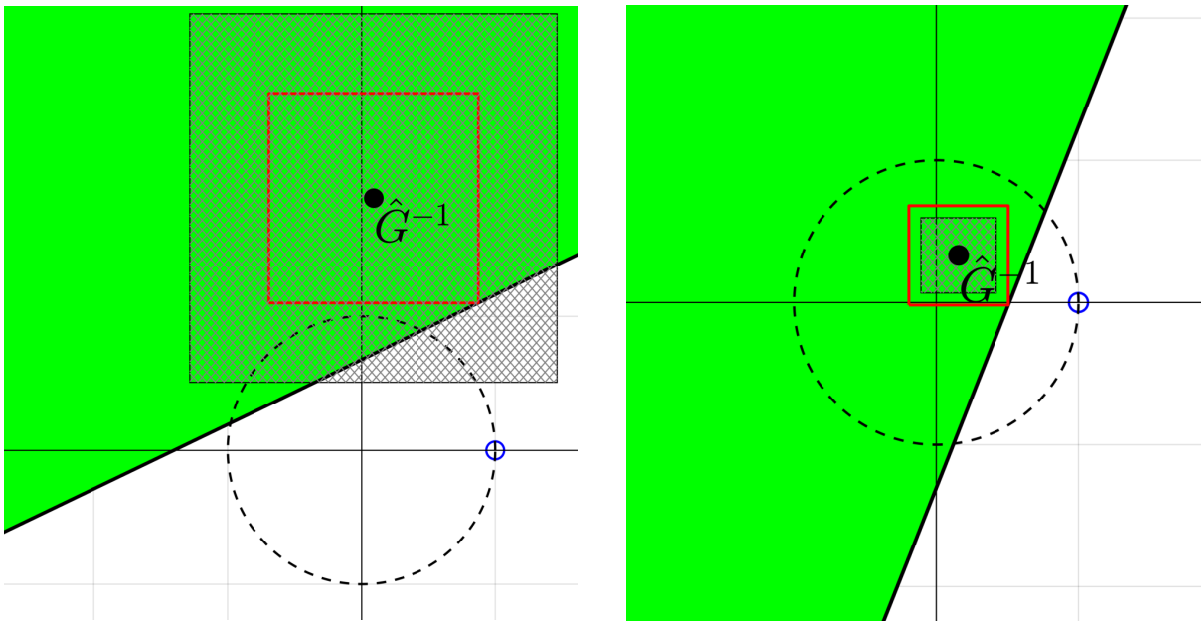
This completes the proof. □

This form of the robustness conditions is useful if the inverse model $(\hat{G}^{-1},\tilde{\Delta}_a,\tilde{\Delta}_b)$ are estimated directly from training data.

Remark 7 (Maximum Allowable Uncertainty - Inverse form)**.** *Lemma 3 specifies the conditions on the maximum allowable uncertainty* $\overline{\Delta}$ *such that the intent estimation improvement in* (4.23) *is satisfied for a given model* $\hat{G}^{-1}$ *at each frequency* ω*. This is equivalent to finding the largest square centered on the model* $\hat{G}^{-1}$ *in the complex plane that is completely contained inside the closure of the acceptable set* $\tilde{\mathcal{M}}$ *of dynamics* G^{-1}*, as shown in Fig. 4.3 as a hatched square, with each side of length* $2\overline{\Delta}$ *where* $\overline{\Delta}$ *is computed according to* (4.71)*.*

4.4.3 Robust Intent Estimation using Blending

Given a model $\hat{G}(\omega)$ with model uncertainties $\Delta_a,\Delta_b(\omega)$ at each frequency ω, the results from the previous section specify whether the intent estimation improvement in (4.23) is guaranteed using the inversion-based intent estimate $\hat{y}_d$ computed according to (4.4). The



(a) $\Delta(\omega) > \overline{\Delta}(\omega)$ (b) $\Delta(\omega) < \overline{\Delta}(\omega)$

Figure 4.4: Graphical representation of the estimated set of dynamics $\hat{\tilde{\mathcal{M}}}(\omega)$ (shown as the hatched square) for a given model $\hat{G}^{-1}$ with model uncertainties $\Delta_a = \Delta_b = \Delta$ and the acceptable set of dynamics $\tilde{\mathcal{M}}(\omega)$ (shown as the green shaded region) when (a) $\Delta(\omega) > \overline{\Delta}(\omega)$, i.e., Lemma 3 conditions are not satisfied, and (b) $\Delta(\omega) < \overline{\Delta}(\omega)$, i.e., Lemma 3 conditions are satified for the given model $\hat{G}^{-1}$, where $\overline{\Delta}(\omega)$ is the maximum allowable uncertainty defined in (4.71) that is half of the length of the square depicted with the red boundary in each case.

simplest approach to estimating the intent would be to reject the model-inversion at frequencies where the modeling uncertainties are high such that the Lemma conditions are violated, and use the demonstrations y at those frequencies, i.e.,

$$\hat{y}_d(\omega) = \begin{cases} \hat{G}^{-1}(\omega)y(\omega), & \text{for } \omega \in \Omega = \left\{\omega \ni (\hat{G}^{-1}(\omega), \Delta_a(\omega), \Delta_b(\omega)) \text{ satisfy Lemma 2}\right\}. \\ y(\omega), & \text{otherwise.} \end{cases} \tag{4.77}$$

But, practically, modeling uncertainty Δ_a, Δ_b is usually obtained as an estimate with a certain level of statistical confidence during model estimation, and so deciding whether to use the inverse-model for intent estimation directly by checking the Lemma conditions in the previous section might lead to false negatives, resulting in rejecting models that are actually valid due to over-estimating its uncertainty.

Instead, a "softmax" approach [57] may be used to blend the two decisions, i.e., (1) to use model-inversion or, (2) use the human demonstrations to estimate the intent, that incorporates the confidence in the estimated model. For example, the intent estimate $\hat{y}_d$ may be computed at each frequency ω as,

$$\hat{y}_d(\omega) = \sigma(\omega)[\hat{G}^{-1}(\omega)y(\omega)] + (1 - \sigma(\omega))[y(\omega)]. \tag{4.78}$$

where $\sigma(\omega)$ is a sigmoidal activation function, defined for each frequency ω as,

$$\sigma(\omega) = \frac{2\exp\left(-\beta\Delta(\omega)/\overline{\Delta}(\omega)\right)}{1 + \exp\left(-\beta\Delta(\omega)/\overline{\Delta}(\omega)\right)}, \tag{4.79}$$

where $\Delta(\omega) = \max\{\Delta_a(\omega), \Delta_b(\omega)\} > 0$ is the estimated model uncertainty, $\overline{\Delta}(\omega) > 0$ is the maximum allowable uncertainty as specified in (4.71), and $\beta > 0$ is a parameter that determines the shape of the modified sigmoidal activation function σ as shown in Fig. 4.5. Note that $\sigma(\omega) \in (0, 1)$ for all frequencies ω, specifically, for $\sigma \to 0$, which happens when $\Delta \gg \overline{\Delta}$ in (4.79), the demonstration y specifies the intent estimate in (4.78) which is desired since the model $\hat{G}$ is not valid according the results from Lemma 1. And, for $\sigma \to 1$,

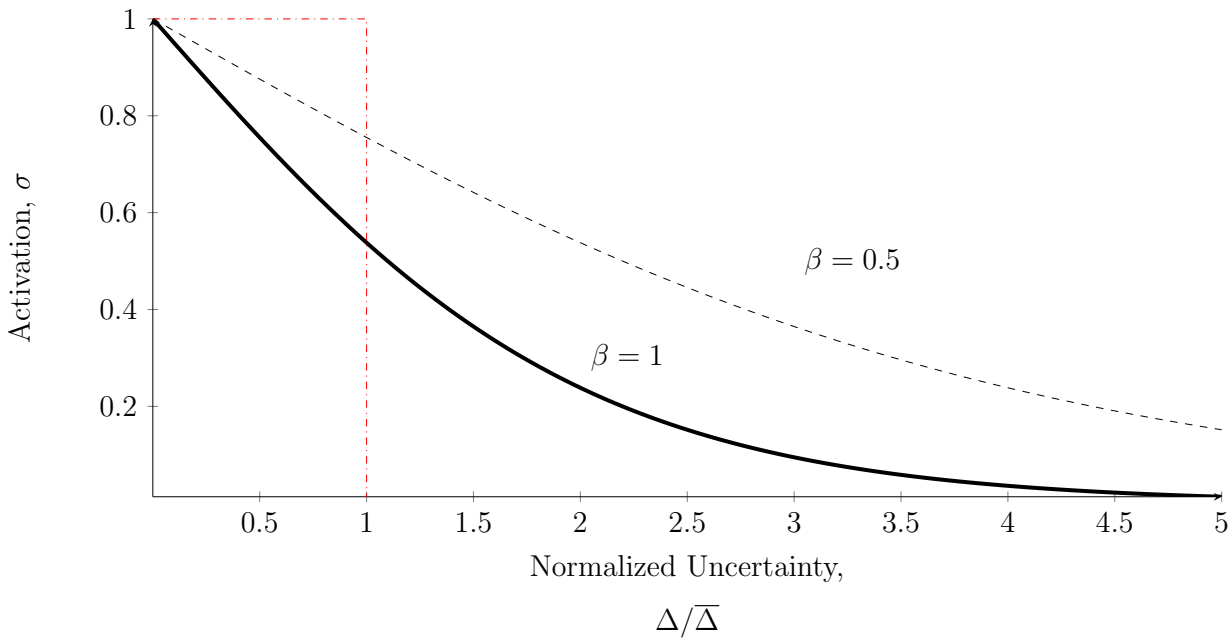


Figure 4.5: Activation function σ corresponding to model uncertainties $\Delta/\overline{\Delta}$ defined in (4.79), used in blending the intent estimates obtained by model-inversion (dominant when $\sigma \to 1$) and the human demonstration (dominant when $\sigma \to 0$).

which happens when $\Delta \ll \overline{\Delta}$ in (4.79), the model-inversion-based intent estimate specifies the intent in (4.78), which is desired since the model is good in this scenario. Further, the parameters (β_1, β_2) must be trained separately to optimize the shape of the activation function σ, as shown in Fig. 4.5, similar to training Softmax classifiers [57] which is outside the scope of this article. Instead, the parameter β is chosen as 1 in order to test the benefit of using this blending approach to estimating the intent under modeling uncertainty.

4.4.4 Complex-valued Gaussian Process Regression (CGPR)

A modified Gaussian Process Regression (GPR) method called Complex-valued Gaussian Process Regression (CGPR), first introduced in [9], is used to obtain human response models (in the frequency domain) from training data. These models are also checked using Least Squares Regression (point-wise for each frequency), with the CGPR allowing for smooth interpolation between the data points obtained during training.

SISO Model Estimation

Consider first the single-input-single-output (SISO) system G, whose transfer function may be estimated as a Gaussian process f,

$$G(\omega) = \frac{Y(\omega)}{U(\omega)} = f(\omega) + \epsilon, \tag{4.80}$$

where, the noise term ϵ is assumed to be zero-mean Gaussian random variable with variance σ_e^2, i.e., $\epsilon \sim \mathcal{N}(0, \sigma_e^2)$. In the Gaussian process framework, it is assumed that the function values $f(\omega_1)$ and $f(\omega_2)$ are correlated according to a kernel function $\hat{k}(\omega_1, \omega_2)$ which quantifies the similarity between the inputs ω_1 and ω_2, specifically the squared-exponential kernel given by,

$$\hat{k}(\omega_1, \omega_2) = \sigma_f \exp\left(-\frac{1}{2\gamma^2}(\omega_1 - \omega_2)^H(\omega_1 - \omega_2)\right), \tag{4.81}$$

where $(\cdot)^H$ implies the conjugate transpose operation, and σ_f and γ are the hyper-parameters that are not known and must be fit to training data. Then, given measured data $\hat{G}_T \in \mathbb{C}^n$

at frequencies $\Omega_T \in \mathbb{R}^n$, the resulting estimate of the underlying function f at a frequency ω is given by [48],

$$E\left[f(\omega)\right] = K(\omega, \Omega_T) K_T^{-1} \hat{G}_T(\Omega_T), \tag{4.82}$$

$$\mathrm{Cov}\left[f(\omega)\right] = K(\omega, \omega) - K(\omega, \Omega_T) K_T^{-1} K(\Omega_T, \omega). \tag{4.83}$$

where, $K(\Omega_1, \Omega_2)$ is the co-variance matrix formed by computing the kernel function $\hat{k}(\omega_i, \omega_j)$ for all pairs of $\omega_i \in \Omega_1$ and $\omega_j \in \Omega_2$ and $K_T = (K(\Omega_T, \Omega_T) + \sigma_e^2 I)$ is the co-variance matrix of the noisy measurements.

The hyper-parameters $\theta = \{\gamma, \sigma_f, \sigma_e\}$ may be learned from the measured training data $\hat{G}_T$, for example using marginal likelihood estimation [48],

$$\log\ p(G \,|\, \omega, \theta) = -\frac{1}{2}\left(\hat{G}_T^T K_T \hat{G}_T + \log|K_T| + n \log 2\pi\right), \tag{4.84}$$

where n is the number of samples at each frequency ω.

Remark 8 (Estimating model uncertainty)**.** *The variance in estimating the model, given in* (4.83) *provides a measure of the uncertainty in the estimated model. Thus, the bounds* Δ_a *and* Δ_b *on the real and imaginary parts of the estimated model* $\hat{G}$ *may be computed as, for example, twice the standard deviation estimated by the CGPR method. Further, since the squared exponential kernel in* (4.81) *considers the real and imaginary parts as independent [48], the bounds may be chosen as,*

$$\Delta_a(\omega) = \Delta_b(\omega) = \alpha\sqrt{\mathrm{Cov}[f(\omega)]}, \tag{4.85}$$

where α *represents the level of confidence in the uncertainty estimates (e.g.,* $\alpha = 1.96$ *for 95% confidence interval).*

Multiple-Input Single-Output (MISO) Model Estimation

From (4.5) it can be seen that if the training data also includes the human input r_h, then the intent estimate may be obtained through a MISO model $(\hat{G}_{inv,1}, \hat{G}_{inv,2})$ which may also be

Algorithm 1: Non-parametric Intent Estimation (SISO)

Input : Training data $(y_{train}(t), y_{d,train}(t))$, Human-demonstrated output $y_{demo}(t)$

Output : Intent estimate $\hat{y}_d(t)$

Parameter: Sampling frequency f_s Hz, Cutoff frequency f_c Hz, Confidence interval (e.g., 95%) for model uncertainty α

Training:

$Y_{train} = \mathcal{F}(y_{train})$, $Y_{d,train} = \mathcal{F}(y_{d,train})$

Estimate $\gamma, \sigma_f, \sigma_e$ based on $\hat{G}_T$ (4.84)

$\hat{G}(\omega) = E[f(\omega)]$ (4.82)

$\Delta(\omega) = \alpha\sqrt{\mathrm{Cov}[f(\omega)]}$ (4.83)

Compute $\Delta_a(\omega) = \Delta_b(\omega) = \Delta(\omega)$ using (4.91)

Compute maximum upper bound $\overline{\Delta}(\omega)$ using Corollary 1

Compute activation index $\sigma(\omega)$ according to (4.79).

Human demonstration:

$Y = \mathcal{F}(y_{demo})$

Inversion-based Intent Estimate:

$\tilde{Y}_d(\omega) = \hat{G}^{-1}(\omega)Y(\omega)$ (4.5)

Blend intent estimate with demonstration:

$\hat{Y}_d(\omega) = \sigma(\omega)\tilde{Y}_d(\omega) + (1 - \sigma(\omega))Y(\omega)$

return $\hat{y}_d = \mathcal{F}^{-1}(\hat{Y}_d)$

estimated using the CGPR framework described in the previous section. The main benefit of this approach is to decouple the model estimation of the closed-loop inverse dynamics into two channels, specifically, one that is influenced by the human action r_h, and the other that depends on the controlled system P. This enables us to potentially apply the learned inverse models to new controlled systems P by simply relearning the model channel associated with the plant P.

Theoretically, it is possible to sample each transfer function by holding the other input to zero, however this is not quite possible experimentally since controlling the human operator's behavior to that extent is challenging and not completely guaranteed. Thus, we need a generalized method to learn the individual transfer functions of a multi-input-single-output (MISO) system simultaneously, as proposed in [9], and is briefly discussed here for completeness.

Since the kernel of the sum of Gaussian processes is the sum of the kernels [48], the multi-input-single-output (MISO) system in (4.5) can be written as,

$$Y(\omega) = f_1(\omega)U_1(\omega) + f_2(\omega)U_n(\omega) + \epsilon, \tag{4.86}$$

where $f_1 = G_{inv,1}$, $f_2 = G_{inv,2}$, $U_1 = u_h$, $U_2 = y$, and $Y = y_d$, with the kernel function,

$$\begin{aligned} \hat{k}(\omega_1, \omega_2) &= \sum_{i=1}^{2} U_i(\omega_1)\hat{k}_i(\omega_1, \omega_2)U_i^*(\omega_2) \\ &= u_h(\omega)\hat{k}_1(\omega_1, \omega_2)u_h^*(\omega) + y(\omega)\hat{k}_2(\omega_1, \omega_2)y^*(\omega), \end{aligned} \tag{4.87}$$

and, the mean of the transfer functions are estimated as [48],

$$\hat{G}_{inv,1}(\omega) = E\left[f_1(\omega)\right], \quad \hat{G}_{inv,2}(\omega) = E\left[f_2(\omega)\right], \tag{4.88}$$

with $E\left[f_i(\omega)\right]$ for $i = 1, 2$ computed according to (4.82), and the variance in the transfer function estimates are given by,

$$\Delta_1(\omega) = \alpha\sqrt{\mathrm{Cov}\left[f_1(\omega)\right]}, \quad \Delta_2(\omega) = \alpha\sqrt{\mathrm{Cov}\left[f_2(\omega)\right]}, \tag{4.89}$$

with $\mathrm{Cov}\left[f_i(\omega)\right]$ for $i = 1, 2$ computed according to (4.83), and α depends on the level of confidence in the uncertainty estimate (e.g., $\alpha = 2$ for 95% confidence interval). Similarly, transfer functions $G_{H,1}$ and $G_{H,2}$ in the forward modeling equation in (4.7) can be estimated using the approach described above.

Remark 9 (MISO Uncertainties)**.** *The SISO model $\hat{G}^{-1}$ is related to the MISO models $\hat{G}_{inv,1}, \hat{G}_{inv,2}$, defined in* (4.5) *for each frequency ω as,*

$$\hat{G}^{-1}(\omega) = \hat{G}_{inv,1}(\omega)\hat{P}^{-1}(\omega) + \hat{G}_{inv,2}(\omega), \tag{4.90}$$

where $\hat{P}$ is a model of the controlled system P, defined in (4.1)*. Also, the equivalent uncertainty Δ in the model $\hat{G}^{-1}$ computed in* (4.90) *may be estimated from the MISO uncertainty estimates Δ_1, Δ_2 defined in* (4.89)*, and the uncertainty Δ_P in the controlled system (plant) inverse model $\hat{P}^{-1}$ for each frequency ω as,*

$$\Delta(\omega) \approx \alpha\sqrt{|\hat{G}_{inv,1}(\omega)|^2(\Delta_P(\omega))^2 + |\hat{P}^{-1}(\omega)|^2(\Delta_1(\omega))^2 + (\Delta_2(\omega))^2} \tag{4.91}$$

where α depends on the level of confidence in the model uncertainty (e.g., $\alpha = 2$ for 95% confidence interval).

Proof. Consider the models $\hat{G}^{-1}, \hat{G}_{inv,1}, \hat{G}_{inv,2}$ in (4.90) in terms of the real/imaginary components,

$$\begin{aligned}
\hat{G}^{-1}(\overline{\omega}) &= \hat{a}(\overline{\omega}) + j\hat{b}(\overline{\omega}),\\
\hat{G}_{inv,1}(\overline{\omega}) &= \hat{a}_1(\overline{\omega}) + j\hat{b}_1(\overline{\omega}),\\
\hat{G}_{inv,2}(\overline{\omega}) &= \hat{a}_2(\overline{\omega}) + j\hat{b}_2(\overline{\omega}),\\
\hat{P}^{-1}(\overline{\omega}) &= \hat{a}_p(\overline{\omega}) + j\hat{b}_p(\overline{\omega}),
\end{aligned} \tag{4.92}$$

where $a_1(\overline{\omega}), a_2(\overline{\omega}), a_p(\overline{\omega}), b_1(\overline{\omega}), b_2(\overline{\omega}), b_p(\overline{\omega}) \in \mathbb{R}$ for each frequency $\overline{\omega}$, which when substituted into (4.90) results in,

$$a(\overline{\omega}) = a_1(\overline{\omega})a_p(\overline{\omega}) - b_1(\overline{\omega})b_p(\overline{\omega}) + a_2(\overline{\omega}), \tag{4.93}$$

$$b(\overline{\omega}) = b_1(\overline{\omega})a_p(\overline{\omega}) + a_1(\overline{\omega})b_p(\overline{\omega}) + b_2(\overline{\omega}). \tag{4.94}$$

Algorithm 2: Non-parametric Intent Estimation (MISO)

Input : Training data $(y_{train}(t), y_{d,train}(t))$, Human input $r_{h,demo}(t)$, Human-demonstrated output $y_{demo}(t)$

Output : Intent estimate $\hat{y}_d(t)$

Parameter: Sampling frequency f_s Hz, Cutoff frequency f_c Hz, Confidence interval (e.g., 95%) for model uncertainty α

Training:

$Y_{train} = \mathcal{F}(y_{train})$, $Y_{d,train} = \mathcal{F}(y_{d,train})$

Estimate $\gamma, \sigma_f, \sigma_e$ based on $\hat{G}_T$ (4.84)

$\hat{G}_{inv,1}(\omega) = E[f_1(\omega)]$, $\hat{G}_{inv,2}(\omega) = E[f_2(\omega)]$ (4.88)

$\Delta_1(\omega) = \alpha\sqrt{\mathrm{Cov}[f_1(\omega)]}$, $\Delta_2(\omega) = \alpha\sqrt{\mathrm{Cov}[f_2(\omega)]}$ (4.89)

Compute equivalent $\hat{G}^{-1}(\omega)$ using (4.90)

Compute equivalent $\Delta_a(\omega) = \Delta_b(\omega) = \Delta(\omega)$ using (4.91)

Compute maximum upper bound $\overline{\Delta}(\omega)$ using Corollary 1

Compute activation index $\sigma(\omega)$ according to (4.79).

Human demonstration:

$Y = \mathcal{F}(y_{demo})$

Inversion-based Intent Estimate:

$\tilde{Y}_d(\omega) = \hat{G}^{-1}(\omega)Y(\omega)$ (4.5)

Blend intent estimate with demonstration:

$\hat{Y}_d(\omega) = \sigma(\omega)\tilde{Y}_d(\omega) + (1 - \sigma(\omega))Y(\omega)$

return $\hat{y}_d = \mathcal{F}^{-1}(\hat{Y}_d)$

Taking the derivative of both sides of (4.93) and (4.94) at frequency $\overline{\omega} = \omega$ results in,

$$\begin{aligned} \partial a(\omega) = \hat{a}_1(\omega)\partial a_p(\omega) + \hat{a}_p\partial a_1(\omega) \\ - \hat{b}_1(\omega)\partial b_p(\omega) - \hat{b}_p(\omega)\partial b_1(\omega) + \partial a_2(\omega) \end{aligned} \tag{4.95}$$

and,

$$\begin{aligned} \partial b(\omega) = \hat{b}_1(\omega)\partial a_p(\omega) + \hat{a}_p\partial b_1(\omega) \\ + \hat{a}_1(\omega)\partial b_p(\omega) + \hat{b}_p(\omega)\partial a_1(\omega) + \partial b_2(\omega) \end{aligned} \tag{4.96}$$

where $\hat{a}_1, \hat{a}_2, \hat{b}_1, \hat{b}_2, \hat{a}_p, \hat{b}_p$ are the mean estimates of $a_1, a_2, b_1, b_2, a_p, b_p$, respectively for each frequency ω. Now, the variance $\sigma_a^2(\omega)$ of $a(\omega)$ at each frequency ω is defined as, (suppressing ω for notational simplicity),

$$\begin{aligned} \sigma_a^2 &= E[(a - \hat{a})^2] \\ &= (\partial a)^2 \\ &= \left(\hat{a}_1\partial a_p + \hat{a}_p\partial a_1 - \hat{b}_1\partial b_p - \hat{b}_p\partial b_1 + \partial a_2\right)^2 \\ &\approx \hat{a}_1^2(\partial a_p)^2 + \hat{a}_p^2(\partial a_1)^2 + \hat{b}_1^2(\partial b_p)^2 + \hat{b}_p^2(\partial b_1)^2 + (\partial a_2)^2 \\ &= |\hat{G}_{inv,1}|^2\Delta_P^2 + |\hat{P}^{-1}|^2\Delta_1^2 + \Delta_2^2 \end{aligned} \tag{4.97}$$

and,

$$\begin{aligned} \sigma_b^2 &= E[(b - \hat{b})^2] \\ &= (\partial b)^2 \\ &= \left(\hat{b}_1\partial a_p + \hat{a}_p\partial b_1 + \hat{a}_1\partial b_p + \hat{b}_p\partial a_1 + \partial b_2\right)^2 \\ &\approx \hat{a}_1^2(\partial b_p)^2 + \hat{a}_p^2(\partial b_1)^2 + \hat{b}_1^2(\partial a_p)^2 + \hat{b}_p^2(\partial a_1)^2 + (\partial b_2)^2 \\ &= |\hat{G}_{inv,1}|^2\Delta_P^2 + |\hat{P}^{-1}|^2\Delta_1^2 + \Delta_2^2 \end{aligned} \tag{4.98}$$

where we may neglect cross correlations, and noting that according to Remark 8, $(\partial a_p)^2 = (\partial b_p)^2 = \Delta_P^2$, $(\partial a_1)^2 = (\partial b_1)^2 = \Delta_1^2$, and $(\partial a_2)^2 = (\partial b_2)^2 = \Delta_2^2$. □

4.5 Human-response modeling using CGPR

An output tracking experiment was conducted, involving multiple human subjects teleoperating a robot arm, in order to validate the proposed robust intent estimation method.

Specifically, the following points were investigated:

(RQ1) Generalizability of trained models to new tracking tasks

(RQ2) Benefit of robustness conditions in Lemma 1

(RQ3) Benefit of "softmax" blending in intent estimation

4.5.1 Apparatus

In this work, the human operator observed the desired (goal) trajectory y_d through an augmented reality (AR) headset, specifically the Microsoft Hololens, as an overlay on top of the end-effector of a Kinova Mico2 robot arm, as shown in Figure 4.6. The AR display consisted of the trajectory y_d traced in real-time by a green marker with/without a scrolling red band that represented the preview information of the y_d trajectory, overlaid on top of the robot's end-effector position y. The AR display was developed using the Unity game engine (in C#) before deploying a standalone Universal Windows Platform (UWP) application to the Hololens.

From the AR display, the operator then estimated the error $e = y_d - q$ and provided an input r_h to the system through a human machine interface, a Haptic controller in this case, which directly controlled the end-effector position of the robot arm through an internal proportional feedback velocity controller. The task of the human operator was to tele-operate the end-effector of the robot arm such that the error $e = y_d - y$ was minimized, i.e., the output trajectory y tracked the desired trajectory y_d as closely as possible.

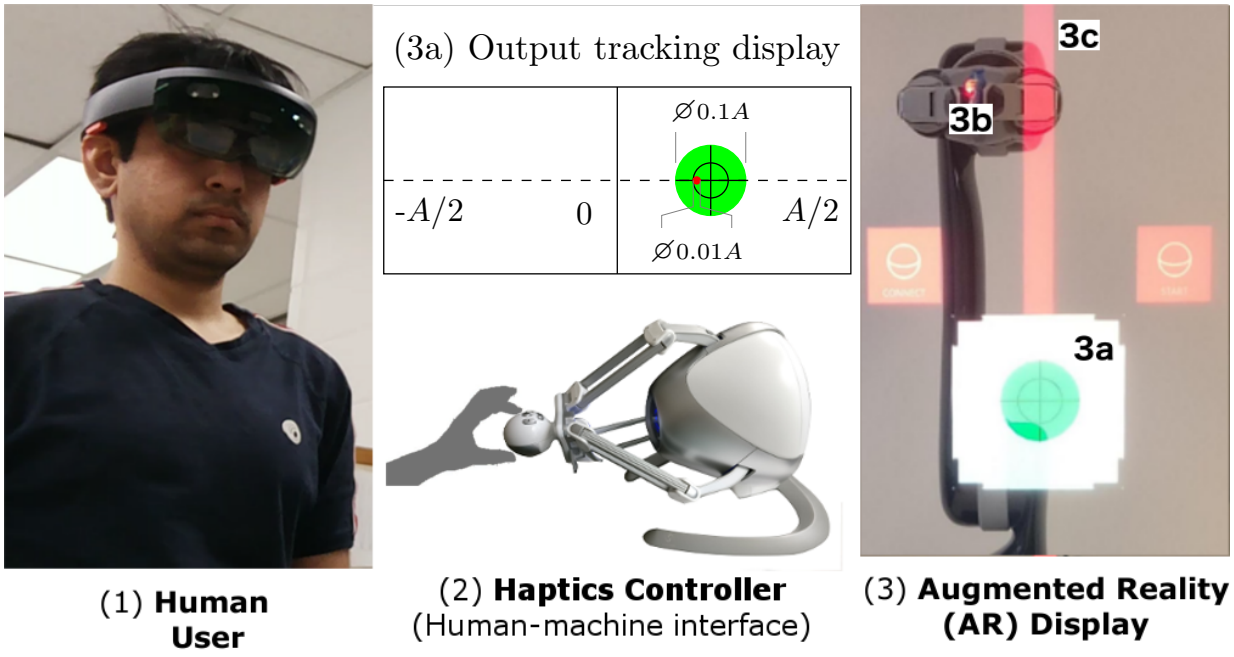


Figure 4.6: Experimental setup for human response modeling and intent estimation during tele-operated output tracking: (1) human operator; (2) haptics controller that provides input r_h to move the end-effector position y of the robot arm G_R; (3) Augmented Reality (AR) headset display showing the desired trajectory y_d overlaid on the robot arm end-effector; (3a) output-tracking target (green circle) with current desired trajectory point y_d located at the center marked by the circular cross-hair (3b) the end-effector position y of the robot G_R marked by a red led marker attached to the robot end-effector, (3c) preview of y_d trajectory depicted as a red band overlaid on the robot end-effector reference frame. The objective of the inference task is to estimate the intended goal trajectory y_d using the human-demonstrated output y.

The internal velocity controller was implemented in Cartesian space with the Application Programming Interface (API) available in the official Software Development Kit (SDK) provided by Kinova Robotics, written in C++. In this mode of operation, the linear speed of the robot was limited to 0.2 (m/s), which aids a human operator to safely interact with the robot. An outer proportional feedback loop was designed around the inner Kinova velocity controller. Thus, the controller was designed to track the robot command u, which resulted in the robot end-effector position y.

Table 4.1: Parameter values for CGPR Training

q_d	Param.	Description	Value
	n	no. of sinusoids	10
$y_{d,train}$	f_n	frequency of n^{th} sinusoid	$[0.05, 1)$ Hz
	K_n	amplitude of n^{th} sinusoid	see Fig. 4.7
	t_{start}	rest-period	5 s
	A	maximum amplitude	10 cm

4.5.2 Procedure: Training Phase

Eight human subjects participated in the experiment, two female and six male adults with ages ranging from 20-30 years. Each participant was given a brief explanation of the setup and a trial round to familiarize with the objectives of the task.

The reference trajectory $y_d = y_{d,train}$ for the model estimation routine was a randomly generated sum-of-sines trajectory as shown in Fig. 4.7 given by

$$y_{d,train}(t) = K_0 + \sum_{i=1}^{n} K_n \sin(2\pi f_n t + \phi_n) \tag{4.99}$$

Table 4.2: Zero-error for output tracking of each subject

Subject	$\overline{e}_{DC}$	
	No goal-preview	1 s goal-preview
1	0.08 ± 0.06 %	0.06 ± 0.04 %
2	6.13 ± 3.28 %	7.22 ± 6.49 %
3	5.26 ± 5.15 %	9.19 ± 6.55 %
4	3.21 ± 3.41 %	2.83 ± 1.37 %
5	2.81 ± 2.46 %	5.48 ± 6.15 %
6	2.67 ± 1.41 %	2.66 ± 2.09 %
7	5.05 ± 7.30 %	3.39 ± 2.29 %
8	3.04 ± 1.88 %	7.29 ± 9.54 %

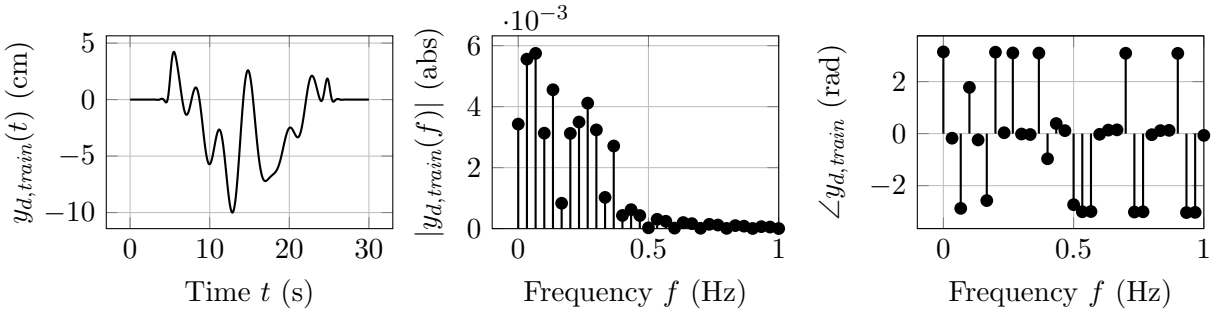


Figure 4.7: Desired output trajectory $y_d = y_{d,train}$ during training phase (one instance out of 10 randomized trials).

where, where n is the number of frequencies, in our case ten, $f_n \in [0.05, 1)$ Hz, K_0 is a gain tuned to prevent saturation of the robot end effector velocity, (K_n, ϕ_n) are amplitudes and phase shifts for each frequency f_n of the sum of sines trajectory given by,

$$\begin{aligned} K_n &= \begin{cases} K, & \text{for } n < 0.5 f_c/f_0, \\ K e^{-10 n f_0/f_c}, & \text{for } n \geq 0.5 f_c/f_0 \end{cases}, \\ \phi_n &= \begin{cases} \phi, & \text{for } n < 0.5 f_c/f_0, \\ \phi e^{-10 n f_0/f_c}, & \text{for } n \geq 0.5 f_c/f_0 \end{cases} \end{aligned} \tag{4.100}$$

where $f_0 = 0.05$ Hz, $f_c = 1$ Hz is the cutoff frequency, $K \in [0.045, 0.055]$ m is the amplitude (chosen randomly from the set), and $\phi \in [0, 5\pi]$ is the phase shift (chosen randomly from the set), and finally filtered using a zero-phase low-pass filter of the form,

$$G_f(\omega) = \left(\frac{a}{s+a}\right)^* \left(\frac{a}{s+a}\right), \tag{4.101}$$

where $a = 2\pi f_c$ with the cutoff frequency $f_c = 1$ Hz, see Fig. 4.7 for an example trajectory.

The sum of sines trajectories (4.99) with randomized amplitude and phase shift were run for ten trials with each trial being one instance of (K, ϕ) chosen randomly from the set $([0.045, 0.055], [0, 5\pi])$ for the following test conditions:

(i) *without goal-preview* (only the green marker representing the current desired position y_d in Fig. 4.6 is displayed to the human operator),

(ii) *with 1 s of goal-preview* (with the red band of length representing the goal-preview of 1 s in Fig. 4.6 along with the green marker are displayed to the human operator).

Each trial lasted 30 seconds with a 5 s rest at the beginning and end of the 20 s long goal trajectory $q_{d,train}$. The randomization of the amplitude and phase of the sum-of-sines training trajectories is important to ensure that the human response is not biased by the specific

reference trajectory, and is only influenced by the above test conditions. The collected data consisted of desired position y_d, output position y, and human input u_h for each trial.

Further, Table 4.1 tabulates $\overline{e}_{DC,i}$ for each subject $i \in [1, 8]$ which represents the level of expertise of each of the subjects in reaching the center of the target when the target is not moving, defined as,

$$\overline{e}_{DC,i} = \frac{1}{N} \sum_{t_k \in [0, t_{start}]} \frac{|y_{d,train}(t_k) - y(t_k)|}{\max_{t \in [0,T]} |y_{d,train}(t)|} \times 100\% \tag{4.102}$$

for each subject $i = 1, 2, \ldots, 8$ where N is the number of samples collected in the interval $[0, t_{start}]$. For example, subject 4 had a zero-error of $\overline{e}_{DC,4} = 3.21 \pm 3.41\%$ when no goal-preview was displayed compared to subject 1 with $\overline{e}_{DC,1} = 0.08 \pm 0.06\%$, which indicates that subject 1 has potentially more expertise in the tracking task than subject 4.

4.5.3 Goal-preview improves manual output tracking

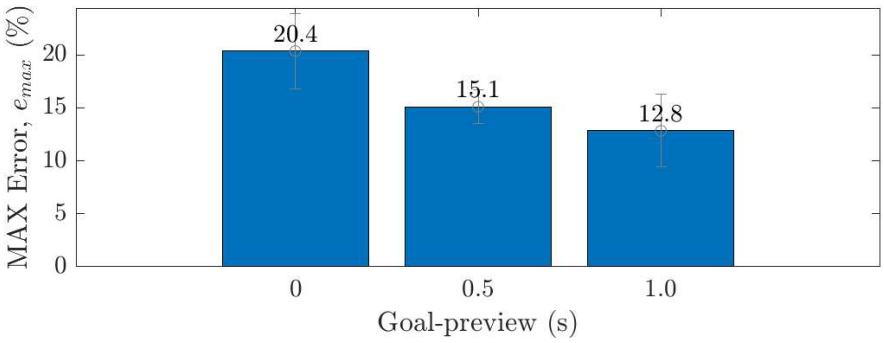


Figure 4.8: Manual tracking error e_{max}, defined in (3.72) while tracking desired output trajectory $y_{d,test}$ for subject 1. Increased goal-preview displayed to the human operator reduced the manual tracking error.

The tracking performance of the human operator was evaluated using the maximum absolute tracking error e_{max}, defined in (3.72) while the intent estimation performance was

evaluated using the estimates $\hat{e}_{max}$ defined as,

$$\hat{e}_{max} = \frac{\max_{t_i \in [0,T]} |y_d(t_i) - \hat{y}_d(t_i)|}{A} \times 100\%, \tag{4.103}$$

$$\tag{4.104}$$

where, T is the time-period of each trial in seconds, N is the number of samples collected during each trial, and A is the amplitude of the desired output y_d.

Fig. 4.8 depicts the manual tracking error summary for varying levels of goal-preview displayed to the human operator, and it is observed that the human operator's output tracking performance improved when larger amount of goal-preview was displayed, e.g., the maximum absolute tracking error $e_{max} = 20.4\%$ when no goal-preview was displayed compared to $e_{max} = 12.8\%$ when 1 s of goal-preview was displayed, which corresponds to an average reduction of 37%. This indicates that the availability of goal-preview makes it easier for the human operator to perform output tracking — alternatively, reduced goal-preview affects the ability of the human operator to demonstrate the intended goal trajectory accurately, motivating the need to invert the human response, especially when goal-preview is decreased.

4.5.4 Results: (RQ1) - Generalizability of trained models

The primary question that needed to be addressed was - Do the trained inverse models predict the human intent in a new output tracking task?

In this regard, the mean estimates of the inverse models $\hat{G}_1, \hat{G}_2$ each defined in (4.5), and their corresponding uncertainty estimates Δ_1, Δ_2 were computed using the proposed CGPR approach applied to the training data, as shown in Fig. 4.10 for an expert subject (Subject 1) and Fig. 4.11 for a novice subject (Subject 5). The mean estimates are depicted using solid curves and the uncertainty estimates are depicted using the shaded regions.

Next, Algorithm 2 was applied to estimate the unknown intended trajectory $y_{d,test}$ (defined

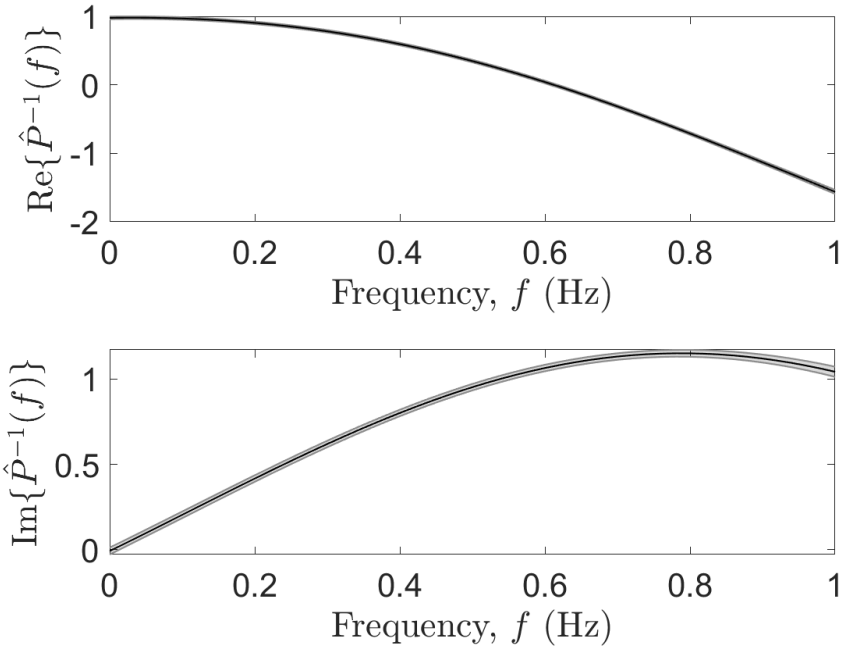
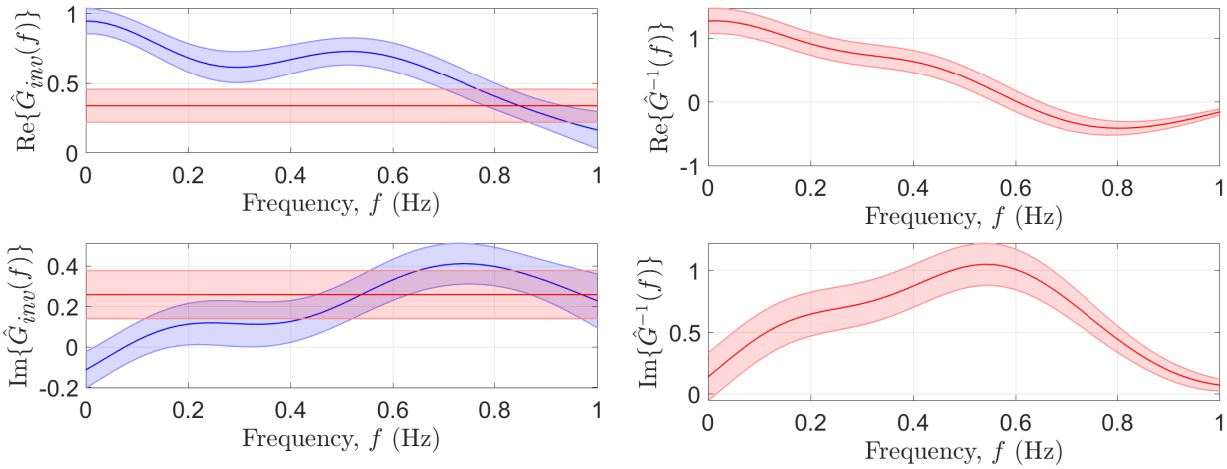
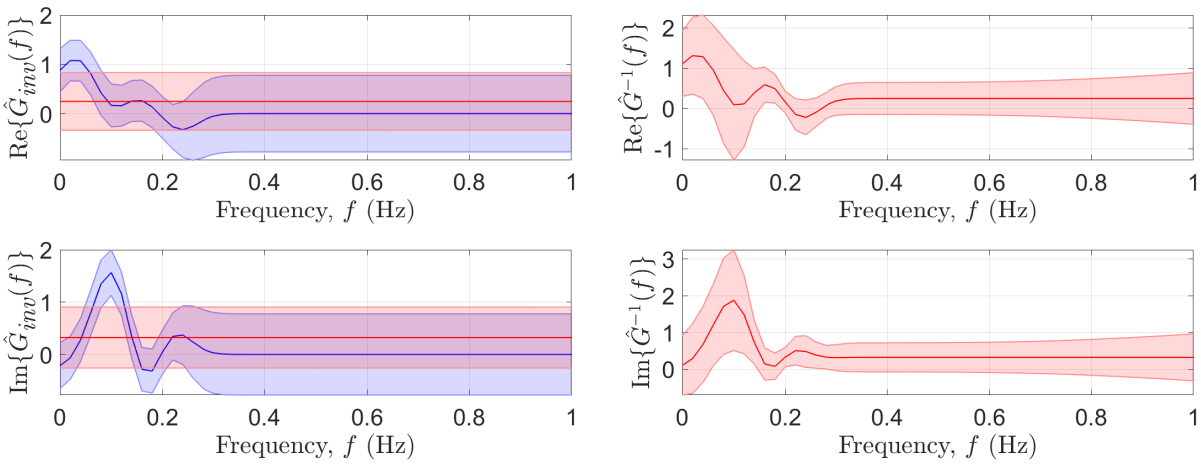


Figure 4.9: Frequency response of the controlled system inverse dynamics model $\hat{P}^{-1}(f)$ for frequencies $f \in [0, 1]$ Hz obtained using the CGPR method described in Sec. 4.4.4. The mean estimate $\hat{P}^{-1}$ is depicted as the solid line and the shaded region depicts the uncertainty estimate Δ_P.



(a) $\hat{G}_1(f), \hat{G}_2(f)$ (b) $\hat{G}^{-1}(f)$

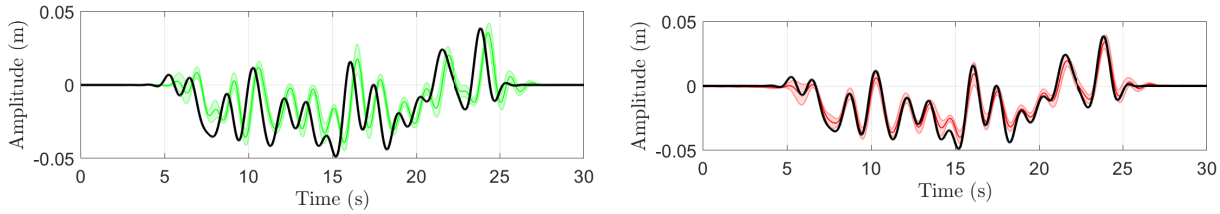
Figure 4.10: Expert Subject (Subject 1) frequency response of the inverse dynamics models (a) $\hat{G}_1$ (shown in blue), $\hat{G}_2$ (shown in red), defined in (4.88), and (b) $\hat{G}^{-1}$ defined in (4.90). The mean estimates are depicted as solid curves while the uncertainty estimates are depicted using the shaded region.



(a) $\hat{G}_1(f), \hat{G}_2(f)$ (b) $\hat{G}^{-1}(f)$

Figure 4.11: Novice Subject (Subject 5) frequency response of the inverse dynamics models (a) $\hat{G}_1$ (shown in blue), $\hat{G}_2$ (shown in red), defined in (4.88), obtained through the CGPR method described in Sec. 4.4.4, and (b) $\hat{G}^{-1}$ defined in (4.90). The mean estimates are depicted as solid curves while the uncertainty estimates are depicted using the shaded region.

similar to the training trajectories $y_{d,train}$ in (4.99) but with a new set of amplitude and phase (A, ϕ)) in a new output tracking task. The test trajectory $y_{d,test}$ was chosen with a cutoff frequency $f_c = 1$ Hz.



(a) Human Demonstration y vs. Intent y_d

(b) Intent Estimate $\hat{y}_d$ vs. Intent y_d

Figure 4.12: Intent Estimation during the test trials with an expert subject (Subject 1) to track the intended trajectory $y_{d,test}$ (shown in black) using (a) the human-demonstrated system output y (shown in green), (ii) the estimate $\hat{y}_d$ computed using the MISO inverse model $[\hat{G}_1, \hat{G}_2] : (r_h, y) \rightarrow y_d$ (shown in red). The solid lines depict the mean and the shaded regions depict one standard deviation over 10 trials. It is seen that the intent estimate $\hat{y}_d$ tracks the intended trajectory $y_{d,test}$ better (especially by correcting the phase lag) than the human-demonstrated system output y.

Fig. 4.12 depicts the intent estimate $\hat{y}_d$ computed using the MISO inverse models $[\hat{G}_1, \hat{G}_2]$ (shown in red), compared to the human-demonstrated system output y (shown in green) and the intended trajectory $y_{d,test}$ (shown in black). It can be seen that the major improvement due to the inversion-based intent estimation is the correction in the phase lag seen between the human-demonstrated system output y and the intended trajectory $y_{d,test}$. Tables 4.16 and 4.17 tabulate the tracking error metric e_{max} for each test case (i) with 1 s goal-preview and (ii) with no goal-preview, respectively, with the reduction $\%E_{red}$ defined as,

$$\%E_{red} = \frac{e_{max} - \hat{e}_{max}}{e_{max}} \times 100\% \tag{4.105}$$

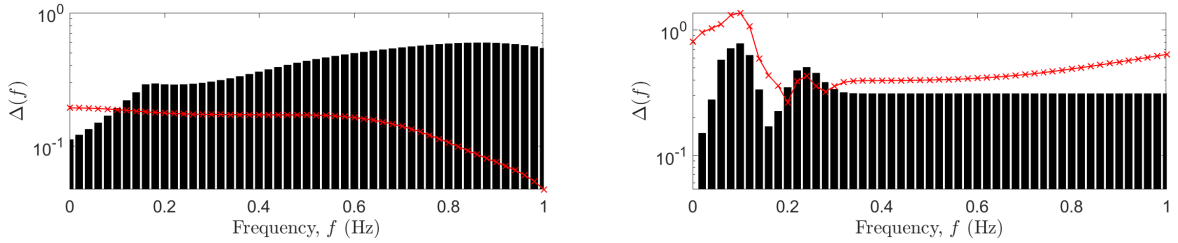
and it can be seen that for the case with 1 s goal-preview, there is an average of $\%E_{red} = 32\%$ reduction in the tracking error due to the proposed inversion-based intent estimation

compared to the human-demonstrated system output for the reference test trajectory $y_{d,test}$. Similar results are obtained for the test case with no goal-preview as shown in Table 4.16. In summary, the results show that overall the inverse models learned using the training data have generalized well to the new output tracking tasks.

4.5.5 Results: (RQ2) - Benefit of robustness conditions in Lemma 1

Next, do the robustness conditions in Lemmas 1 predict when the naive model-inversion results in inaccurate intent estimation?

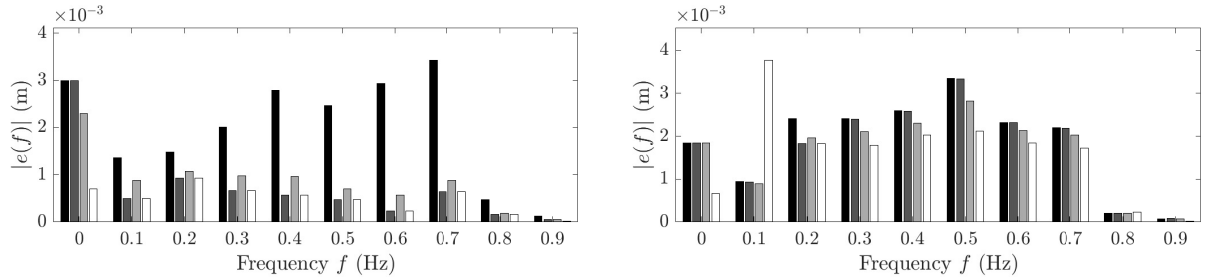
The estimated inverse models result in reduced intent estimation error at some frequency ω compared to the human demonstrations when the condition in (4.23) is satisfied at that frequency. Lemma 1 bounds the size of the uncertainty of the CGPR estimated inverse model for successful intent estimation at a frequency ω, i.e., the estimated inverse model is valid only at frequencies ω where the magnitude of the modeling uncertainty is small according to the conditions in Lemma 1.



(a) Subject 1

(b) Subject 5

Figure 4.13: Modeling uncertainty Δ (shown as red asterisk markers) compared to the maximum allowed uncertainty $\overline{\Delta}$ (shown as black bars) for each frequency $f \in [0, 1]$ Hz according to Lemma 1 for (a) Subject 1, (b) Subject 5 with no goal-preview displayed. Subject 1 has lower uncertainty Δ that satisfies the conditions in Lemma 1 at all frequencies $f < 1$ Hz, compared to Subject 5 whose modeling uncertainty Δ violates the conditions in Lemma 1 at some frequencies $f \in [0, 1]$ Hz, and so is not guaranteed to reduce intent estimation error at those frequencies for Subject 5.



(a) Subject 1 (b) Subject 5

Figure 4.14: Human-demonstration tracking error magnitude $|e(f)| = |y_d(f) - y(f)|$ (shown as black bars) compared to the intent estimation error magnitude $|\hat{e}(f)| = |y_d(f) - \hat{y}_d(f)|$: (1) using Lemma 1 with "hardmax" intent estimation in (4.77) (shown as solid gray bars), (2) using Lemma 1 with "softmax" intent estimation in (4.78) (shown as lighter gray bars), and (3) without applying Lemma 1 (shown as white bars), for each frequency $f \in [0, 1]$ Hz for (a) Subject 1, and (b) Subject 5, with no goal-preview displayed. According to uncertainty bounds in Fig. 4.13, intent estimation error is guaranteed (with 95% confidence) to be smaller than the human-demonstrated tracking error for Subject 1, while for Subject 5 the conditions in Lemma 1 ensure the modeling error at frequency $f = 0.1$ Hz is compensated by the conditions in Lemma 1.

For example, for subject 1, the training data had the least variance among all the subjects resulting in a large bound on modeling uncertainty Δ, as seen in Fig. 4.13a. This resulted in reduced intent estimation error $\hat{e}(f)$ compared to the human-demonstrated tracking error $e(f)$ at each frequency $f \in [0, 1]$ Hz, as seen in Fig. 4.14a, and in Table 4.17 for subject 1. In contrast, the training data from subject 5 when no goal-preview was displayed showed significant variance from trial to trial resulting in the modeling uncertainty $\Delta > \overline{\Delta}$ for all frequencies $f \in [0, 1]$ Hz, as seen in Fig. 4.13b, resulting in an increase in the intent estimation error using the computed inverse models at frequency $f = 0.1$ Hz, as seen in Fig. 4.14b for each frequency $f \in [0, 1]$ Hz, resulting in only a reduction of $\%E_{red} = 1\%$ without applying the proposed algorithm in Algorithm 2 compared to 7% improvement when Algorithm 2 was applied, as shown in Table 4.17 for subject 5.

4.5.6 Results: (RQ3) - Benefit of "softmax" blending in intent estimation

Next, does the blending of the human demonstrations and the naive model-inversion-based intent estimate result in improvements over

- naive model-inversion-based intent estimation ?
- robust model-inversion-based intent estimation without blending demonstrations and intent estimates?

Using the "hardmax" intent estimation, as defined in (4.77) guarantees robustness according to Lemma 1, as seen in Fig. 4.14b, where at frequency $f = 0.1$ Hz for Subject 5, there is a large intent estimation error when naively inverting the learned inverse model at this frequency, which is successfully avoided by applying the robustness conditions from Lemma 1. But, in other cases, when the model uncertainty is low/moderate it is seen that the "hardmax" approach is too restrictive, and the intent estimation performance could be potentially improved by using the "softmax" intent estimation method in (4.78) which blends the model-inversion-based intent and the demonstration based on the normalized modeling uncertainty $\Delta/\overline{\Delta}$ at each frequency ω. This is observed for Subject 1 in Fig. 4.14a, where it is seen that the intent estimation error using the "softmax" approach leads to improvement over the demonstration and the "hardmax" intent estimation error.

4.5.7 Realizing the estimated intent

Finally, once the estimate $\hat{y}_d$ of the intended trajectory y_d is obtained, an iterative learning scheme, e.g., [58] where the input u_k at each iteration step k is updated for each frequency ω as,

$$u_{k+1}(\omega) = u_k(\omega) + K_\rho(\omega)\hat{P}^{-1}(\omega)(\hat{y}_d(\omega) - y_k(\omega)), \tag{4.106}$$

where y_k is the robot system output at each iteration step k, and K_ρ is the iteration gain that depends on the modeling uncertainty in $\hat{P}$. When the learning scheme in (4.106) converges, the system output y converges to the estimated intent $\hat{y}_d$ as shown in Fig. 4.15.

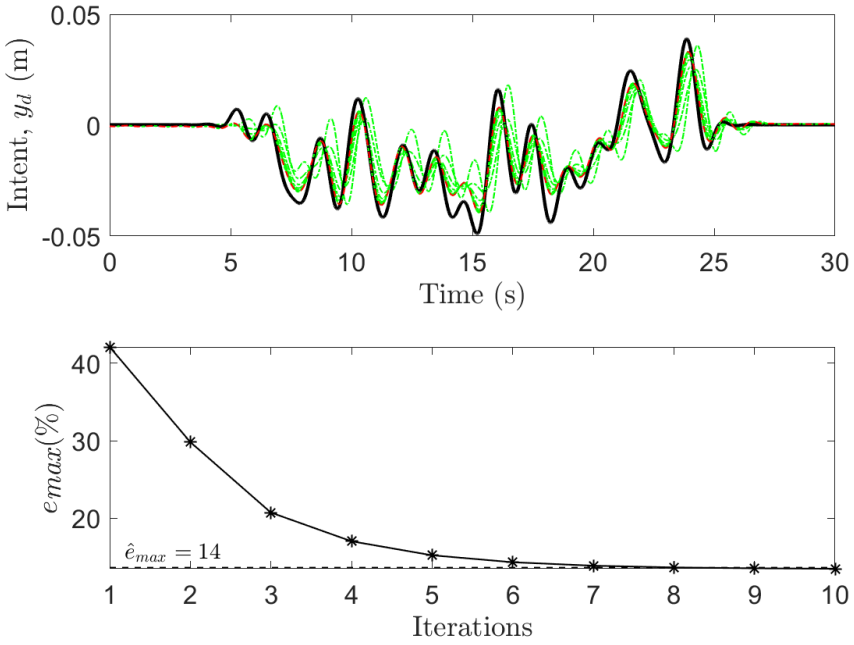


Figure 4.15: Inversion-based iterative learning control used to converge the robot system output y (iterations shown as green dash-dotted lines, converged output shown as red dashed lines) to the intended trajectory y_d (shown as black solid line) once an intent estimate $\hat{y}_d$ was obtained using the Algorithm 2 for Subject 1 with no goal-preview displayed.

Thus, we see that overall the model-inversion-based intent estimates specify the unknown intent of a human operator during teleoperation better than the human demonstrated output, even for expert operators. Further, it was seen that with novice operators there is typically large variance in their tracking performance, which when used to estimate dynamics models results in large modeling uncertainties. But, the proposed robust intent estimation method accounted for these large modeling uncertainties by blending in the human demonstrations

at frequencies where the inverse-models are predicted to not give improvements in intent estimation. In this way, the proposed method may be used to get better task representations using learning from demonstration methods by improving the estimated intent underlying the human demonstrations.

4.6 Conclusions

In conclusion, this chapter demonstrated the need for estimating the underlying intent of human-in-the-loop demonstrations using data-based model inversion. Specifically, inverse models of the closed-loop system including the human operator were estimated using the non-parametric complex-valued Gaussian Process Regression method on data collected during the training phase. Further, a necessary and sufficient condition on the size of the modeling uncertainty with respect to the computed model was introduced which provides a method to predict the validity of the computed models a priori, so that the training may be repeated, if necessary. Finally, it was shown that on an average the intent estimation using the proposed model-inversion method achieved approximately 35% reduction in the tracking error as demonstrated by the human operator, which allows for improving human-guided learning algorithms to better estimate the underlying human intent. Matlab code and experimental data for this section are hosted at this repository [1], and may be used freely, according to the stipulations of the associated licenses on the scripts.

[1] `https://github.com/jaguar243/robust-intent-estimation/tree/306583bcb70912b81f31b9c5c892fd556bf7012b`

Figure 4.16: Maximum absolute intent estimation error e_{max}[1] on test trajectory $y_{d,test}$ with 1s goal-preview

Subject	e_{max}	$\hat{y}_{d,1}$ (SISO)		$\hat{y}_{d,2}$ (MISO)	
		$\hat{e}_{max}$[2]	$\%E_{red}$[3]	$\hat{e}_{max}$[2]	$\%E_{red}$[3]
1	29.06 ± 3.68	15.53 ± 3.91 $[16.33 \pm 4.14]$ (16.42 ± 4.25)	47 [44] (44)	19.29 ± 3.27 $[24.26 \pm 4.83]$ (15.37 ± 4.48)	34 [17] (47)
2	46.03 ± 2.67	37.93 ± 3.02 $[34.09 \pm 3.24)$ (33.36 ± 2.79)	18 [26] (28)	36.63 ± 3.94 $[35.89 \pm 4.51]$ (36.22 ± 4.57)	20 [22] (21)
3	43.64 ± 3.44	34.72 ± 2.54 $[32.49 \pm 3.11]$ $(32.94 \pm 2.67]$	20 [26] (25)	32.02 ± 2.33 $[33.32 \pm 4.14]$ (34.13 ± 3.10)	27 [24] (22)
4	42.40 ± 3.71	16.10 ± 9.48 $[13.82 \pm 11.24]$ (13.97 ± 11.38)	62 [67] (67)	18.82 ± 6.81 $[17.50 \pm 7.47]$ (16.50 ± 9.45)	56 (59) (61)
5	46.91 ± 4.86	39.13 ± 3.43 $[36.75 \pm 3.83]$ (31.99 ± 2.91)	17 [22] (32)	38.75 ± 3.29 $[37.45 \pm 3.16]$ (36.98 ± 2.22)	17 [20] (21)
6	40.60 ± 2.69	24.46 ± 5.26 $[24.45 \pm 6.08]$ (24.44 ± 6.11)	40 [40] (40)	24.66 ± 4.48 $[21.96 \pm 6.16]$ (23.94 ± 6.02)	39 [46] (41)
7	48.43 ± 3.92	33.07 ± 3.86 $[32.30 \pm 3.45]$ (28.85 ± 3.75)	32 [33] (40)	30.13 ± 4.51 $[29.04 \pm 4.35]$ (25.67 ± 3.70)	38 [40] (47)
8	50.32 ± 5.84	36.25 ± 3.65 $[35.33 \pm 3.50]$ (33.52 ± 2.36)	28 [30] (33)	39.18 ± 4.49 $[37.83 \pm 4.30]$ (32.87 ± 2.81)	22 [25] (35)
overall	43.42 ± 6.62	29.65 ± 9.63 $[28.20 \pm 8.91]$ (26.94 ± 7.88)	33 ± 16 $[36 \pm 15]$ (38 ± 13)	29.94 ± 8.26 $[29.66 \pm 7.71]$ (27.71 ± 8.64)	32 ± 13 $[31 \pm 15]$ (37 ± 15)

[1] $[\mu_x \pm \sigma_x] \times 100\%$, where $\mu_x = \sum_{i=1}^{n} x_i/n$ is the mean, and $\sigma_x = \sqrt{\sum_{i=1}^{n} x_i^2/(n-1)}$ is one standard deviation for $n = 10$ test trials for each performance metric x,

[2] values reported for each subject as

- first row: "softmax" intent estimation method defined in (4.78).
- second row: "hardmax" intent estimation method defined in (4.77).
- third row: "naive" intent estimation method (robustness conditions in Lemmas 1 and 2 not applied).

[3] defined in (4.105).

Figure 4.17: Maximum absolute intent estimation error e_{max}[1] on test trajectory $y_{d,test}$ with no goal-preview

Subject	e_{max}	$\hat{y}_{d,1}$ (SISO)		$\hat{y}_{d,2}$ (MISO)	
		$\hat{e}_{max}$[2]	$\%E_{red}$[3]	$\hat{e}_{max}$[2]	$\%E_{red}$[3]
1	42.06 ± 5.56	18.59 ± 4.52 $[15.18 \pm 5.09]$ (15.34 ± 5.13)	56 [64] (64)	18.16 ± 4.57 $[16.48 \pm 4.61]$ (13.64 ± 5.85)	57 [61] (68)
2	47.07 ± 2.84	40.50 ± 4.53 $[39.36 \pm 4.44)$ (39.40 ± 4.46)	14 [16] (16)	33.59 ± 3.91 $[34.76 \pm 4.51]$ (34.43 ± 4.53)	29 [26] (27)
3	50.84 ± 4.03	39.86 ± 3.75 $[39.93 \pm 4.25]$ $(32.66 \pm 2.43]$	22 [21] (36)	38.88 ± 2.99 $[41.53 \pm 3.15]$ (34.50 ± 2.50)	24 [18] (32)
4	45.20 ± 2.45	29.44 ± 4.66 $[27.57 \pm 4.48]$ (27.41 ± 4.37)	35 [39] (39)	27.39 ± 4.02 $[26.83 \pm 3.62]$ (27.81 ± 4.06)	39 (41) (38)
5	43.42 ± 5.98	44.67 ± 6.64 $[42.28 \pm 5.92]$ (45.81 ± 7.74)	-3 [3] (-6)	39.66 ± 4.32 $[40.27 \pm 3.75]$ (43.06 ± 8.57)	9 [7] (1)
6	43.86 ± 5.36	25.38 ± 8.01 $[21.76 \pm 8.11]$ (21.93 ± 8.04)	42 [50] (50)	33.98 ± 6.27 $[35.49 \pm 6.36]$ (30.80 ± 6.07)	23 [19] (30)
7	44.67 ± 9.91	35.50 ± 10.12 $[34.03 \pm 11.52]$ (31.74 ± 13.09)	21 [24] (29)	35.23 ± 11.64 $[34.30 \pm 13.50]$ (32.79 ± 16.00)	21 [23] (27)
8	45.34 ± 6.85	38.20 ± 4.55 $[37.65 \pm 6.09]$ (34.80 ± 5.61)	16 [17] (23)	38.59 ± 4.60 $[39.19 \pm 4.49]$ (37.31 ± 4.61)	15 [14] (18)
overall	45.31 ± 2.68	34.02 ± 8.81 $[32.22 \pm 9.75]$ (31.14 ± 9.62)	25 ± 18 $[29 \pm 20]$ (31 ± 21)	33.18 ± 7.25 $[33.61 \pm 8.31]$ (31.79 ± 8.61)	27 ± 15 $[26 \pm 17]$ (30 ± 19)

[1] $[\mu_x \pm \sigma_x] \times 100\%$, where $\mu_x = \sum_{i=1}^{n} x_i/n$ is the mean, and $\sigma_x = \sqrt{\sum_{i=1}^{n} x_i^2/(n-1)}$ is one standard deviation for $n = 10$ test trials for each performance metric x,

[2] values reported for each subject as

- first row: "softmax" intent estimation method defined in (4.78).
- second row: "hardmax" intent estimation method defined in (4.77).
- third row: "naive" intent estimation method (robustness conditions in Lemmas 1 and 2 not applied).

[3] defined in (4.105).

Chapter 5

ONE-SHOT INTENT ESTIMATION - TIME-DOMAIN

5.1 Introduction

The research problem (P3), i.e., one-shot learning of human intent estimation, is addressed in this Chapter with a special emphasis on time-domain implementation and its associated challenges. In Section 3.2, it was shown that the underlying unknown intent may be inferred by inverting known models of human response, and this was extended to situations where nominal models are not directly applicable in Section 3.4. Section 3.2.2 discussed the effect of modeling uncertainty in a robust manner using a frequency-domain iterative learning controller that converges the overall system output to the unknown human intent, and it was shown, both theoretically and experimentally, that it improves tracking performance over just manual control. Chapter 4 described a conservative model-inversion method where trained non-parametric frequency response models of the human-in-the-loop dynamics could be inverted to obtain an estimate of the human intent without iterations. But, so far the model-inversion methods were restricted to an offline setting where the intent is inferred batch-wise, primarily in the frequency-domain.

This chapter considers the case of estimating the human intent directly in the time-domain. This is important when batch-wise or iterative methods are not suitable for the given output tracking task, e.g., in tele-operating surgical robots or in determining the controller input in powered prostheses based on real-time measurements. Briefly, the proposed approach is still primarily trying to invert the human response, and so the first step is to learn an impulse response model of the inverted human response, which is then converted to a set of FIR filters for each input channel of the learned model.

The main research questions that are addressed in this chapter are as follows:

(RQ1) Are time-domain impulse response functions computed from non-parametric frequency response models similar to impulse-response functions obtained directly in time-domain?

(RQ2) Empirically, what is the effect of preview information on the intent estimation performance using the learned inverse impulse response functions, specifically, how much preview-time is required for a specific level of intent estimation accuracy?

(RQ3) Empirically, what amount of preview information is required to estimate intent using the learned inverse impulse response functions such that tracking error is reduced compared to the human demonstrations?

First, a brief description of non-parametric impulse response estimation from given input/output training data is presented, which forms the basis for more sophisticated techniques such as kernel-based regression techniques in time-domain [47], similar to the frequency-domain CGPR method described in Section 4.4.4. Finally, human-in-the-loop output tracking data from multiple human subjects is used to study the amount of preview information required for successful intent estimation using the learned inverse impulse response functions.

5.2 Problem Formulation

Briefly, the idea is to compute the impulse response models that relate the available data, i.e., the human input, r_h and the system output y to the unknown desired output (intent) y_d, as shown in Fig. 1.1.

Firstly, in the forward direction the human input r_h for some time t is given by

$$r_h(t) = K_1 \left[(G_{H,1} * y_d)(t) + (G_{H,2} * y)(t) \right], \tag{5.1}$$

where, K_1 represents the human-machine interface which is usually just a proportional gain, $G_{H,1}, G_{H,2}$ are the impulse response functions from $y_d \rightarrow r_h$ and $y \rightarrow r_h$, respectively, and

the $*$ refers to the convolution operation, given by,

$$(G_{H,1} * y_d)(t) = \int_{t-T_1}^{t+T_2} G_{H,1}(t-\tau) y_d(\tau)\, d\tau \tag{5.2}$$

for some time instant t, and similarly for the other input channel.

In an analogous manner, the proposed method assumes that the intended goal y_d is also related to the human input r_h and the system output y through a set of impulse response functions, g_1, g_2 for some time t as,

$$y_d(t) = (g_1 * r_h)(t) + (g_2 * y)(t). \tag{5.3}$$

Thus, if we know g_1, g_2 then we can recover the intent y_d from u_h and y.

Problem Statement: Find the inverse impulse response functions g_1, g_2 given training data $[r_h(t), y(t), y_d(t)]$ for some time instances t.

5.2.1 Discrete-time De-convolution

The continuous time convolution in (5.3) may be converted to discrete time as,

$$y_d(t) = \sum_{t_k=t-N_1T_s}^{t+N_2T_s} g_1(t-t_k) r_h(t_k) + \sum_{t_k=t-\tilde{N}_1T_s}^{t+\tilde{N}_2T_s} g_2(t-t_k) y(t_k) = \begin{bmatrix} R_H(t) & \vdots & Y(t) \end{bmatrix} \begin{bmatrix} G_1(t) \\ \cdots \\ G_2(t) \end{bmatrix} \tag{5.4}$$

where $R_H(t), Y(t)$ are convolution matrices for r_h and y at time t whose elements are the respective signals sampled during the time $t \in [t - N_1T_s, t + N_2T_s]$ where T_s is the sampling rate and N_1, N_2 are the pre-/post-view buffer lengths, respectively. The weights G_1, G_2 are then computed by solving the least squares problem,

$$\min_G \|Y_d - UG\|_2^2 \tag{5.5}$$

where $G = \begin{bmatrix} G_1 \\ G_2 \end{bmatrix}$, $U = \begin{bmatrix} R_H(t) & \vdots & Y(t) \end{bmatrix}_{t=0:T}$ and $Y_d = [y_d(t)]_{t=0:T}$ where T is the time period of each training cycle.

5.2.2 Kernel-based Regression

Causal impulse response estimation using kernel-based regression techniques are discussed in [47], but we would like to get non-causal impulse response estimates since it is commonly observed that human operators utilize significant amount of preview information during human-in-the-loop output tracking. The previous section described a naive approach to estimating the discretized impulse response estimate, but this does not provide any information on the model uncertainty and significantly depends on the chosen discretization parameters.

In contrast, this section describes obtaining a non-parametric impulse response estimate by taking the inverse Fourier transform of the non-parametric frequency response obtained through the CGPR method in frequency-domain, as described in Section 4.4.4. For example, consider the mean and variance estimates $(\hat{G}(\omega), \sigma(\omega))$ of some unknown transfer function G obtained at some frequencies ω using the CGPR method in the frequency-domain. Then, the equivalent time-domain impulse response model $\hat{g}(t)$ is given for each time t as,

$$\hat{g}(t) = \mathcal{F}^{-1}(\hat{G}(\omega)) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \hat{G}(\omega) e^{j\omega t} \, d\omega, \tag{5.6}$$

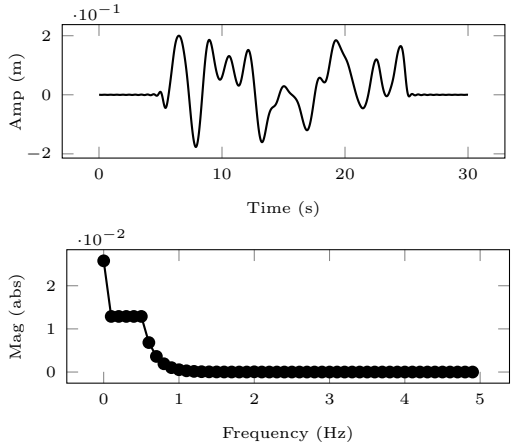
which may further be approximated using the Discrete Fast Fourier Transform algorithm. Since, the inverse Fourier transform is a linear function the variance σ in the frequency-domain model $\hat{G}(\cdot)$ corresponds to the variance in the time-domain model $\hat{g}(\cdot)$.

5.3 Multiple subject impulse response estimation results

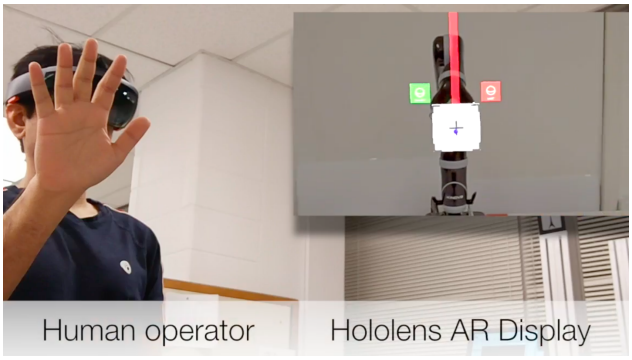
5.3.1 Procedure

The desired output y_d was chosen as a sum-of-sines trajectory with frequencies $f_T \in [0.05, 5]$ Hz with random phase components, see Fig. 5.1a.

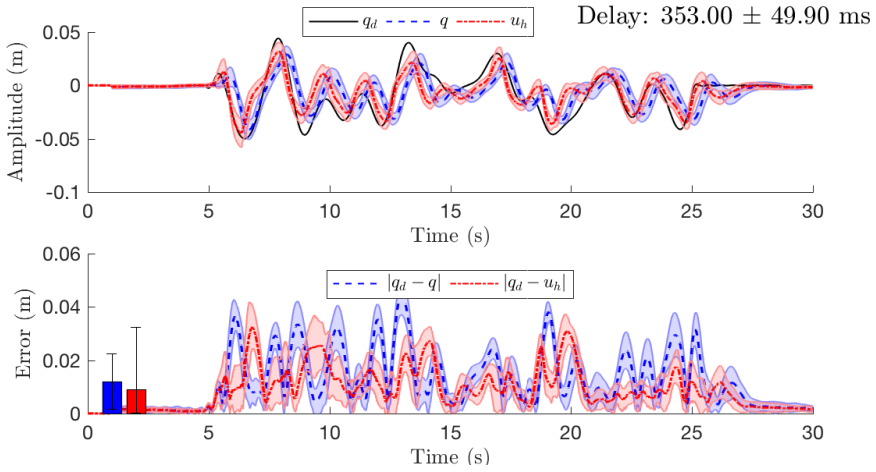
A total of $N = 10$ cycles of data was collected continuously where the human operator was trying to track y_d (depicted holographically as a scrolling path with the current point indicated as a green circle) with the robot end-effector (system output y). Fig. 5.1c depicts



(a) Reference trajectory y_d — training data



(b) Holographic experimental setup



(c) Experimental data: training phase.

Figure 5.1: Experimental setup to obtain training data for impulse response estimation with multiple human subjects: (a) reference trajectory y_d, (b) output-tracking task where human operator is shown the intended reference trajectory y_d through an Augmented Reality display, (c) Training data (r_h, y, y_d) in time-domain.

the experimental data from the training phase, where we can see that there is a phase lag in the controlled system output y compared to the human input r_h due to the internal velocity control loop of the robot.

The data $y(t), y_d(t), r_h(t)$ is collected for the task with (a) 1 s goal-preview displayed to the human operator, and (b) no goal-preview displayed to the human operator, which forms the training data that is used to estimate the inverse impulse response estimates for each case. Finally, the learned inverse impulse response estimates are used to estimate intent on a new human-in-the-loop output tracking task given the human demonstrations $y(t)$ considering the four cases:

1. slow reference trajectory $y_{d,1}$ (with frequency bandwidth $f_{BW} = 0.5$ Hz) and 1 s goal-preview displayed

2. slow reference trajectory $y_{d,1}$ (with frequency bandwidth $f_{BW} = 0.5$ Hz) and 0 s goal-preview displayed

3. fast reference trajectory $y_{d,2}$ (with frequency bandwidth $f_{BW} = 1$ Hz) and 1 s goal-preview displayed

4. fast reference trajectory $y_{d,2}$ (with frequency bandwidth $f_{BW} = 1$ Hz) and 0 s goal-preview displayed

The test reference trajectories $y_{d,1}$ and $y_{d,2}$ are chosen similar to the training trajectory but with randomized amplitudes and phases.

5.3.2 Comparison of time-domain and frequency-domain regression methods

This section addresses research problem (RQ1), specifically that the impulse response estimates obtained directly by regression in the time-domain are comparable to the impulse response obtained by transforming the frequency-domain regression models into time-domain.

First, consider the naive de-convolution approach described in Section 5.2.1, where it can be seen that it depends on a set of parameters, namely the buffer lengths N_1, N_2 and the buffer sampling rate T_s. A detailed analysis of the empirical effect of these parameters is carried out on the intent estimation performance of the computed inverse impulse response estimates, as shown in Figs. 5.2,5.3,5.4 which compare the MAX/RMS intent estimation error for various pairs of buffer lengths and buffer sampling rates. *Overall, it is seen that with higher buffer length and higher sampling rate (or smaller sampling interval) the intent estimation error decreases, as expected.*

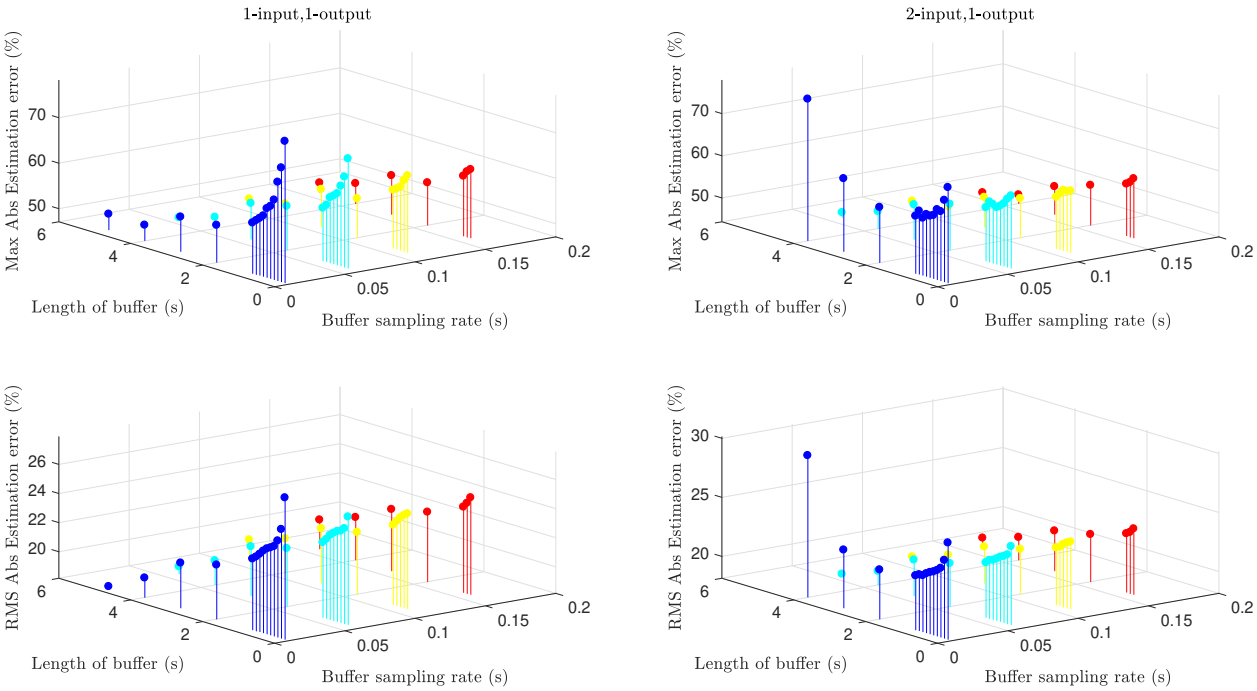


Figure 5.2: Naive Discrete-time De-convolution Results: Combined waterfall analysis of effect of buffer length and sampling rate on MAX/RMS intent estimation error.

Fig. 5.5 compares the naive-deconvolution approach in time-domain with the impulse response estimates obtained by transforming the CGPR frequency response estimates to

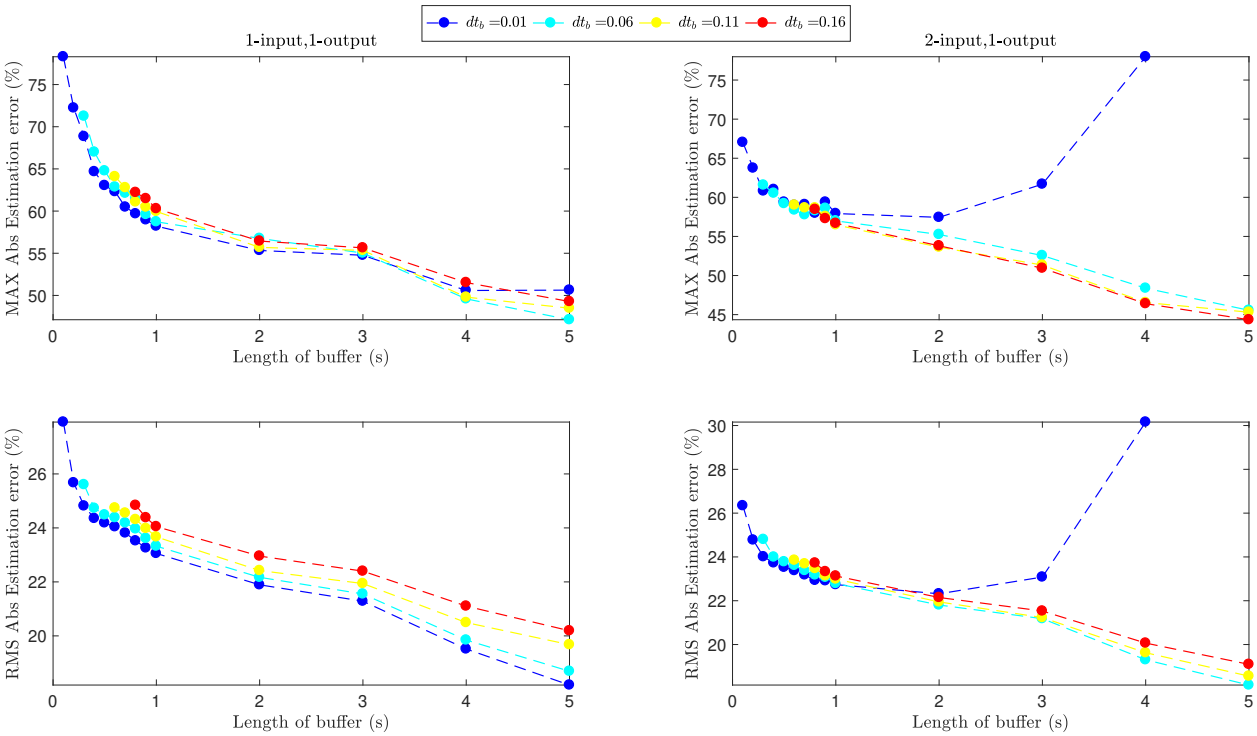


Figure 5.3: Naive Discrete-time De-convolution Results: Effect of buffer length on MAX/RMS intent estimation error.

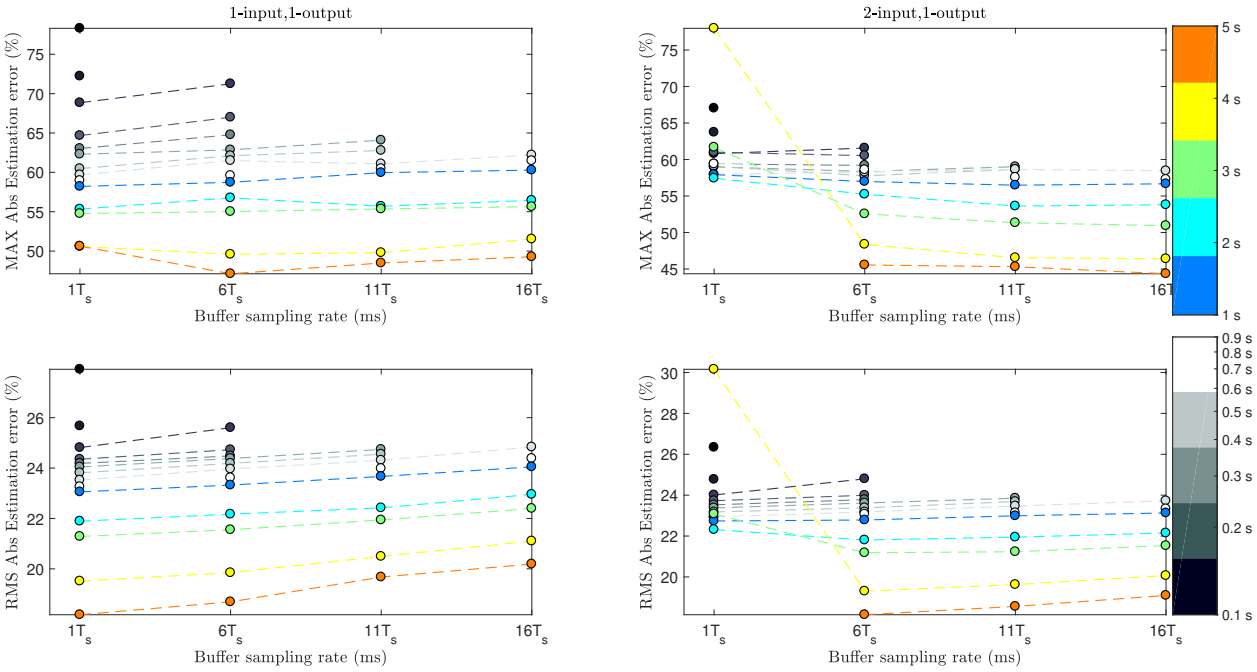
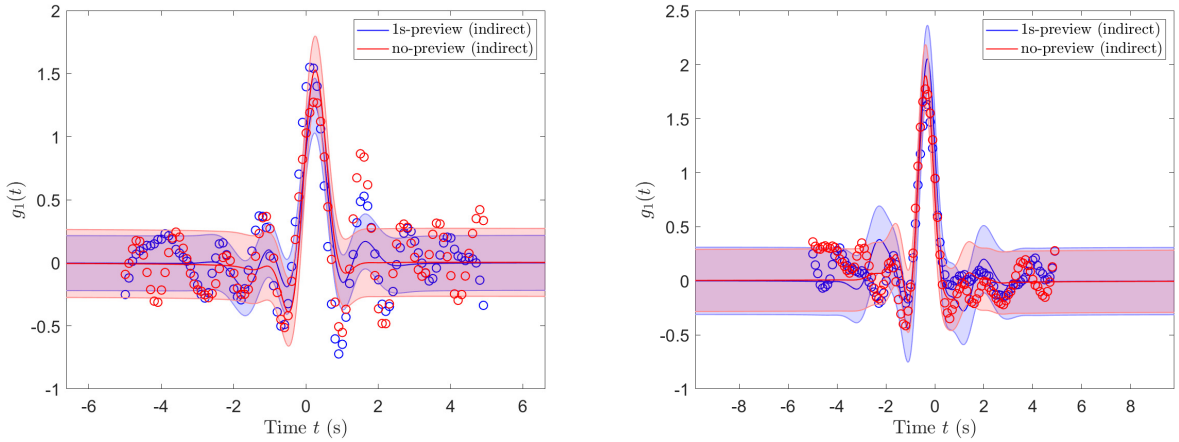


Figure 5.4: Naive Discrete-time De-convolution Results: Effect of Buffer sampling rate on MAX/RMS intent estimation error.



(a) $g(t) = \mathcal{F}^{-1}(\hat{G}_{y_d \to y}(\omega))$ (b) $g_{inv}(t) = \mathcal{F}^{-1}(\hat{G}_{y \to y_d}(\omega))$

Figure 5.5: Forward and inverse impulse response models obtained by transforming the non-parametric frequency response models obtained using the CGPR method described in Section 4.4.4 (shown as solid lines with shaded region depicting 2 standard deviations from the mean estimate), and the circular markers denote the time-domain estimate obtained using the least-squares method described in Section 5.2.1. Note that the least squares approach leads to a noisier estimate compared to the regularized estimate obtained using the squared exponential kernel of the CGPR method.

time-domain. It is seen that the estimates are fairly close but, the naive de-convolution approach is significantly noisier compared to the CGPR method where the regularization using the squared exponential kernel in the CGPR approach mitigates the effect of noise in the training data.

5.3.3 Effect of preview-time on intent estimation

The effect of preview-time on the intent estimation performance is studied using the following performance metrics: the maximum absolute tracking error, e_{max} and the root mean square error e_{rms}, each defined in time-domain as,

$$e_{max} = \frac{\max_{t_i \in [0,T]} |y_d(t_i) - y(t_i)|}{A} \times 100\%, \tag{5.7}$$

$$e_{rms} = \frac{\left(\frac{1}{N}\sum_{i=1}^{N} |y_d(t_i) - y(t_i)|\right)^{1/2}}{A} \times 100\%, \tag{5.8}$$

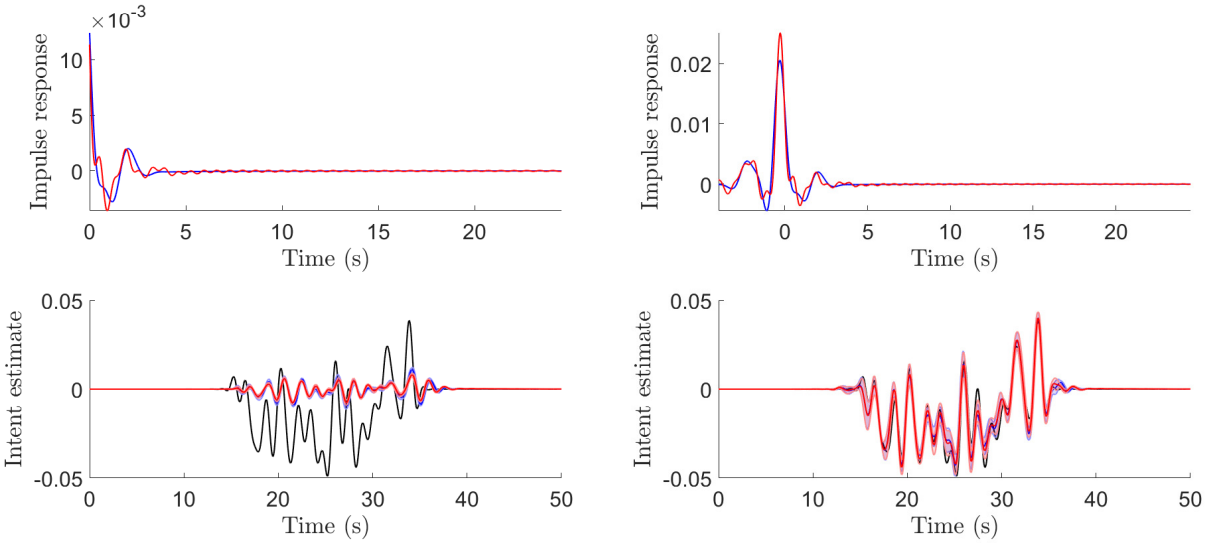
and, the corresponding intent estimation error metrics as,

$$\hat{e}_{max} = \frac{\max_{t_i \in [0,T]} |y_d(t_i) - \hat{y}_d(t_i)|}{A} \times 100\%, \tag{5.9}$$

$$\hat{e}_{rms} = \frac{\left(\frac{1}{N}\sum_{i=1}^{N} |y_d(t_i) - \hat{y}_d(t_i)|\right)^{1/2}}{A} \times 100\%, \tag{5.10}$$

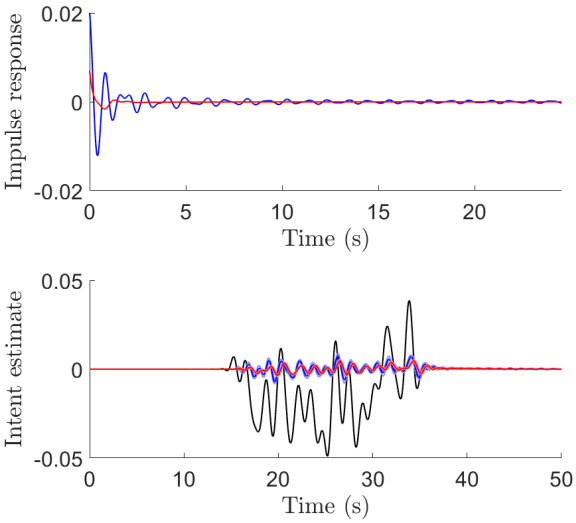
where, T is the time-period of each trial in seconds, N is the number of samples collected during each trial, and A is the amplitude of the desired output y_d.

Fig. 5.6 depicts the estimated impulse response using the CGPR method with (a) no preview-time (or pre-actuation time), i.e., considering a causal impulse response, and (b) 4 seconds of preview-time (or pre-actuation time), i.e., considering a non-causal impulse response. It is seen that the estimated intent $\hat{y}_d$ (SISO estimate shown in blue, and MISO estimate shown in red) matches the intended goal trajectory y_d (shown in black) more closely with a larger amount of pre-actuation time.

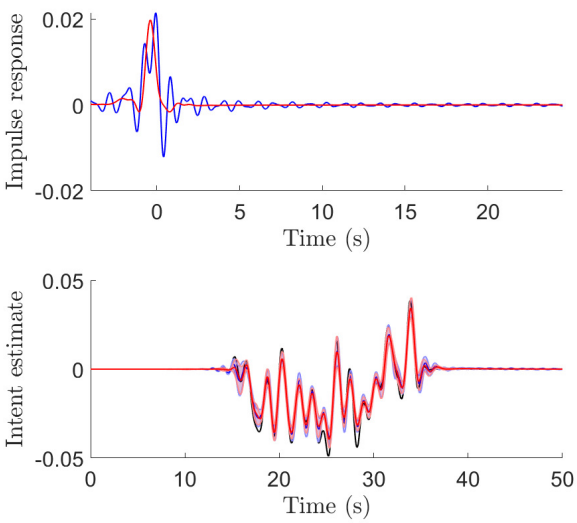


(a) 1 s Goal-Preview, $T_{pre} = 0$ s

(b) 1 s Goal-Preview, $T_{pre} = 4$ s



(c) 0 s Goal-Preview, $T_{pre} = 0$ s



(d) 0 s Goal-Preview, $T_{pre} = 4$ s

Figure 5.6: Inverse impulse response estimate obtained using the CGPR method, as described in Section 5.2.2 (top plots) with the estimated intent $\hat{y}_d$ (SISO estimate shown in blue, and MISO estimate shown in red) compared to the intended goal y_d (shown in black) for Subject 1 with (a) 1 s goal-preview displayed to Subject and preview-time $T_{pre} = 0$ s used in the inversion, (b) 1 s goal-preview displayed to Subject and preview-time $T_{pre} = 4$ s used in the inversion, (c) 0 s goal-preview displayed to Subject and preview-time $T_{pre} = 0$ s used in the inversion, (d) 0 s goal-preview displayed to Subject and preview-time $T_{pre} = 4$ s used in the inversion. The shaded region depicts two standard deviations from the mean estimates.

(RQ2) Amount of preview-time for best possible intent estimation performance

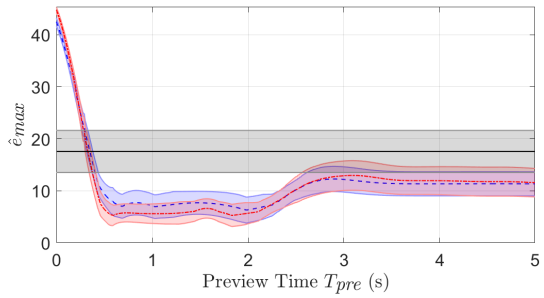
To validate this observation, the intent estimation was repeated with varying amounts of preview-time and the resulting intent estimation error metrics were plotted as a function of the preview-time, e.g., as shown in Figs. 5.7 and 5.8 for Subject 1. It is seen that in each case, there is a specific preview-time that results in the best intent estimation performance, for example when no goal-preview is displayed to the human operator, the required preview-time $T_{pre,best} = 3.10$ s results in the lowest intent estimation error. Similar observations are made for the multiple test subjects and are tabulated in Table 5.9.

(RQ3) Minimum preview-time for better intent estimation than human demonstration

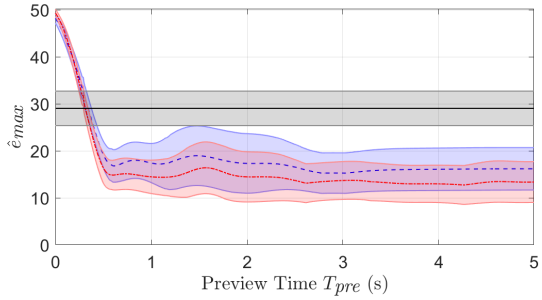
Additionally, it is also seen that there is a minimum preview-time $T_{pre,min}$ such that the intent estimation performance is better than the human demonstrated output in tracking the intended goal-trajectory, for example, Fig. 5.7d indicates that for Subject 1 with no-goal preview displayed, the required minimum preview-time $T_{pre,min} = 0.22$ s. Similar observations are made for the multiple test subjects and are tabulated in Table 5.9.

5.3.4 Conclusions

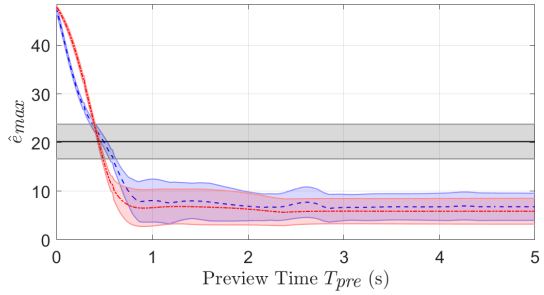
In conclusion, it is seen that the time-domain intent estimation method using learned inverse impulse response estimates is dependent on the amount of preview-information that is available to perform the inversion. Further it was shown that not only was it important to consider non-causal impulse response estimates for inverting the human-in-the-loop dynamics for successful intent estimation, but it was also shown that the amount of preview required is also subject-specific, i.e., it depends on the training data. Specifically, there exists a minimum amount of preview-time required for the intent estimation using the learned inverse impulse response estimates that guarantees that the resulting intent estimate better tracks the intended goal as compared to the human demonstrations. Practically, implementation



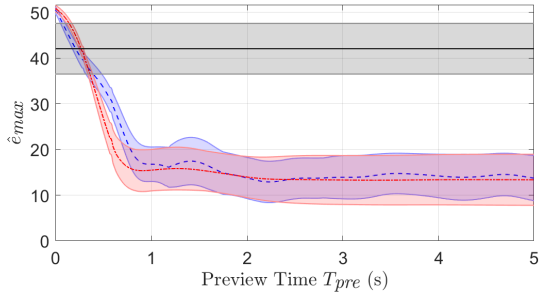
(a) 1 s Goal-Preview, $y_{d,1}$ (slow)



(b) 1 s Goal-Preview, $y_{d,2}$ (fast)



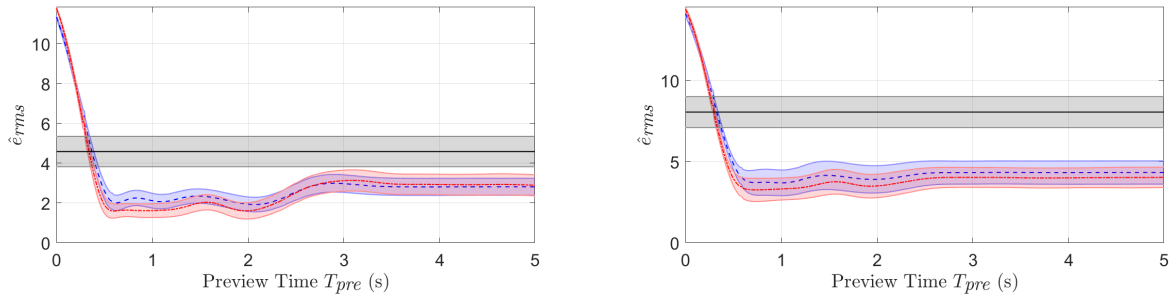
(c) 0 s Goal-Preview, $y_{d,1}$ (slow)



(d) 0 s Goal-Preview, $y_{d,2}$ (fast)

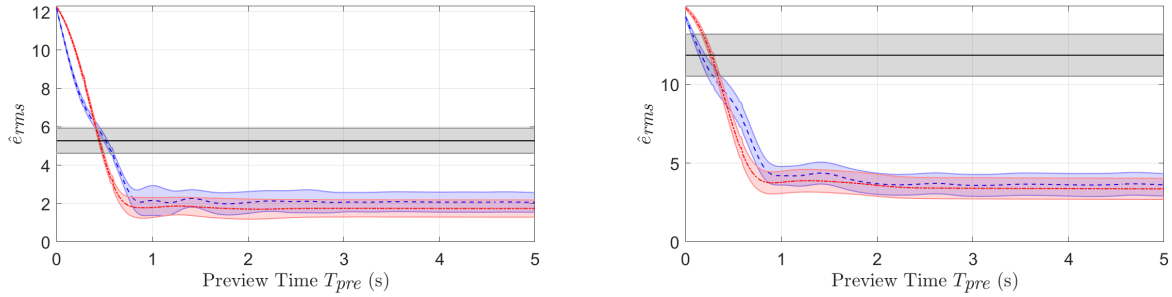
Figure 5.7: Maximum absolute intent estimation error $\hat{e}_{max}$, defined in (5.9) as a function of the preview-time T_{pre} for Subject 1 with (a) slow reference trajectory $y_{d,1}$, and (b) fast reference trajectory $y_{d,2}$ both with 1 s goal-preview displayed to the human operator, and (c) slow reference trajectory $y_{d,1}$, and (d) fast reference trajectory $y_{d,2}$ both with 0 s goal-preview displayed to the human operator. In each case, the black line depicts the average human demonstrated error $(y_d - y)$, the blue line depicts the average SISO intent estimation error, $(y_d - \hat{y}_{d,1})$, and the red line depicts the average MISO intent estimation error $(y_d - \hat{y}_{d,2})$, with the shaded region depicting one standard deviation about the mean. It is seen that in each case the intent estimation error reduced with larger preview-time T_{pre} as expected.

.



(a) 1 s Goal-Preview, $y_{d,1}$ (slow)

(b) 1 s Goal-Preview, $y_{d,2}$ (fast)



(c) 0 s Goal-Preview, $y_{d,1}$ (slow)

(d) 0 s Goal-Preview, $y_{d,2}$ (fast)

Figure 5.8: Root-Mean-Square (RMS) intent estimation error $\hat{e}_{rms}$, as defined in (5.10), as a function of the preview-time T_{pre} for Subject 1 with (a) slow reference trajectory $y_{d,1}$, and (b) fast reference trajectory $y_{d,2}$ both with 1 s goal-preview displayed to the human operator, and (c) slow reference trajectory $y_{d,1}$, and (d) fast reference trajectory $y_{d,2}$ both with 0 s goal-preview displayed to the human operator. In each case, the black line depicts the average human demonstrated error $(y_d - y)$, the blue line depicts the average SISO intent estimation error, $(y_d - \hat{y}_{d,1})$, and the red line depicts the average MISO intent estimation error $(y_d - \hat{y}_{d,2})$, with the shaded region depicting one standard deviation about the mean. It is seen that in each case the intent estimation error reduced with larger preview-time T_{pre} as expected.

.

of this inversion method online for real-time intent estimation would require the addition of a time-delay in the loop equivalent to the required amount of preview-time as computed during the training phase (offline). This implies that the designer of an online controller based on this estimated intent must optimize the choice of the preview-time that maximizes intent estimation accuracy while ensuring that the resultant time-delay introduced in the control loop is within the acceptable levels for human-in-the-loop control, as discussed in Section 2.2.2.

Figure 5.9: Maximum absolute intent estimation error e_{max}[1] on test trajectories $y_{d,test\text{-}1}, y_{d,test\text{-}2}$ using SISO/MISO trained inverse impulse response models.

Desired output		$y_{d,test\text{-}1}$ (slow)					$y_{d,test\text{-}2}$ (fast)				
Goal pre-view	Sub	e_{max}	$\hat{e}_{max,1}$ ($T_{pre,best}$)	$\hat{e}_{max,2}$ ($T_{pre,best}$)	$T_{pre,min}$ (SISO)	$T_{pre,min}$ (MISO)	e_{max}	$\hat{e}_{max,1}$ ($T_{pre,best}$)	$\hat{e}_{max,2}$ ($T_{pre,best}$)	$T_{pre,min}$ (SISO)	$T_{pre,min}$ (MISO)
1 s	1	17.57 ± 4.04	6.64 ± 2.42 (2.09)	5.27 ± 2.13 (1.83)	0.38	0.34	29.07 ± 3.67	15.27 ± 4.31 (2.79)	12.74 ± 4.14 (4.27)	0.35	0.32
	2	35.86 ± 6.02	33.14 ± 1.60 (5.00)	24.45 ± 3.25 (5.00)	3.40	1.21	46.03 ± 2.67	41.64 ± 3.57 (5.00)	38.36 ± 6.16 (3.35)	1.41	0.91
	3	19.94 ± 4.40	13.92 ± 3.26 (3.41)	18.47 ± 3.49 (3.07)	0.94	2.95	43.63 ± 3.44	37.36 ± 3.19 (1.05)	36.52 ± 3.49 (4.32)	0.13	0.81
	4	27.21 ± 6.94	21.48 ± 2.46 (4.95)	19.25 ± 4.31 (2.88)	0.63	0.87	42.60 ± 3.74	28.27 ± 2.99 (4.95)	24.40 ± 2.27 (3.20)	0.40	0.35
	5	32.60 ± 3.92	22.26 ± 6.58 (5.00)	20.56 ± 7.39 (5.00)	0.63	0.64	46.92 ± 4.86	27.12 ± 6.55 (5.00)	26.25 ± 5.55 (5.00)	0.43	0.49
	6	25.86 ± 4.78	19.57 ± 7.90 (5.00)	17.90 ± 10.25 (0.71)	0.97	0.50	40.60 ± 2.68	35.14 ± 4.13 (5.00)	20.54 ± 6.84 (0.71)	0.76	0.36
	7	20.07 ± 6.89	13.22 ± 7.67 (2.51)	8.04 ± 11.18 (0.89)	0.86	0.63	48.44 ± 3.93	27.49 ± 3.79 (1.24)	24.93 ± 4.99 (1.07)	0.45	0.45
	8	23.39 ± 7.53	20.97 ± 5.66 (1.04)	21.17 ± 5.88 (1.07)	0.86	0.89	50.31 ± 5.85	45.57 ± 4.59 (5.00)	47.76 ± 4.48 (5.00)	0.16	0.12
	overall	25.31 ± 6.42	18.86 ± 7.94 (3.63 ± 1.59)	16.83 ± 6.69 (2.56 ± 1.74)	1.08 ± 0.96	1.00 ± 0.83	43.45 ± 6.62	30.21 ± 9.40 (3.75 ± 1.78)	28.14 ± 10.98 (3.37 ± 1.67)	0.51 ± 0.41	0.48 ± 0.26
0 s	1	20.21 ± 3.57	6.48 ± 2.84 (2.85)	5.66 ± 2.75 (2.38)	0.52	0.45	42.07 ± 5.55	13.92 ± 4.76 (3.10)	13.37 ± 5.59 (5.00)	0.22	0.27
	2	35.98 ± 3.31	26.47 ± 4.43 (3.54)	25.61 ± 4.18 (3.89)	0.97	0.87	47.08 ± 2.84	36.45 ± 4.67 (3.18)	32.62 ± 6.01 (3.14)	0.40	0.55
	3	23.17 ± 3.34	20.37 ± 3.55 (0.33)	21.73 ± 1.50 (5.00)	0.19	2.58	50.84 ± 4.03	48.41 ± 3.70 (0.17)	35.64 ± 5.78 (2.42)	0.05	0.44
	4	24.06 ± 5.50	21.35 ± 5.84 (0.33)	19.98 ± 5.07 (1.05)	0.19	0.79	45.20 ± 2.45	44.80 ± 2.62 (0.33)	37.47 ± 3.83 (1.58)	0.12	0.10
	5	23.50 ± 4.35	20.73 ± 4.50 (0.33)	20.73 ± 4.50 (0.33)	0.19	0.19	43.42 ± 5.98	43.84 ± 6.84 (0.33)	43.84 ± 6.84 (0.33)	0.14	0.14
	6	25.68 ± 5.62	12.62 ± 3.11 (1.19)	15.16 ± 5.26 (3.12)	0.78	0.91	43.86 ± 5.36	32.42 ± 5.79 (1.43)	33.10 ± 5.34 (1.42)	0.57	0.67
	7	25.63 ± 11.84	13.44 ± 13.44 (5.00)	14.37 ± 12.54 (2.02)	0.55	0.57	44.67 ± 9.91	29.98 ± 12.36 (1.11)	28.12 ± 12.53 (4.53)	0.40	0.27
	8	24.25 ± 1.81	23.03 ± 1.42 (5.00)	19.99 ± 1.35 (5.00)	1.67	1.14	45.34 ± 6.86	37.57 ± 4.06 (5.00)	37.04 ± 4.20 (5.00)	0.62	0.60
	overall	25.31 ± 4.64	18.06 ± 6.59 (2.32 ± 2.05)	17.87 ± 6.13 (2.85 ± 1.73)	0.63 ± 0.51	0.94 ± 0.73	45.31 ± 2.68	33.76 ± 10.37 (1.83 ± 1.75)	30.71 ± 8.58 (2.93 ± 1.78)	0.32 ± 0.21	0.38 ± 0.22

[1] Values reported as $[\mu_x \pm \sigma_x] \times 100\%$, where $\mu_x = \sum_{i=1}^{n} x_i/n$ is the mean, and $\sigma_x = \sqrt{\sum_{i=1}^{n} x_i^2/(n-1)}$ is one standard deviation for $n = 10$ test trials, for each performance metric, x, and $\%E_{red}$ is defined in (4.105).

Chapter 6

CONCLUSIONS AND FUTURE WORK

This thesis described a model-inversion based approach to improve robot learning from demonstrations by estimating the unknown intent of human-in-the-loop demonstrations. Broadly, the imperfections in the demonstrations are postulated to be the effect of the human-in-the-loop dynamics that causes a deviation between the human intent and demonstrations, and so the approach presented in this thesis is to learn and compensate these human-in-the-loop dynamics to recover the underlying intent from the observed human demonstrations.

Conventionally, learning from demonstration assumes that the human demonstrator is an expert at the task, and so the task representation directly emulates this supposedly expert behavior. But, this thesis relaxed this assumption since it was shown that human intentions and demonstrations deviate when the human-in-the-loop dynamics become significant, as described in Section 2.1. In this regard, various human operator modeling approaches were explored in Section 2.2 to find an appropriate model that could be inverted to obtain the human intent from the human actions. Specifically, the empirical crossover model was described in detail as a simple parametric model that could be inverted to obtain the human intent from demonstrations under restricted task conditions — e.g., simple types of controlled systems and limited loop delays. Additionally, various parametric/non-parametric system identification techniques were explored that could learn the forward/inverse human-in-the-loop dynamics from labeled training data.

With this background, the main contributions of this thesis are as follows:

(C1) Stable inversion of simple human-in-the-loop dynamics models to infer intent.

Chapter 3 addressed this research problem through an iterative learning approach, where the human operator iterative guides the learning of the exact inverse control input that results in perfect tracking of the unknown human intent. Specifically, nominal human models were shown to be invertible to obtain human intent even when such models were not available for general linear controlled systems. This was done by modifying the apparent controlled system perceived by the human operator using an online inverse controller, as shown in Section 3.4.

(C2) Iterative convergence guarantees under modeling uncertainties and noise

Chapter 3 provided detailed convergence analyses in the presence of modeling uncertainty and output noise, considering the most general structure for the human-in-the-loop dynamics.

- Specifically, the rate of convergence of the frequency-domain iterative controller is bounded above by the size of the modeling uncertainty point-wise at each frequency, as shown in Section 3.2.2.

- Conditions also included robustness guarantees against output noise, as shown in Section 3.2.2.

- (C1+C2) Experimental validation of the robust iterative controller with multiple human subjects in teaching a robot controller the exact inverse input required to track an unknown intended goal trajectory.

  - Results using the empirical crossover model to represent the human-in-the-loop dynamics indicated that the stable inversion was restricted to low frequencies (typically about 0.2-0.3 Hz) owing to the larger modeling uncertainties at higher frequencies, see Sections 3.3 and 3.4.2 with the online inverse controller.

  - Data-based non-parametric approaches were shown to result in larger bandwidth owing to lower model uncertainties at higher frequencies, see Section 3.5.

(C3) Robust inversion-based intent estimation without iterations

Chapter 4 discussed the conditions on the modeling uncertainty at each frequency such that directly inverting the given human-in-the-loop dynamics model will result in improved tracking of the unknown human intent compared to the average human-demonstrations.

- Specifically, the conditions indicate that the regions of acceptable uncertainty around the mean estimate of a given model depends on the relative location of the mean model estimate in the complex plane for each frequency— a closed disc for models with mean estimates of magnitude less than one, a half-plane excluding 1 for models with mean estimates with magnitude one, and a plane excluding an open disc around 1 for models with mean estimates of magnitude greater than one, as shown in Section 4.4.1.

- Given the mean and co-variance estimates of a human-in-the-loop dynamics model, Lemma 2 described the conditions that determine whether inverting the given model results in reduced intent estimation compared to the human demonstrations. In summary, the approach considers whether the given uncertainty estimates lie inside the acceptable regions of uncertainty as prescribed by Lemma 1 in Section 4.4.1, and to invert the model only at those frequencies where the model uncertainties lie within the acceptable regions of uncertainty.

- It was also shown that given training data including the human input r_h and the human demonstrated output y, a multiple-input-single-output (MISO) intent estimation model resulted in better overall intent estimation performance (on a new but similar reference trajectory) compared to a SISO intent estimation model with only the human demonstrated output y as the input, as shown in Section 4.5.

- Experimental validation with multiple human subjects indicated that overall, the proposed robust inversion approach resulted in the intent estimation error to be greater than or equal to the tracking error by the human demonstrated output, as shown in Section 4.5.

Limitations

The major limitation of the proposed work relates to the feasibility of the learned intent estimator models which may be computed iteratively, as discussed in Chapter 3, or using the robust method proposed in Chapter 4. In both cases, the models are feasible for estimating an unknown intent in a new output tracking task if and only if the training data used to compute the models sufficiently describes the new task conditions. So, practically in order to safely use learned inverse models for generalizing the intent estimation to new tracking tasks, the practitioner is cautioned to verify that the task conditions are similar to those present during the training phase used to estimate the inverse models.

Further, Chapters 3 and 4 described the intent estimation in the frequency-domain, which is sufficient if the intent estimation is to be done batch-wise. In contrast, in cases where intent estimation must be done sequentially or online, frequency-domain methods are not readily applicable, often resulting in injecting delays in the control loop to enable collection of enough batch-wise data to transform to the frequency-domain. Thus, directly estimating the human intent in the time-domain is important when trying to estimate and use the human intent online for developing online control inputs.

In this regard, Chapter 5 presents the challenges in estimating the human-in-the-loop intent in real-time. Specifically,

- The human-in-the-loop inverse impulse response was observed to include significant amount of non-causality — approximately 2-4 seconds of preview-time is required to accurately invert the human-in-the-loop dynamics.

- This is problematic since it was shown in Section 2.2.2 that the human-in-the-loop tracking performance is adversely affected by introduction of time-delays in the control loop — maximum allowable time-delay is about 0.5 seconds.

Consequently, the accuracy of real-time intent estimation using this model-inversion based approach is limited by the amount of preview-information that is available at a particular time instant for a given task.

Future Work

In this regard, the following are some possible direction for future work:

- Characterize the benefit of using the real-time estimated intent compared to the observed demonstration in predicting intended motion primitives, e.g., is there a benefit to using the estimated intent in classifying among a set of sub-tasks of an overall pick-and-place task compared to directly using the observed demonstrations for classification?

- Extend the non-parametric regression to learn the non-causal human-in-the-loop inverse dynamics models to time-domain. This will help in directly estimating a functional approximation in the time-domain as opposed to functional approximation in the frequency-domain transformed to time-domain using FFT, as described in Chapter 5 which might result in inaccuracies due to the approximate nature of the FFT algorithm.

Appendix A

CODE DOCUMENTATION

Software License

A.1 Kinova Mico2 Cartesian Controller

A.2 Introduction

This project is a C++ implementation of a Cartesian controller that provides basic human-machine interface modules for real-time end-effector Cartesian position/velocity control of the Kinova Mico2 robot arm using the built-in velocity controllers in the Kinova API. The project is written in C++ and is intended to be run on Windows 8.1/10 with Visual Studio 2017 (v15). The code is distributed under the MIT license for maximum flexibility of use. By using this software, you are agreeing to the license. Please read the license prior to using this project.

This introduction is broken down into the following sections.

- Software License
- Installation Notes

A note on units: All units in the library, unless specified, are in SI (international standard), i.e.: radians, meters, kilograms, etc...

A.3 Installation Notes

This project was built in Visual Studio 2017 (Community Edition) and the solution file is provided for reference. Additional dependencies need to be installed before building this project with the source and library files from the dependencies correctly linked to the project. Instructions for setting up the project and building it are presented below.

A.4 Install Dependencies

The following are the list of dependencies that are used in this project:

---

[0]Code is available at `https://jaguar243.github.io/kinova-cartesian-controller/`

- Kinova Mico2 SDK [to control the Kinova Mico-2 robot arm]
- Kinect SDK v1.8 [to use the Microsoft Kinect sensor for skeleton tracking]
- CHAI3D (version 3.2.0) [to use the Novint Falcon Haptics controller]

A.4.1 Kinova Mico2 SDK

1. Download the SDK from the `Kinova website` and follow the instructions in the documentation to install the 32-bit version of the API.
2. Set the System Environment Variable KINOVASDK_DIR to the installation directory.

A.4.2 Kinect SDK v1.8

1. Download and install the Kinect v1.8 SDK from the `Microsoft website` and follow the instructions in the documentation to install the 32-but version of the API.
2. Check whether the system environment variable KINECTSDK10_DIR is set to the correct installation directory.

A.4.3 CHAI3D (version 3.2.0)

1. Download the multiplatform version of the CHAI3D SDK (currently tested with version 3.2.0) from the `CHAI3D website`
2. Install the dependencies (HDF5 and ZLIB) and build the appropriate Visual Studio Solution that matches the available edition.
3. Set the system environment variable CHAI3D_DIR to the installation directory

A.5 Class Structure

A.5.1 Experiment Class Reference

`#include <CartesianControl.h>`

Public Member Functions

- void HoloLensCartesianTeleop (char *argv[], KinovaAPIFunctions kinova)
- double interp_lut (double t, double *y_lut, int L_lut, double Tp_lut, double fs_lut)
- void MovetoStartPos (KinovaAPIFunctions kinova)
- void MoveEndEffectorPos (KinovaAPIFunctions kinova, float xe, float ze)
- bool load_LUT1D (char *filename, char delim)

Public Attributes

- double T = 10.0
- double T_calib = 10.0
- double T_TF = 10.0
- double Ts = 1.0 / 125.0
- const double fs_qd_lut = 3000.0
- const double fs_uc_lut = 3000.0
- double scale_LUT = 1
- double * LUT
- int len_LUT
- const float Kp = 2.0f

Member Function Documentation

void Experiment::HoloLensCartesianTeleop (char argv[], **KinovaAPIFunctions** kinova)*

double Experiment::interp_lut (double t, *double** y_lut, *int* L_lut, *double* Tp_lut, *double* fs_lut)*

This function linearly interpolates the values in the Lookup table for a given input time instant.

Inputs:

1. (double) t : current time instant (s)
2. (double∗) ylut : Lookup Table
3. (int) L_lut : Length of the Lookup table
4. (double) Tp_lut : Time period of Lookup table (s)
5. (double) fs_lut : Sampling frequency of Lookupt table (Hz)

Output:

1. (double) : output of interpolation

bool Experiment::load_LUT1D (*char* ∗ filename, *char* delim *)*

Load Lookup table from CSV file

Inputs:

1. (char∗) filename: Filename of CSV file
2. (char) delim : Delimiter for CSV file

Output:

1. (bool) : Status of LUT load operation

void Experiment::MoveEndEffectorPos (**KinovaAPIFunctions** kinova, *float* xe, *float* ze *)*

Move End-Effector to Desired Position (Angular Control) [NOT USED IN EXPERIMENT] INPUTS:

1. (KinovaAPIFunctions) kinova : A class object that can access the Kinova API to control/monitor the robot arm
2. (float) xe : Desired Cartesian end-effector position in the X-direction
3. (float) ze : Desired Cartesian end-effector position in the Z-direction

void Experiment::MovetoStartPos (**KinovaAPIFunctions** kinova *)*

Move the Robot Arm to the Start Position by setting the angular position of each joint: [270, 215, 90, 180] degrees

Member Data Documentation

const double Experiment::fs_qd_lut = 3000.0

const double Experiment::fs_uc_lut = 3000.0

const float Experiment::Kp = 2.0f

int Experiment::len_LUT

*double * Experiment::LUT*

double Experiment::scale_LUT = 1

double Experiment::T = 10.0

double Experiment::T_calib = 10.0

double Experiment::T_TF = 10.0

double Experiment::Ts = 1.0 / 125.0

The documentation for this class was generated from the following files:

- CartesianControl.h
- angularCommandControl.cpp
- expt_HoloLensCartesianTeleop.cpp
- KinovaHMICartesianControl/lutLinearInterp.cpp

A.5.2 FIRFilter Class Reference

`#include <CartesianControl.h>`

Public Member Functions

- bool firFloatInit ()

  Function to initialize the FIR Filter.

- double * firFloat (double *input, int length)

  Function to compute the output of the FIR Filter.

- double * firFloat (double *input, int length)

  Function to compute the output of the FIR Filter.

Public Attributes

- const char * filename = "./lowpass.mat"

  Filename of FIR Filter weights.

- int length = 1
- double * output = new double[length]
- double insamp [BUFFER_LEN]

  buffer array to hold input samples

- double * coeffs

  FIR Filter weights.

- double cutoff_freq_hz = 0.2

  Cutoff frequency of the FIR Filter in Hz.

Member Function Documentation

*double * FIRFilter::firFloat (double * input, int length)*

Function to compute the output of the FIR Filter.

FIR Filter Computation

This function computes the output of the FIR Filter for the given content of the input buffer in three steps:

1) Put new input at the high end of the buffer

2) Apply the filter to each input sample

3) Shift input samples back in time for next time instant

bool FIRFilter::firFloatInit ( )

Function to initialize the FIR Filter.

FIR Filter Initialization

This function initializes the FIR filter by searching for a CSV file formatted as

num_weights

w1

w2

.

.

.

wn

which is read into the coeffs variable.

Member Data Documentation

*double * FIRFilter::coeffs*

FIR Filter weights.

double FIRFilter::cutoff_freq_hz = 0.2

Cutoff frequency of the FIR Filter in Hz.

*const char * FIRFilter::filename = "./lowpass.mat"*

Filename of FIR Filter weights.

double FIRFilter::insamp

buffer array to hold input samples

int FIRFilter::length = 1

*double * FIRFilter::output = new double[**length**]*

The documentation for this class was generated from the following files:

- CartesianControl.h
- firFilter.cpp

A.5.3 kinectSkelTrack Class Reference

`#include <CartesianControl.h>`

Classes

- struct KinectInfo

Public Member Functions

- bool initKinect ()
- void getSkeletalData ()
- KinectInfo getKinectData ()

Public Attributes

- int width = 640
- int height = 480
- HANDLE depthStream
- INuiSensor * sensor
- Vector4 skeletonPosition [NUI_SKELETON_POSITION_COUNT]

Member Function Documentation

void kinectSkelTrack::getSkeletalData ( )

Get the skeletal data from the current frame

bool kinectSkelTrack::initKinect ( )

Initialize the Kinect sensor for skeleton tracking of the user closest to the sensor.

A.5.4 Member Data Documentation

HANDLE kinectSkelTrack::depthStream

int kinectSkelTrack::height = 480

*INuiSensor * kinectSkelTrack::sensor*

Vector4 kinectSkelTrack::skeletonPosition

int kinectSkelTrack::width = 640

The documentation for this class was generated from the following files:

- CartesianControl.h
- HMIfunctions.cpp

A.6 Augmented Reality Rendering with Microsoft Hololens and Unity

This project was built using Unity 2017.3.0f and is available for download here.

A.6.1 Building and Deployment

1. Start the Unity Editor and import the folder `Unity` in the repository to open the project.

2. Build the project for the Hololens UWP target with the latest Windows SDK version installed on development machine.

3. Open the generated Visual Studio solution and deploy to the Hololens as a standalone UWP application.

4. To enable socket communication with the UWP application running on the Hololens, make sure that the IP address of the target in the Unity project is set to the required

IP of the host machine before building the Visual Studio solution in step 2. Also, make sure that the Hololens and the host machine are in the same network, i.e., their IP addresses are visible to each other.

A.7 Useful MATLAB scripts

A.7.1 Online Inverse Controller - FIR model

```
function inverse_weights_calc()
load sys_params.mat
err = sys_params.model_error;
xi_p1 = (1+err)*sys_params.Xi_p1; xi_p2 = (1+err)*sys_params.Xi_p2;
omN_p1 = (1+err)*sys_params.Omega_n_p1 * 2* pi; omN_p2 = ...
    (1+err)*sys_params.Omega_n_p2 * 2* pi;

omN_z1 = (1+err)*sys_params.Omega_n_z1 * 2* pi;
xi_z1 = (1+err)*sys_params.Xi_z1;

K = omN_p1^2*omN_p2^2/omN_z1^2;%(1+err)*sys_params.K;

s = tf('s');
Ts = 1e-2;

%% MINIMUM-PHASE WEIGHTS
% Post-view tme
Tp_s = 40*Ts
% system model
G_hat = K * (s^2+2*xi_z1*omN_z1*s+omN_z1^2) / ...
    ((s^2+2*xi_p1*omN_p1*s+omN_p1^2) * (s^2+2*xi_p2*omN_p2*s+omN_p2^2));

```

```
21 sys_model = ss(tf(G_hat));
22
23 % dcgain(sys_model)
24 % return
25
26 A = sys_model.A;
27 B = sys_model.B;
28 C = sys_model.C;
29 D = sys_model.D;
30
31 ns = floor(Tp_s/Ts);
32 if mod(ns,2)
33     ns = ns+1;
34 end
35 %------------------ Inverse input calculation ---------------------
36 Tt = [C;C*A];
37 Tb = [1 0 0 0;
38       0 1 1 0];
39 T = [Tt;Tb];
40 T_i = inv(T);
41 T_il = T_i(:,1:2);
42 T_ir = T_i(:,3:4);
43 By = C*A*B;
44 Ay = C*A*A;
45 % internal dynamics matrices
46 A_inv = Tb*(A - B/By*Ay)*T_ir;
47 B_inv = [Tb*(A - B/By*Ay)*T_il Tb*B/By];
48 C_inv = -By\Ay*T_ir;
49 D_inv = [-By\Ay*T_il By\eye(size(By))];
50
51 % Split the internal dynamics into stable and unstable parts
52 [T_split,D] = eigs(A_inv);
```

```
53 [T_split,~] = cdf2rdf(T_split,D);
54 lam = eigs(A_inv);
55 [~,ii] = sort(real(lam(:)),'descend');
56 T_split = T_split(:,ii);
57 A_split = T_split\A_inv*T_split;
58 B_split = T_split\B_inv;
59
60 % Offline calculation of weights
61 % Zero order hold integration
62 W = [];
63 As = A_split;
64 Bs = B_split;
65 Ms1 = expm(As*Ts);
66 Ms2 = As\(Ms1-eye(numel(lam(real(lam)<0))))*Bs;
67 for i=1:ns-1
68     W1 = (C_inv*T_split) * Ms1^(ns-i) * Ms2;
69     W = [W(:);W1(:)];
70 end
71 W2 = (C_inv*T_split) * (Ms1^(0) * Ms2) + D_inv;
72 W = [W(:);W2(:)];
73 save('weights_mp_zoh.mat','W');
74
75 %% NONMINIMUM-PHASE WEIGHTS
76 % Pre-view time
77 Tp_u = 4/(xi_z1*omN_z1)
78 % system model
79 G_hat = K * (s^2-2*xi_z1*omN_z1*s+omN_z1^2) / ...
       ((s^2+2*xi_p1*omN_p1*s+omN_p1^2) * (s^2+2*xi_p2*omN_p2*s+omN_p2^2));
80
81 sys_model = ss(tf(G_hat));
82
83 A = sys_model.A;
```

```
84  B = sys_model.B;
85  C = sys_model.C;
86  D = sys_model.D;
87
88  nu = floor(Tp_u/Ts);
89  if mod(nu,2)
90      nu = nu+1;
91  end
92  %------------------ Inverse input calculation ----------------------
93  Tt = [C;C*A];
94  Tb = [1 0 0 0;
95        0 1 1 0];
96  T = [Tt;Tb];
97  T_i = inv(T);
98  T_il = T_i(:,1:2);
99  T_ir = T_i(:,3:4);
100 By = C*A*B;
101 Ay = C*A*A;
102 % internal dynamics matrices
103 A_inv = Tb*(A - B/By*Ay)*T_ir;
104 B_inv = [Tb*(A - B/By*Ay)*T_il Tb*B/By];
105 C_inv = -By\Ay*T_ir;
106 D_inv = [-By\Ay*T_il By\eye(size(By))];
107
108 % Split the internal dynamics into stable and unstable parts
109 [T_split,D] = eigs(A_inv);
110 [T_split,~] = cdf2rdf(T_split,D);
111 lam = eigs(A_inv);
112 [~,ii] = sort(real(lam(:)),'descend');
113 T_split = T_split(:,ii);
114 A_split = T_split\A_inv*T_split;
115 B_split = T_split\B_inv;
```

```
116
117 % Offline calculation of weights
118 %% Zero order hold integration
119 W = [];
120 Au = A_split;
121 Bu = B_split;
122 Mu1 = expm(-Au*Ts);
123 Mu2 = Au\(Mu1-eye(numel(lam(real(lam)>0))))*Bu;
124
125 W1 = (C_inv*T_split) * (Mu1^(0) * Mu2) + D_inv;
126 W = [W(:);W1(:)];
127 for i=1:nu
128     W2 = (C_inv*T_split) * (Mu1^(i) * Mu2);
129     W = [W(:);W2(:)];
130 end
131 save('weights_nmp_zoh.mat','W');
132 %%
133 disp('Done');
```

A.7.2 Iteration Gain Selection according to ILC Convergence Conditions in Lemma 1

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Iteration Gain selection
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Function computes the feedback modeling error, Delta_fb for
% human-in-the-loop system with inverse control. This is used to inform the
% choice of the iteration gain to ensure convergence of the ILC algorithm.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function iter_gain_selection(err)
clc; close all;
WT = 2*pi*logspace(-1,0,1e3);
%% System model
s = tf('s');
K = 1;
taud = 0.25;
G = K*exp(-taud*s);
% System Params
omn1 = 0.3*(2*pi); omn2 = 0.5*(2*pi); omnz = 3.0*(2*pi);
xi1 = 0.1; xi2 = 0.1; xiz = 0.707;
KgainMP = (omn1*omn2/omnz)^2; KgainNMP = -(omn1*omn2/omnz)^2;
% Model params
omn1_hat = (1+err)*omn1; omn2_hat = (1+err)*omn2; omnz_hat = (1+err)*omnz;
xi1_hat = (1+err)*xi1; xi2_hat = (1+err)*xi2; xiz_hat = (1+err)*xiz;
KgainMP_hat = (omn1_hat*omn2_hat/omnz_hat)^2; KgainNMP_hat = ...
    -(omn1_hat*omn2_hat/omnz_hat)^2;
% System
Gmp = KgainMP * (s^2 + 2*xiz*omnz*s + omnz^2) /...
    ((s^2+2*xi1*omn1*s+omn1^2)*(s^2+2*xi2*omn2*s+omn2^2));
Gnmp = KgainNMP * (s^2 - 2*xiz*omnz*s + omnz^2) /...
```

```
    ((s^2+2*xi1*omn1*s+omn1^2)*(s^2+2*xi2*omn2*s+omn2^2));
% Model
Gmp_hat = KgainMP_hat * (s^2 + 2*xiz_hat*omnz_hat*s + omnz_hat^2) /...
    ((s^2+2*xi1_hat*omn1_hat*s+omn1_hat^2) * ...
        (s^2+2*xi2_hat*omn2_hat*s+omn2_hat^2));
Gnmp_hat = KgainNMP_hat * (s^2 - 2*xiz_hat*omnz_hat*s + omnz_hat^2) /...
    ((s^2+2*xi1_hat*omn1_hat*s+omn1_hat^2) * ...
        (s^2+2*xi2_hat*omn2_hat*s+omn2_hat^2));
% Apparent system
GA_mp = Gmp/Gmp_hat; GA_nmp = Gnmp/Gnmp_hat*exp(-taud*s);
%% Human model
Kp = 1;
tau_I = 0.2;
taue = 0.44;
GH = Kp*exp(-taue*s)/(tau_I*s+1); GH_hat = Kp*exp(-taue*s)/(tau_I*s+1);
%% Feedback freq response
% Actual
GFBmp = GH*GA_mp/(1+GH*GA_mp);
GFBnmp = GH*GA_nmp/(1+GH*GA_nmp);
[GFBmp_mag,GFBmp_phase] = bode(GFBmp,WT);
GFBmp_mag = reshape(GFBmp_mag,size(WT));
GFBmp_phase = reshape(GFBmp_phase,size(WT));
[GFBnmp_mag,GFBnmp_phase] = bode(GFBnmp,WT);
GFBnmp_mag = reshape(GFBnmp_mag,size(WT));
GFBnmp_phase = reshape(GFBnmp_phase,size(WT));
% Model
GFBmp_hat = GH_hat/(1+GH_hat);
GFBnmp_hat = GH_hat*G/(1+GH_hat*G);
[GFBmp_hat_mag,GFBmp_hat_phase] = bode(GFBmp_hat,WT);
GFBmp_hat_mag = reshape(GFBmp_hat_mag,size(WT));
GFBmp_hat_phase = reshape(GFBmp_hat_phase,size(WT));
[GFBnmp_hat_mag,GFBnmp_hat_phase] = bode(GFBnmp_hat,WT);
```

```
GFBnmp_hat_mag = reshape(GFBnmp_hat_mag,size(WT));
GFBnmp_hat_phase = reshape(GFBnmp_hat_phase,size(WT));
%% Modeling error
DeltaFBmp_mag = GFBmp_mag./GFBmp_hat_mag;
DeltaFBmp_phase = unwrap(GFBmp_phase) - unwrap(GFBmp_hat_phase);
DeltaFBnmp_mag = GFBnmp_mag./GFBnmp_hat_mag;
DeltaFBnmp_phase = unwrap(GFBnmp_phase) - unwrap(GFBnmp_hat_phase);
%% Maximum Iteration gain
rho_starmp = 2*cos(deg2rad(DeltaFBmp_phase))./(DeltaFBmp_mag);
rho_starnmp = 2*cos(deg2rad(DeltaFBnmp_phase))./(DeltaFBnmp_mag);
cutoffmp = find(rho_starmp <= 0,1);
cutoffnmp = find(rho_starnmp <= 0,1);
if ~isempty(cutoffmp)
    Cutoff_freqmp  = WT(cutoffmp)/(2*pi)
else
    Cutoff_freqmp  = 0.2
end
rho_starmp(cutoffmp+1:end) = 0*rho_starmp(cutoffmp+1:end);
if ~isempty(cutoffnmp)
    Cutoff_freqnmp  = WT(cutoffnmp)/(2*pi)
else
    Cutoff_freqnmp  = 0.2
end
rho_starnmp(cutoffnmp+1:end) = 0*rho_starnmp(cutoffnmp+1:end);
rho = 0.5*ones(size(WT));
rho(WT>=min([2*pi*Cutoff_freqnmp,2*pi*Cutoff_freqmp,2*pi*0.2])) = 0;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Plots
%%%%%%%%%%%
figure(1);
clf;
set(gcf,'PaperPosition',[0.01 0.01 9 6]);
```

```
90  set(gcf,'Name','Feedback frequency response');
91
92  % Magnitude, GFB_mag (dB)
93  subplot(211);
94  box on;
95  grid on;
96  set(gca,'FontSize',12);
97  hold on
98  plot(WT/(2*pi),mag2db(GFBmp_mag),'k');
99  plot(WT/(2*pi),mag2db(GFBmp_hat_mag),'k--');
100 h = plot(WT/(2*pi),mag2db(GFBnmp_mag),'r');
101 set(h,'Color',[0.5 0.5 0.5]);
102 h = plot(WT/(2*pi),mag2db(GFBnmp_hat_mag),'r--');
103 set(h,'Color',[0.5 0.5 0.5]);
104 ylim([-10 25]);
105 % xlim([WT(1)/(2*pi) 1]);
106 xlabel('Frequency, $f$ Hz','FontSize',14);
107 ylabel({'\makebox[4in][c]{Magnitude,}','\makebox[4in][c]{$|G_{fb}(f)|$ ...
        (dB)}'},'FontSize',14,...
108     'HorizontalAlignment','center','Interpreter','latex');
109 legend({'Actual, $G_{fb}$ (MP)','Model, $\hat{G}_{fb}$ (MP)',...
110         'Actual, $G_{fb}$ (NMP)','Model, $\hat{G}_{fb}$ (NMP)'},...
111     'FontSize',14,'Interpreter','latex','location','northoutside',...
112     'orientation','horizontal');
113 legend boxoff;
114 grid on;
115 % vline(Cutoff_freq,'k--');
116
117 % Phase, GFB_phase (deg)
118 subplot(212);
119 box on;
120 set(gca,'FontSize',12);
```

```
121 hold on
122 plot(WT/(2*pi),GFBmp_phase,'k');
123 plot(WT/(2*pi),GFBmp_hat_phase,'k--');
124 h = plot(WT/(2*pi),GFBnmp_phase,'r');
125 set(h,'Color',[0.5 0.5 0.5]);
126 h = plot(WT/(2*pi),GFBnmp_hat_phase,'r--');
127 set(h,'Color',[0.5 0.5 0.5]);
128 % xlim([WT(1)/(2*pi) 1]);
129 ylim([-360 5]);
130 xlabel('Frequency, $f$ Hz','FontSize',14);
131 ylabel({'\makebox[4in][c]{Phase,}',...
132     '\makebox[4in][c]{$\angle\;G_{fb}(f)$ (deg)}'},'FontSize',14,...
133     'HorizontalAlignment','center','Interpreter','latex');
134 grid on;
135 set(findall(gcf,'Type','Line'),'linewidth',1.5);
136
137 %% Modeling error and Maximum Iteration Gain
138 figure(2);
139 clf;
140 set(gcf,'PaperPosition',[0.01 0.01 8 8]);
141 set(gcf,'Name','Modeling error');
142
143 % Magnitude, DeltaFB_mag (dB)
144 subplot(311);
145 box on;
146 set(gca,'FontSize',12);
147 hold on
148 plot(WT/(2*pi),mag2db(DeltaFBmp_mag),'k');
149 plot(WT/(2*pi),mag2db(DeltaFBnmp_mag),'k-.');
150 % xlim([WT(1)/(2*pi) 1]);
151 % vline(Cutoff_freq,'k--');
152 % text(WT(cutoff)/(2*pi)+0.05,max(DeltaFB_mag)-2,...
```

```
153 %     {'Iteration gain cutoff frequency,',...
154 %     sprintf('$f_c=%2.1f$ Hz',WT(cutoff)/(2*pi))});
155 xlabel('Frequency, $f$ Hz','FontSize',14);
156 ylabel({'\makebox[4in][c]{Magnitude ...
       error,}','\makebox[4in][c]{$|\Delta_{fb}(f)|$ (dB)}'},'FontSize',14,...
157     'HorizontalAlignment','center','Interpreter','latex');
158 grid on;
159 legend({'Minimum-phase, $m=0\quad$','Nonminimum-phase, $m=1$'},...
160     'FontSize',14,'Interpreter','latex','location','northoutside',...
161     'orientation','horizontal');
162 legend boxoff;
163 text(0.95,15,'(a)')
164
165 % Phase, DeltaFB_phase (deg)
166 subplot(312);
167 box on;
168 set(gca,'FontSize',12);
169 hold on
170 plot(WT/(2*pi),DeltaFBmp_phase,'k');
171 plot(WT/(2*pi),DeltaFBnmp_phase,'k-.');
172 % xlim([WT(1)/(2*pi) 1]);
173 % ylim([-90 90]);
174 % set(gca,'YTick',[-90 -45 0 45 90]);
175 % vline(Cutoff_freq,'k--');
176 % text(WT(cutoff)/(2*pi)+0.05,min(DeltaFB_phase)+2,...
177 %     {'Iteration gain cutoff frequency,',...
178 %     sprintf('$f_c=%2.1f$ Hz',WT(cutoff)/(2*pi))});
179 xlabel('Frequency, $f$ Hz','FontSize',14);
180 ylabel({'\makebox[4in][c]{Phase error,}',...
181     '\makebox[4in][c]{$\angle\;\Delta_{fb}(f)$ (deg)}'},'FontSize',14,...
182     'HorizontalAlignment','center','Interpreter','latex');
183 grid on;
```

```
184 text(0.95,175,'(b)')
185
186 % Maximum Iteration gain, rho_star
187 subplot(313);
188 box on;
189 set(gca,'FontSize',12);
190 hold on
191 plot(WT/(2*pi),rho_starmp,'k');
192 plot(WT/(2*pi),rho_starnmp,'k-.');
193 plot(WT/(2*pi),rho,'k--');
194 plot(0.2,0.5,'ko','MarkerSize',7,'MarkerFaceColor','black')
195 plot(0.2,0,'ko','MarkerSize',7)
196 xlim([WT(1)/(2*pi) 1]);
197 ylim([0 2.5]);
198 vline(min([Cutoff_freqnmp,Cutoff_freqmp,0.2]),'k:');
199 text(min([Cutoff_freqnmp,Cutoff_freqmp,0.2])*1.1,2.2,...
200     {sprintf('$f_c^*=%2.2f$ Hz',min([Cutoff_freqnmp,Cutoff_freqmp,0.2]))});
201 xlabel('Frequency, $f$ Hz','FontSize',14);
202 ylabel({'\makebox[4in][c]{Iteration gain,}',...
203     '\makebox[4in][c]{$\rho(f)$}'},'FontSize',14,...
204     'HorizontalAlignment','center','Interpreter','latex');
205 grid on;
206 h = legend({'$\rho^*$ (MP)$\quad$','$\rho^*$ (NMP)$\quad$','Chosen ...
       iteration gain, $\rho$'},...
207     'location','northoutside','orientation','horizontal');
208 legend boxoff;
209 set(h,'Interpreter','latex','FontSize',14);
210 set(findall(gcf,'Type','Line'),'linewidth',1.5);
211 text(0.95,2.25,'(c)')
```

A.7.3 Iterative Learning Control - Frequency domain

```
function uopt = ILC(y_k, yd_k, uh_k, uc_k, time)

model_unknown = 1;

% FFT of the output and input at iteration k
NFFT = length(y_k)-1;
wopt = [-NFFT/2:NFFT/2]*2*pi/max(time);
Y_k = fftshift(fft(y_k));
ED_k = fftshift(fft(yd_k-y_k));
UH_k = fftshift(fft(uh_k));
UC_k = fftshift(fft(uc_k));

s = tf('s');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% If model of plant known before hand
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Model of Plant (Actual Dynamics)
mass = 1.0;
omega_n = 2.0;
xi = 1.0;
kspring = mass*omega_n^2;

G_m = 1/(s^2+2*xi*omega_n*s+omega_n^2);

% Human feedback model (Analytical Verbal Model)
K = 25;
```

```
tau_L = 2.5;
tau_D = 0.2;
tau_N = 0.1;
G_h = pade(K*(tau_L*s+1)*exp(-tau_D*s)/(tau_N*s+1));

% Feedback transfer function
G_fb = G_m*G_h/(1+G_m*G_h);

% Optimal inverse of G_FB
K1 = 1e2;
a = 0.01*omega_n;
Q = K1*a/(s+a);
R = (s+a)/a;

G_fb_inv = G_fb'*Q/(R+G_fb'*Q*G_fb);

G_FB_inv = reshape(freqresp(G_fb_inv,wopt),size(wopt));

phase = atan(imag(G_FB_inv)./real(G_FB_inv));
mag = (real(G_FB_inv).^2 + imag(G_FB_inv).^2).^(0.5);

figure(5);
subplot(2,1,1);
semilogx(wopt,mag);
ylabel('Mag, G_opt_inv');
hold on
subplot(2,1,2);
semilogx(wopt,phase);
ylabel('Phase');
xlabel('\omega, rad/s');
hold on
% Iterative scheme
```

```
61 rho = 0.25;
62 % UC_k = UC_k + rho*G_FB_inv.*(UH_k);
63
64 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
65 % If model is not known before hand
66 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
67 if model_unknown > 0
68     % System Model (Frequency response) derived from output and
69     % input
70     G_hat = Y_k./(UH_k+UC_k);
71
72     % Frequency response of human model (Using Analytical Verbal model)
73     GH_freq = reshape(freqresp(G_h,wopt),size(wopt));
74
75     % Feedback Frequency response
76     G_fb_freq = G_hat.*GH_freq./(1+G_hat.*GH_freq);
77     UC_k = UC_k + rho*ED_k./G_hat;
78 end
79
80 % Control Input for ILC update
81 uopt = ifft(ifftshift(UC_k));
82 end
```

A.7.4 Visualizing the allowable modeling uncertainty from Lemma 1 and 2 for one-shot robust intent estimation

```
% Testing the Lemma 1 Condition for arbitrary choices of Ghat and G
% Author: RBW
% Modified: 2:35 PM, 6/29/2018
figure(1); clf;
rng(123);

case_= 2;

switch case_
case 1
% Case 1: |Ghat| > 1 and Ghat->0
    rand_rad = [logspace(log10(0.01),log10(1),25) ones(1,10) ...
       logspace(log10(1),log10(3),25)];
    rand_ang = 2*pi*rand(1,1).*ones(size(rand_rad));
    filename = 'case1.gif';
case 2
% Case 2: |Ghat| < 1 and Ghat->1
    rand_rad = linspace(3,1,50);
    rand_ang = linspace(1.5*pi,2*pi,numel(rand_rad));
    filename = 'case2.gif';
end

ahat = rand_rad.*cos(rand_ang);
bhat = rand_rad.*sin(rand_ang);
Ghat = ahat + 1j*bhat;

Lx = -(1-ahat)./(abs(1./Ghat).^2-1);
```

```
27 Ly = (bhat)./(abs(1./Ghat).^2-1);
28 Lrad = abs(abs(1-Ghat).*abs(1./Ghat)./(abs(1./Ghat).^2-1));
29
30 %% Acceptable regions (Lemma 1)
31 Dhat = zeros(size(ahat));
32 for i=1:size(Ghat,1)
33     for j=1:size(Ghat,2)
34         h = figure(1);
35         cla
36         axis equal
37         hold on
38         xlim([-max(rand_rad) max(rand_rad)]);
39         ylim([-max(rand_rad) max(rand_rad)]);
40         x_lim = xlim;
41         y_lim = ylim;
42         % plot(real(G(keep1)),imag(G(keep1)),'g.'); % Plot the G for ...
               which Lemma 1 condition is true
43         % plot(real(G(~keep1)),imag(G(~keep1)),'r.'); % Plot the G for ...
               which Lemma 1 condition is false
44         if abs(Ghat(i,j))~=1
45             if abs(Ghat(i,j)) < 1
46                 patch(ahat(i,j)+Lx(i,j)+Lrad(i,j).*cos(0:0.01:2*pi), ...
                       bhat(i,j)+Ly(i,j)+Lrad(i,j).*sin(0:0.01:2*pi), 'g'); ...
                       % Plot the interior circle with the condition
47                 plot(ahat(i,j)+Lx(i,j)+Lrad(i,j).*cos(0:0.01:2*pi), ...
                       bhat(i,j)+Ly(i,j)+Lrad(i,j).*sin(0:0.01:2*pi), 'r-', ...
                       'linewidth', 1.5); % Plot the circle with the condition
48             else
49                 patch([x_lim(1) x_lim(1) x_lim(2) x_lim(2)], [y_lim(1) ...
                       y_lim(2) y_lim(2) y_lim(1)], 'g');
50                 patch(ahat(i,j)+Lx(i,j)+Lrad(i,j).*cos(0:0.01:2*pi), ...
                       bhat(i,j)+Ly(i,j)+Lrad(i,j).*sin(0:0.01:2*pi), 'w'); ...
```

```
                % Plot the interior circle with the condition
51              plot(ahat(i,j)+Lx(i,j)+Lrad(i,j).*cos(0:0.01:2*pi), ...
                    bhat(i,j)+Ly(i,j)+Lrad(i,j).*sin(0:0.01:2*pi), 'r-', ...
                    'linewidth', 1.5); % Plot the circle with the condition
52          end
53      elseif real(Ghat(i,j))~=1 && imag(Ghat(i,j))~=0
54          ddones = x_lim(1):0.01:x_lim(2);
55          x = ahat(i,j) + ddones;
56          y = bhat(i,j) + (1-ahat(i,j)).*ddones./bhat(i,j) - ...
                abs(1-Ghat(i,j)).^2/(2*bhat(i,j));
57          patch([x_lim(1) x_lim(1) x(1) x(end)],[y_lim(1) y_lim(2) ...
                y(1) y(end)],'g');
58          plot(x,y,'r-','linewidth',1.5);
59      end
60      Gt = text(ahat(i,j),bhat(i,j),sprintf('$\\hat{G}: ...
            (|\\hat{G}|=%.2f)$',abs(Ghat(i,j))));
61      Gt.Color = 'b';
62      plot(cos(0:0.01:2*pi),sin(0:0.01:2*pi),'k--'); % Plot the unit ...
            circle
63      plot(ahat(1:i,1:j),bhat(1:i,1:j),'b.'); % Plot the point Ghat = ...
            (ahat,bhat)
64      xL = xlim;
65      yL = ylim;
66      line([0 0], yL, 'color', 'k');  %y-axis
67      line(xL, [0 0], 'color', 'k');  %x-axis
68      drawnow
69      % Capture the plot as an image
70      frame = getframe(gca);
71      im = frame2im(frame);
72      [imind,cm] = rgb2ind(im,256);
73      % Write to the GIF File
74      if j == 1
```

```
75              imwrite(imind, cm, filename, 'gif', 'Loopcount', inf, ...
                    'DelayTime', 0);
76          else
77              imwrite(imind, cm, filename, 'gif', 'WriteMode', 'append', ...
                    'DelayTime', 0);
78          end
79
80          %% Lemma 2
81          % Compute the size of the maximum possible set M
82          if abs(Ghat(i,j)) == 1
83              Dhat(i,j) = ...
                    abs(1-Ghat(i,j)).^2./(2*(abs(1-ahat(i,j))+abs(bhat(i,j))));
84          elseif abs(Ghat(i,j)) < 1
85              b = (abs(1-ahat(i,j))+abs(bhat(i,j)));
86              a = abs(1./Ghat(i,j)).^2-1;
87              c = -abs(1-Ghat(i,j)).^2/2;
88              Dhat1 = (-b+sqrt(b.^2-4*a.*c))./(2*a);
89              Dhat2 = (-b-sqrt(b.^2-4*a.*c))./(2*a);
90              Dhat(i,j) = max([Dhat1,Dhat2]);
91          else
92              % Solve the quadratic (a*Dhat^2 + b*Dhat + c = 0) for Dhat
93              b = -(abs(Lx(i,j))+abs(Ly(i,j)));
94              a = 1;
95              c = 0.5*(Lx(i,j).^2+Ly(i,j).^2-Lrad(i,j).^2);
96              Delta_cond = cell(size(Dhat));
97              Delta11 = (-b(i,j) + sqrt(b(i,j).^2-4*a.*c(i,j))./(2*a) % ...
                    Root 1
98              Delta12 = (-b(i,j) - sqrt(b(i,j).^2-4*a.*c(i,j))./(2*a) % ...
                    Root 2
99              Dhat(i,j) = min([Delta11 Delta12]); % Choose closest valid ...
                    solution of the quadratic
100             Delta_cond{i,j} = 'Delta < |Ly| && Delta < |Ly|';
```

```
101             if Dhat(i,j) > abs(Lx(i,j)) % Delta > |Lx| --- Choose ...
                    delta_a = |Lx|
102                 Dhat(i,j) = abs(abs(Ly(i,j))-Lrad(i,j));
103                 Delta_cond{i,j} = 'Delta > |Lx|';
104             elseif Dhat(i,j) > abs(Ly(i,j)) % Delta > |Ly| --- Choose ...
                    delta_b = |Ly|
105                 Dhat(i,j) = abs(abs(Lx(i,j))-Lrad(i,j));
106                 Delta_cond{i,j} = 'Delta > |Ly|';
107             end
108         end
109     end
110 end
```

A.7.5 Visualizing the allowable modeling uncertainty from Lemma 3 for one-shot robust intent estimation

```
% Plotting the Lemma 2 conditions for a given (Ghatinv, Delta) model
% Author: Rahul B. Warrier
% Modified: August 2, 2018
%
% INPUTS:   Ghatinv : (n_omega x 1) complex-valued mean estimates
%           Delta_a : (n_omega x 1) uncertainty in real component
%           Delta_b : (n_omega x 1) uncertainty in imag component
% OUTPUTS:  Dhat    : Upper bound on model uncertainty
function [Dhat,x_lim] = LemmaConditionsCheck(Ghatinv, Delta_a, Delta_b, ...
    textlabel_handle, freq, x_lim, varargin)
try
    % Default Parameters
    plot_results = 1;
    save_plot = 0;
    save_animation = 0;
    animation_filename = 'Lemma12_Animation';
    results_folder = '.';
    filename = 'Lemma12_Plot';
    textlabel = '';
    fc = 1;
    keep = true * ones(size(Ghatinv));

    % User-defined parameters
    for i=1:numel(varargin)
        switch varargin{i}
        case 'plot_results'
            plot_results = varargin{i+1};
```

```
27          case 'save_plot'
28              save_plot = varargin{i+1};
29          case 'results_folder'
30              results_folder = varargin{i+1};
31          case 'filename'
32              filename = varargin{i+1};
33          case 'save_animation'
34              save_animation = varargin{i+1};
35          case 'animation_filename'
36              animation_filename = varargin{i+1};
37          case 'textlabel'
38              textlabel = varargin{i+1};
39          case 'fc'
40              fc = varargin{i+1};
41          end
42      end
43
44      if ~isempty(textlabel_handle) && ~isempty(freq)
45          for i=1:numel(freq)
46              textlabel{i} = textlabel_handle(freq(i));
47          end
48          keep = freq <= fc;
49      end
50
51      % Initialize
52      if plot_results, figure(2); clf; set(gcf, 'PaperUnits', 'inches', ...
            'Visible','on', 'WindowStyle', 'docked'); width=8; height=width; ...
            set(gcf, 'PaperPositionMode', 'manual'); papersize = get(gcf, ...
            'PaperSize'); left = (papersize(1)- width)/2; bottom = ...
            (papersize(2)- height)/2; set(gcf, 'PaperPosition', [left bottom ...
            width height]); end
53      % Real/imag components of Ghatinv
```

```
54      ahat = real(Ghatinv);
55      bhat = imag(Ghatinv);
56      Dhat = zeros(size(Ghatinv));
57      % Grid size for possible deltas
58      if isempty(x_lim)
59          Delta = 2*max([abs(real(Ghatinv(keep))) abs(imag(Ghatinv(keep)))]);
60      else
61          Delta = max(x_lim);
62      end
63      for i=1:numel(Ghatinv(keep))
64          % Form a grid of deltas in set N
65          delta_a = linspace(ahat(i)-2*Delta,ahat(i)+2*Delta,200);
66          delta_b = linspace(bhat(i)-2*Delta,bhat(i)+2*Delta,200);
67          [delta_a,delta_b] = meshgrid(delta_a,delta_b);
68          % Ginv = Ghatinv-delta
69          Ginv = (ahat(i)-delta_a) +1j*(bhat(i)-delta_b);
70          % Lemma 1 condition
71          lemma1_condition = abs(Ginv-Ghatinv(i)) - abs(Ginv - 1) < 0;
72          % Plot the results
73          if plot_results
74              cla
75              axis equal
76              box on
77              grid on
78              hold on
79              %
80              xlim([-Delta Delta]);
81              ylim([-Delta Delta]);
82              x_lim = xlim;
83              y_lim = ylim;
84              plot(1,0,'bo') % Plot the point (1,0)
```

```
85              plot(cos(0:0.01:2*pi),sin(0:0.01:2*pi),'k--'); % Plot the ...
                    unit circle
86          end
87          % Compute the size of Mhat according to Lemma 2 (assuming ...
                Delta_a <= Delta and Delta_b <= Delta)
88          Dhat(i) = (((ahat(i)-1).^2 + ...
                bhat(i).^2)./(2*(abs(ahat(i)-1)+abs(bhat(i)))));
89          % Coordinates of the set Mhat (using theoretical upper bound ...
                uncertainty)
90          Mhat_x = [ahat(i)-Dhat(i), ahat(i)+Dhat(i), ahat(i)+Dhat(i), ...
                ahat(i)-Dhat(i), ahat(i)-Dhat(i)];
91          Mhat_y = [bhat(i)-Dhat(i), bhat(i)-Dhat(i), bhat(i)+Dhat(i), ...
                bhat(i)+Dhat(i), bhat(i)-Dhat(i)];
92          % Coordinates of the set M_tilde (using given uncertainty ...
                estimates)
93          Mtilde_x = [ahat(i)-Delta_a(i), ahat(i)+Delta_a(i), ...
                ahat(i)+Delta_a(i), ahat(i)-Delta_a(i), ahat(i)-Delta_a(i)];
94          Mtilde_y = [bhat(i)-Delta_b(i), bhat(i)-Delta_b(i), ...
                bhat(i)+Delta_b(i), bhat(i)+Delta_b(i), bhat(i)-Delta_b(i)];
95          if plot_results && keep(i)
96              a_ok = real(Ginv(lemma1_condition)); b_ok = ...
                    imag(Ginv(lemma1_condition));
97              setMin = boundary(a_ok,b_ok,0);
98              h1 = patch(a_ok(setMin),b_ok(setMin),'g');
99              set(h1,'LineWidth',1.5);
100             %
101             x_lim = xlim;
102             y_lim = ylim;
103             if iscell(textlabel)
104                 text(diff(x_lim)/4,diff(y_lim)/3,sprintf('%s', ...
                        textlabel{i}), 'BackgroundColor', 'white');
105             else
```

```
106                 text(diff(x_lim)/4,diff(y_lim)/3,sprintf('%s', ...
                        textlabel), 'BackgroundColor', 'white');
107             end
108             plot(1,0,'bo') % Plot the point (1,0)
109             plot(cos(0:0.01:2*pi),sin(0:0.01:2*pi),'k--'); % Plot the ...
                    unit circle
110             plot([0 0], [y_lim(1) y_lim(2)], 'k-', 'linewidth', 0.5);
111             plot([x_lim(1) x_lim(2)], [0 0], 'k-', 'linewidth', 0.5);
112             h1 = plot(Mhat_x,Mhat_y,'r-','linewidth',1.2); % Plot the ...
                    boundaries of the set Mtilde
113             h1 = patch(Mtilde_x,Mtilde_y,'k'); % Plot the boundaries of ...
                    the set Mtilde
114             hPatch1 = findobj(h1, 'Type', 'patch');
115             hh2 = hatchfill(hPatch1, 'cross', 45, 3);
116             set(hh2, 'Color', [0.5 0.5 0.5]);
117             set(h1, 'EdgeColor', 'black')
118             plot(ahat(i), bhat(i), 'ko', 'MarkerFaceColor', 'k', ...
                    'MarkerSize', 7.5); % Plot the point Ghatinv = (ahat,bhat)
119             labelpoints(ahat(i), bhat(i), '$\hat{G}^{-1}$', 'SE', ...
                    'Interpreter', 'latex', 'FontSize', 20);
120             drawnow
121             if save_animation && abs(Ghatinv(i)) > 0.05*max(abs(Ghatinv))
122                 % Capture the plot as an image
123                 frame = getframe(gca);
124                 im = frame2im(frame);
125                 [imind,cm] = rgb2ind(im,256);
126                 % Write to the GIF File
127                 if i == 1
128                     animation_filename = sprintf('%s/%s', ...
                            results_folder, animation_filename);
129                     fprintf('\nWriting to GIF: %s\n', animation_filename);
```

```
130                     imwrite(imind, cm, animation_filename, 'gif', ...
                            'Loopcount', inf, 'DelayTime', 0);
131                 else
132                     imwrite(imind, cm, animation_filename, 'gif', ...
                            'WriteMode', 'append', 'DelayTime', 0);
133                 end
134             end
135             if save_plot, export_fig(sprintf('%s/%s.eps', ...
                    results_folder, filename), '-depsc'); end
136         end
137     end
138 catch E
139     E
140     keyboard
141 end
142 end
```

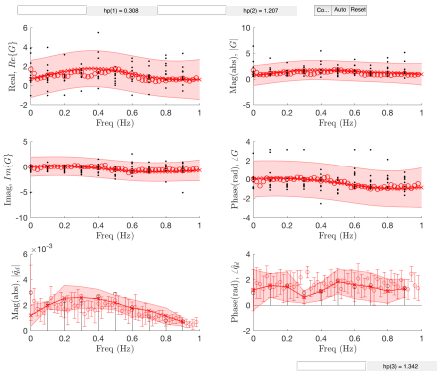
Appendix B

HUMAN MODELING

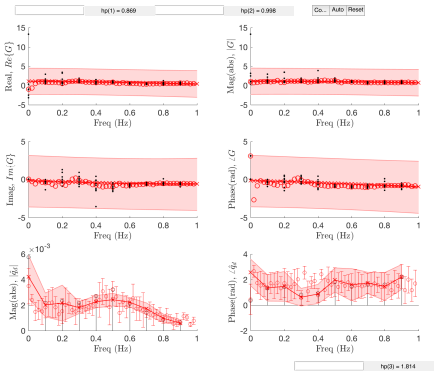
This chapter presents the modeling results for one of the multiple subjects using the non-parametric CGPR technique described in Chapter 4.

B.1 Models obtained from Train/Test trajectories

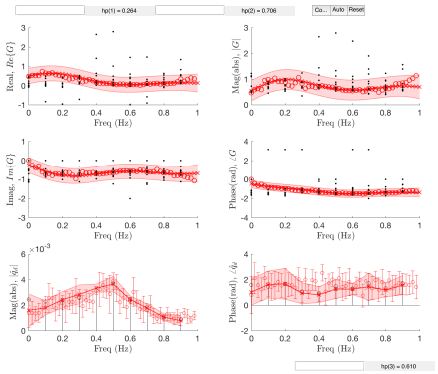
B.1.1 SISO (FWD) $[e \rightarrow r_h]$



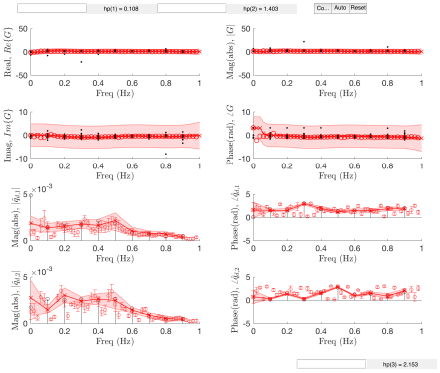
(a) 1s preview (Train)



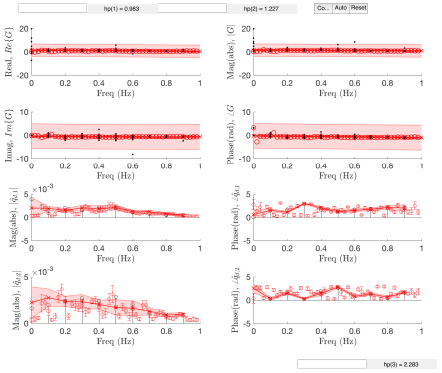
(b) 0.5s preview (Train)



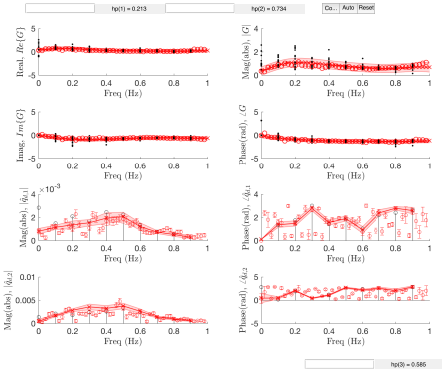
(c) No preview (Train)



(d) 1s preview (Test)



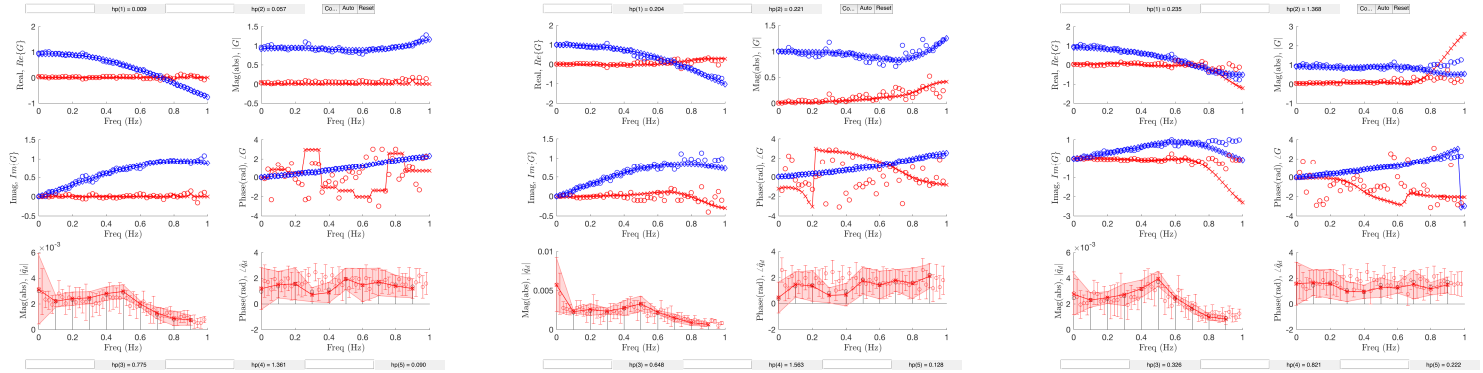
(e) 0.5s preview (Test)



(f) No preview (Test)

Figure B.1: SISO Forward models $e \rightarrow r_h$. (—○—): experimental data, (—x—): estimated data, (●): Least-square estimate.

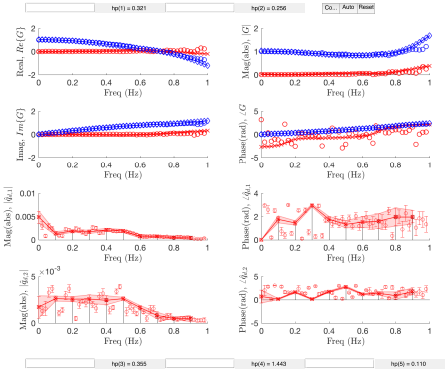
B.1.2 MISO (FWD) $[(qd, q) \to r_h]$



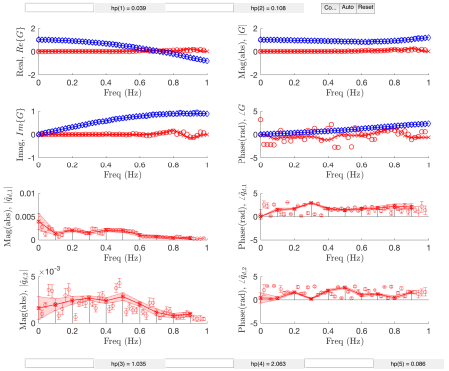
(a) 1s preview (Train)

(b) 0.5s preview (Train)

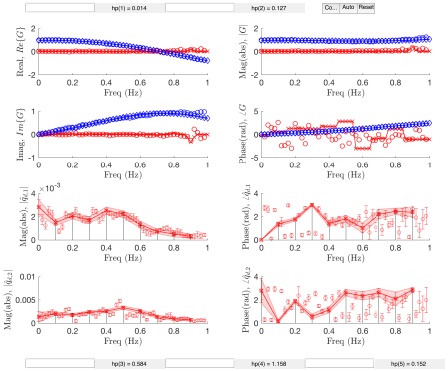
(c) No preview (Train)



(d) 1s preview (Test)



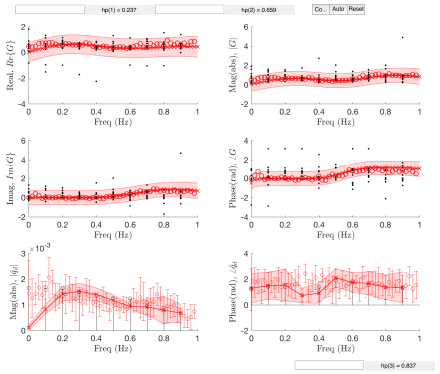
(e) 0.5s preview (Test)



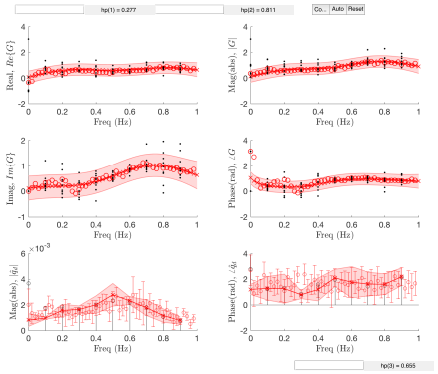
(f) No preview (Test)

Figure B.2: MISO Forward models $(q_d, q) \to r_h$. ( ): $q_d \to r_h$ (fwd); ( ): $q \to r_h$ (fwd), (●): Least-square estimate $q_d \to r_h$, (●): Least-square estimate $q \to r_h$.

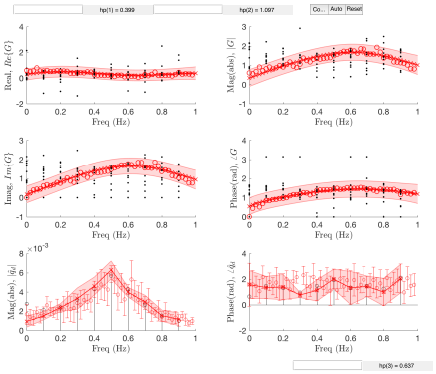
B.1.3 SISO (INV) [$(r_h \rightarrow e$]



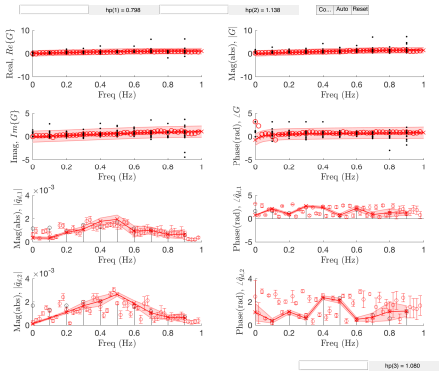
(a) 1s preview (Train)



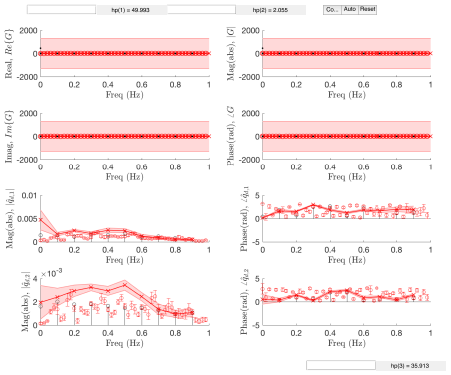
(b) 0.5s preview (Train)



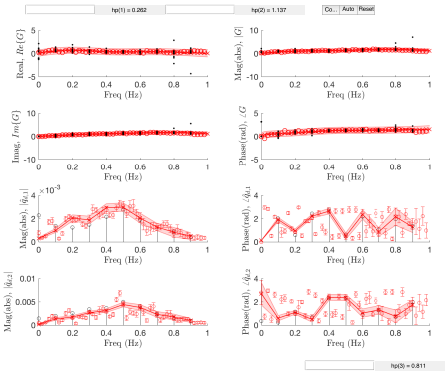
(c) No preview (Train)



(d) 1s preview (Test)



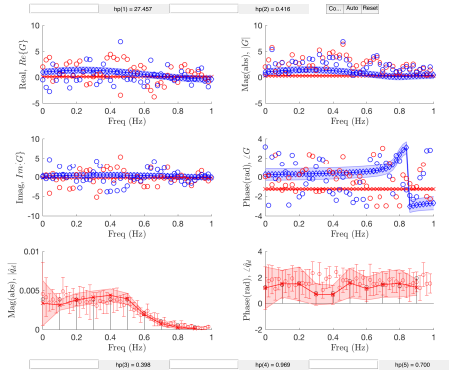
(e) 0.5s preview (Test)



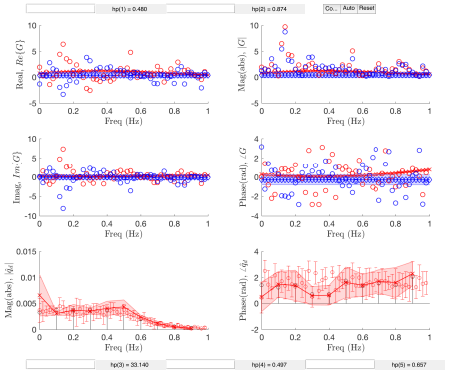
(f) No preview (Test)

Figure B.3: SISO Inverse models $r_h \rightarrow e$. (—○—): experimental data, (—x—): estimated data, (●): Least-square estimate.

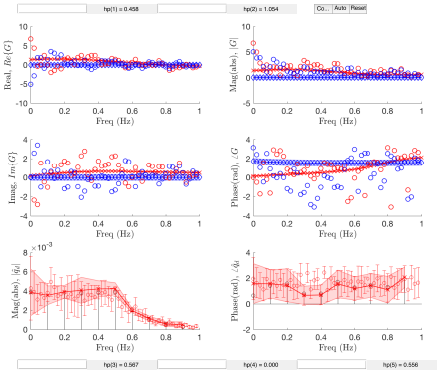
B.1.4 MISO (INV) $[(r_h, q) \to q_d]$



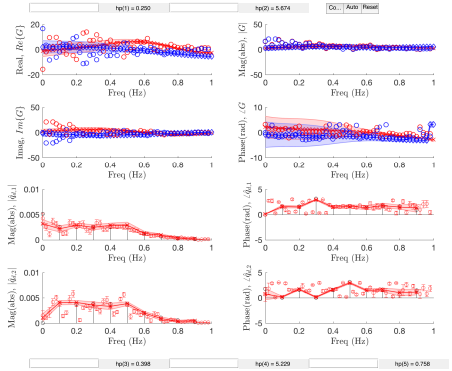
(a) 1s preview (Train)



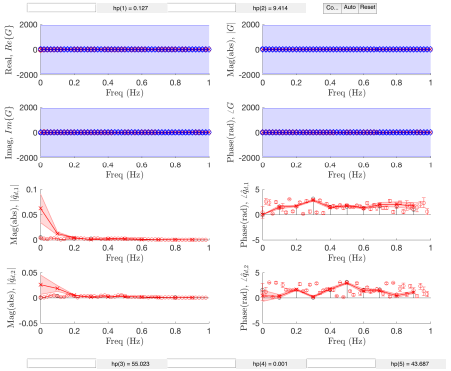
(b) 0.5s preview (Train)



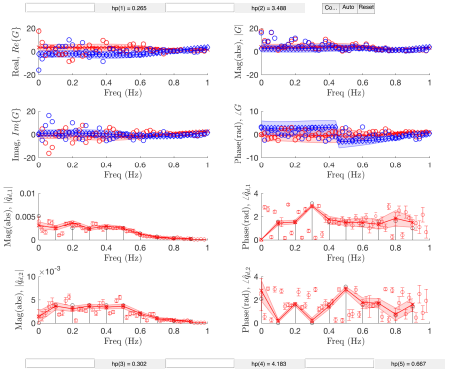
(c) No preview (Train)



(d) 1s preview (Test)



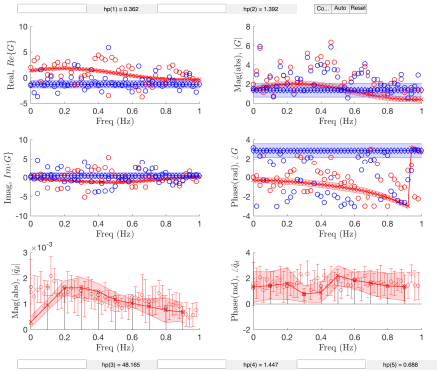
(e) 0.5s preview (Test)



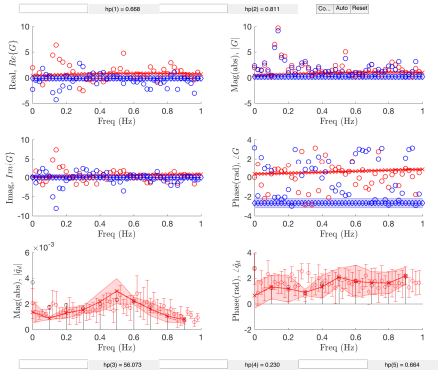
(f) No preview (Test)

Figure B.4: MISO Inverse models $(r_h, q) \to q_d$. (—x—): $r_h \to q_d$ (inv); (—o—): $q \to q_d$ (inv), (●): Least-square estimate $r_h \to q_d$, (●): Least-square estimate $q \to q_d$.

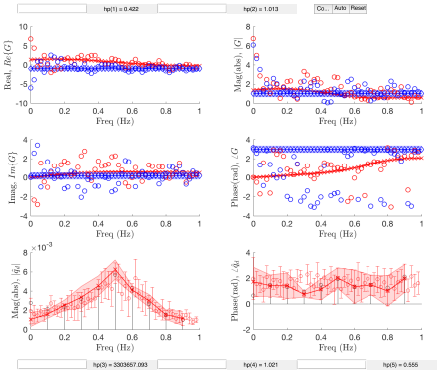
B.1.5 MISO (INV) $[(r_h, q) \to e]$



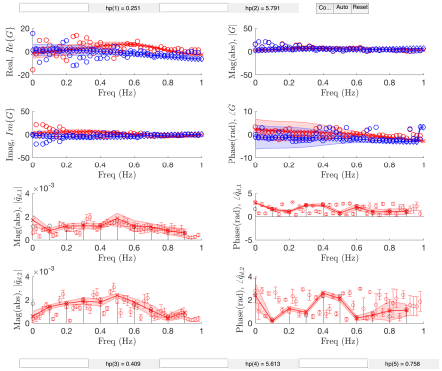
(a) 1s preview (Train)



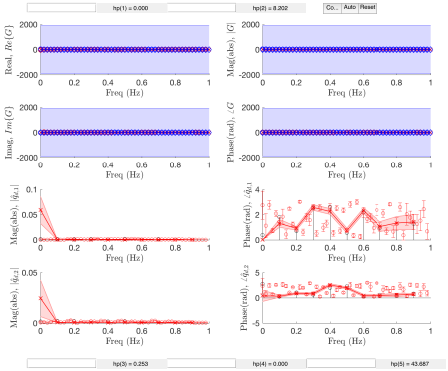
(b) 0.5s preview (Train)



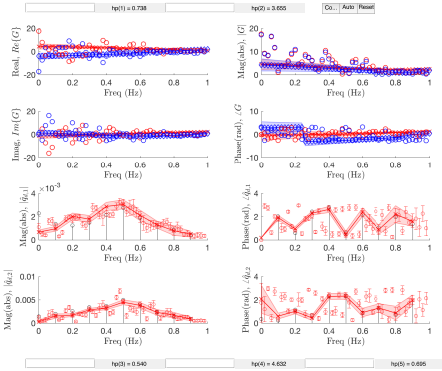
(c) No preview (Train)



(d) 1s preview (Test)



(e) 0.5s preview (Test)

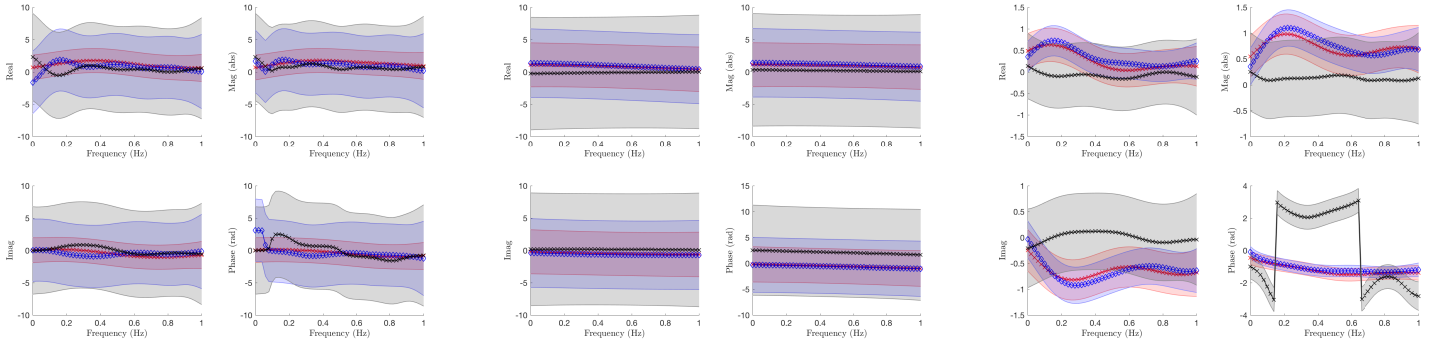


(f) No preview (Test)

Figure B.5: MISO Inverse models $(r_h, q) \to e$. (—x—): $r_h \to e$ (inv); (—o—): $q \to e$ (inv), (●): Least-square estimate $r_h \to e$, (●): Least-square estimate $q \to e$.

B.2 Difference in the Models obtained from Train/Test trajectories

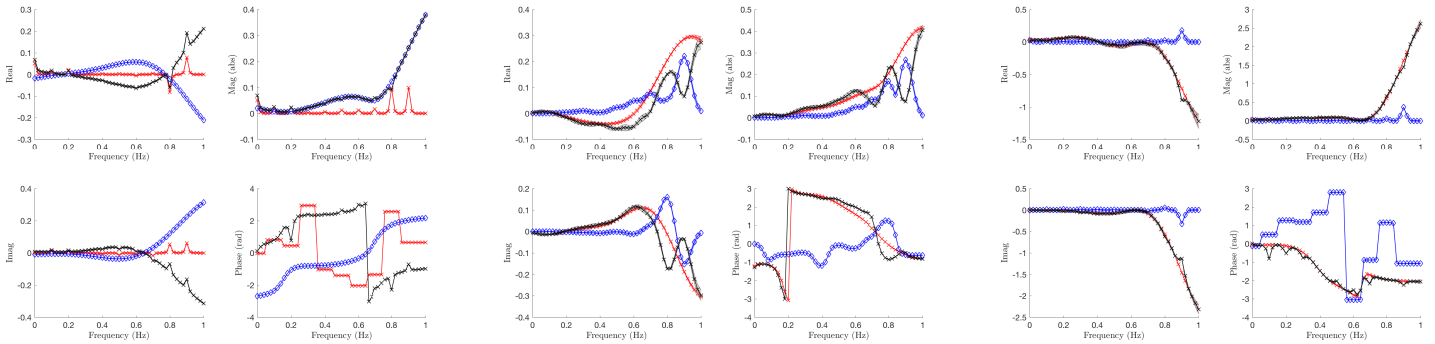
B.2.1 SISO (FWD) $[e \rightarrow r_h]$



(a) 1s preview (b) 0.5s preview (c) No preview

Figure B.6: SISO Forward models $e \rightarrow r_h$. (—x—): from Training data (g_{train}), (———): from Test data (g_{test}), (—x—): Difference ($g_{train} - g_{test}$).

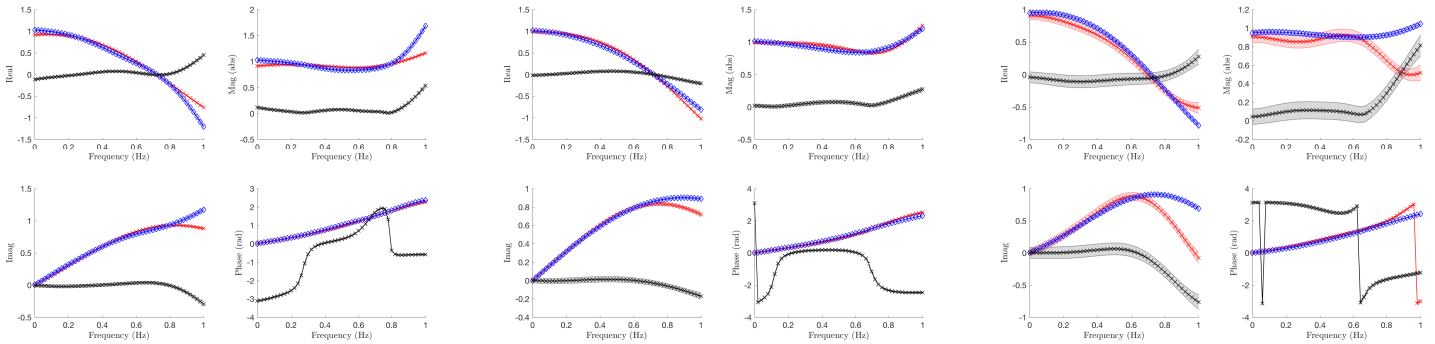
B.2.2 MISO (FWD) $[(q_d, q) \rightarrow r_h]$: Channel 1 $q_d \rightarrow r_h$



(a) 1s preview (b) 0.5s preview (c) No preview

Figure B.7: MISO Forward models $q_d \rightarrow r_h$. (—x—): from Training data (g_{train}), (————): from Test data (g_{test}), (—x—): Difference ($g_{train} - g_{test}$).

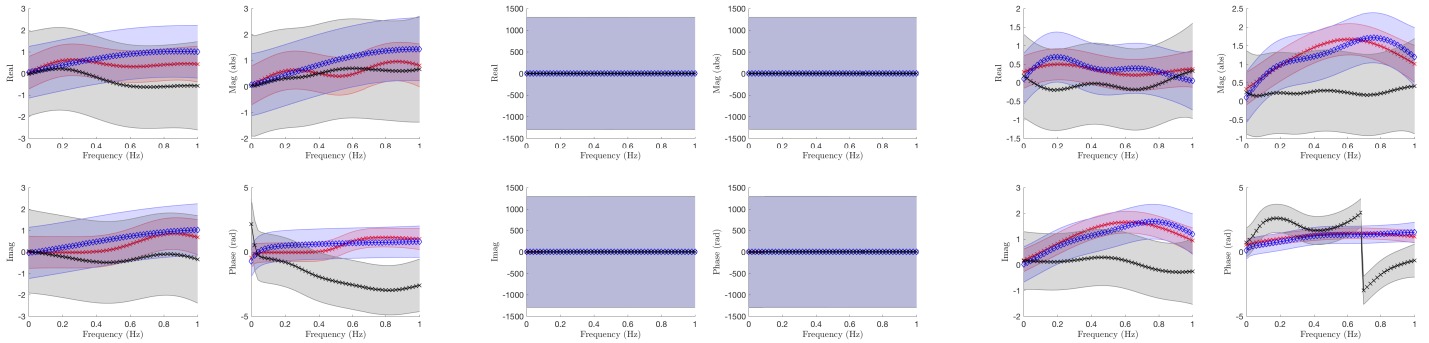
B.2.3 MISO (FWD) $[(q_d, q) \rightarrow r_h]$: Channel 2 $q \rightarrow r_h$



(a) 1s preview (b) 0.5s preview (c) No preview

Figure B.8: MISO Forward models $q \rightarrow r_h$. (—x—): from Training data (g_{train}), (——): from Test data (g_{test}), (—x—): Difference ($g_{train} - g_{test}$).

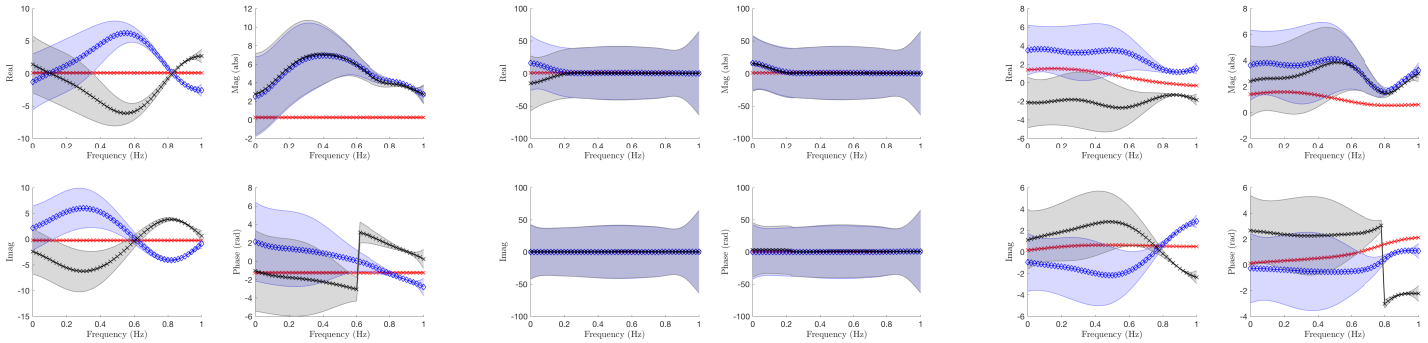
B.2.4 SISO (INV) $[r_h \rightarrow e]$



(a) 1s preview (b) 0.5s preview (c) No preview

Figure B.9: SISO Inverse models $r_h \rightarrow e$. (—x—): from Training data (g_{train}), (————): from Test data (g_{test}), (—x—): Difference ($g_{train} - g_{test}$).

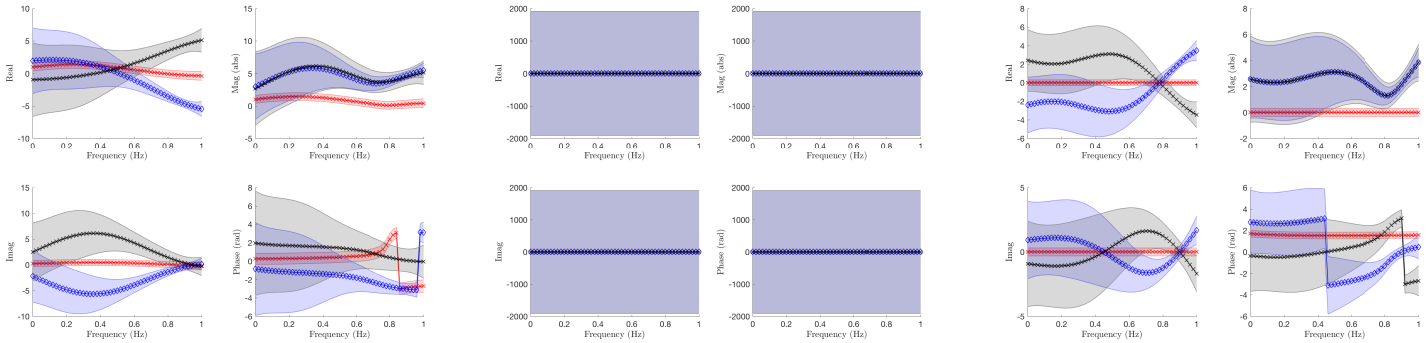
B.2.5 MISO (INV) $[(r_h, q) \rightarrow q_d]$: Channel 1 $r_h \rightarrow q_d$



(a) 1s preview (b) 0.5s preview (c) No preview

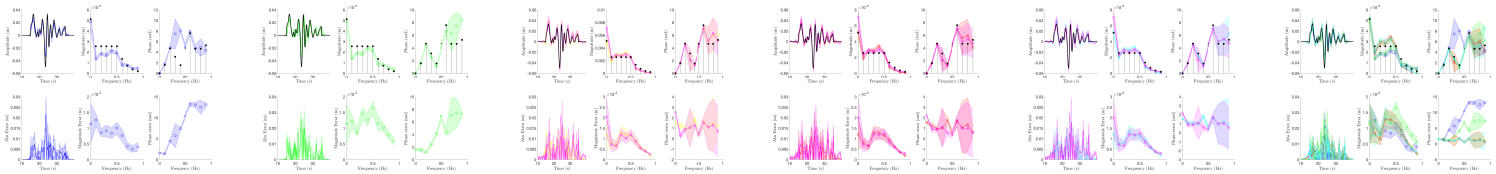
Figure B.10: MISO Inverse models $r_h \rightarrow q_d$. (—x—): from Training data (g_{train}), (——): from Test data (g_{test}), (—x—): Difference ($g_{train} - g_{test}$).

B.2.6 MISO (INV) $[(r_h, q) \rightarrow q_d]$: Channel 2 $q \rightarrow q_d$



(a) 1s preview (b) 0.5s preview (c) No preview

Figure B.11: MISO Forward models $q \rightarrow q_d$. (—x—): from Training data (g_{train}), (————): from Test data (g_{test}), (—x—): Difference ($g_{train} - g_{test}$).



(a) 1s preview (r_h)

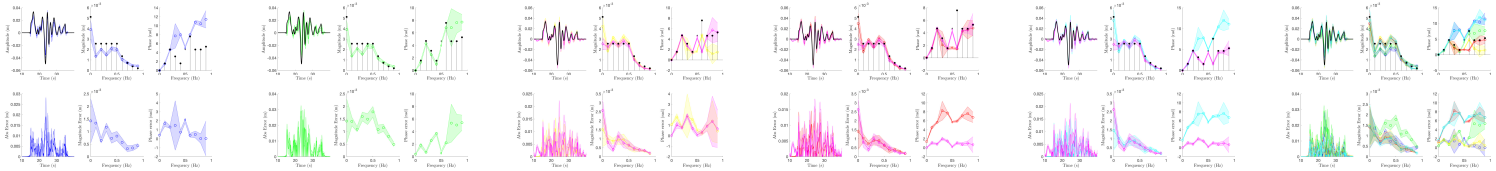
(b) 1s preview (q)

(c) 1s preview ($\hat{q}_{d,1}$)

(d) 1s preview ($\hat{q}_{d,2}$)

(e) 1s preview ($\hat{q}_{d,2^*}$)

(f) 1s preview (overall)



(g) 0.5s preview (r_h)

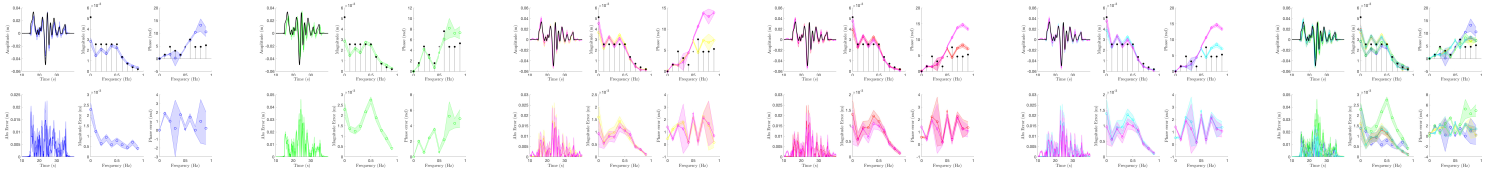
(h) 0.5s preview (q)

(i) 0.5s preview ($\hat{q}_{d,1}$)

(j) 0.5s preview ($\hat{q}_{d,2}$)

(k) 0.5s preview ($\hat{q}_{d,2^*}$)

(l) 0.5s preview (overall)



(m) No preview (r_h)

(n) No preview (q)

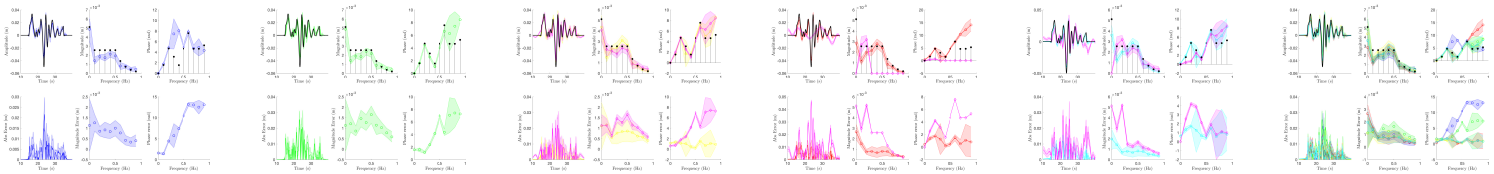
(o) No preview ($\hat{q}_{d,1}$)

(p) No preview ($\hat{q}_{d,2}$)

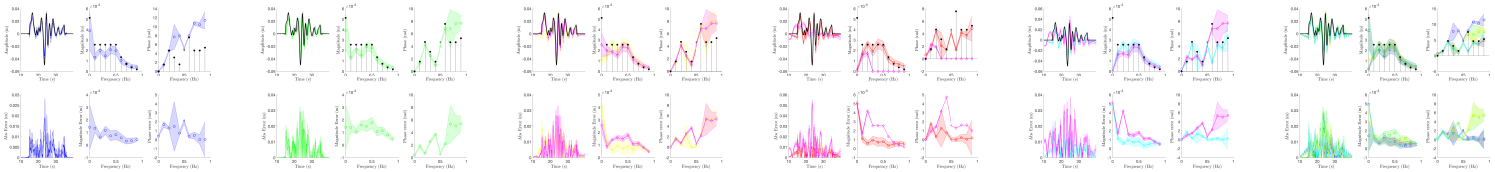
(q) No preview ($\hat{q}_{d,2^*}$)

(r) No preview (overall)

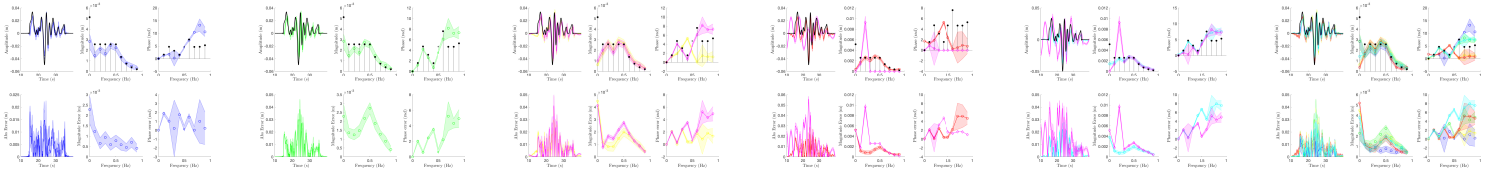
Figure B.12: Intent estimation on test trajectory 1 using inverse models trained on sum-of-sines training data. (———): desired output q_d, (———): human input r_h, (———): system output q, (———): 1-channel estimate $\hat{q}_{d,1}$, (———): 2-channel estimate $\hat{q}_{d,2}$ using model $[(r_h, q) \to q_d]$, (———): 2-channel estimate $\hat{q}_{d,2^*}$ using model $[(r_h, q) \to e]$, (———): Least-square estimate.



(a) 1s preview (r_h) (b) 1s preview (q) (c) 1s preview ($\hat{q}_{d,1}$) (d) 1s preview ($\hat{q}_{d,2}$) (e) 1s preview ($\hat{q}_{d,2^*}$) (f) 1s preview (overall)

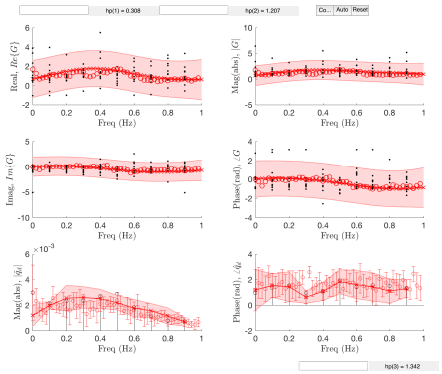


(g) 0.5s preview (r_h) (h) 0.5s preview (q) (i) 0.5s preview ($\hat{q}_{d,1}$) (j) 0.5s preview ($\hat{q}_{d,2}$) (k) 0.5s preview ($\hat{q}_{d,2^*}$) (l) 0.5s preview (overall)

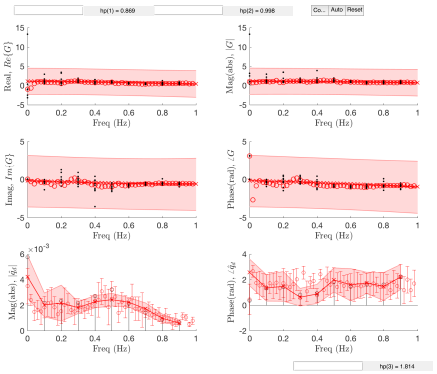


(m) No preview (r_h) (n) No preview (q) (o) No preview ($\hat{q}_{d,1}$) (p) No preview ($\hat{q}_{d,2}$) (q) No preview ($\hat{q}_{d,2^*}$) (r) No preview (overall)

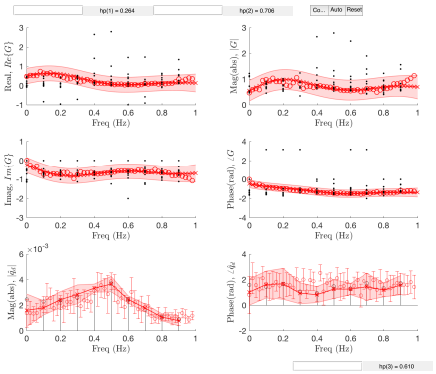
Figure B.13: Intent estimation on test trajectory 1 using inverse models trained on swept-sines training data. (———): desired output q_d, (———): human input r_h, (———): system output q, (———): 1-channel estimate $\hat{q}_{d,1}$, (———): 2-channel estimate $\hat{q}_{d,2}$ using model $[(r_h, q) \to q_d]$, (———): 2-channel estimate $\hat{q}_{d,2^*}$ using model $[(r_h, q) \to e]$, (———): Least-square estimate.



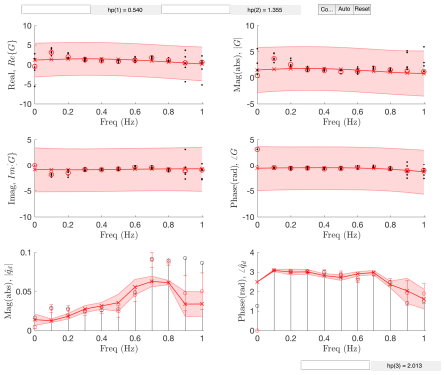
(a) 1s preview (sumsines)



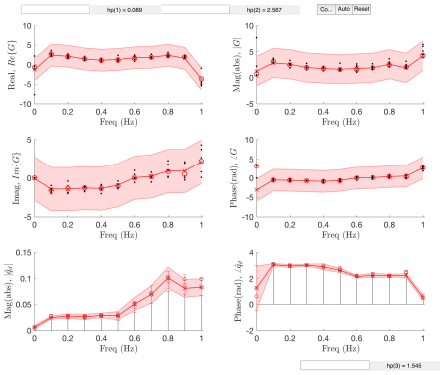
(b) 0.5s preview (sumsines)



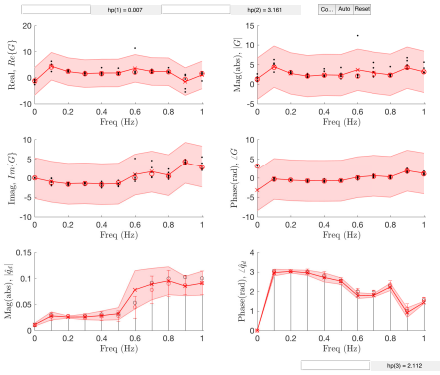
(c) No preview (sumsines)



(d) 1s preview (sweptsines)

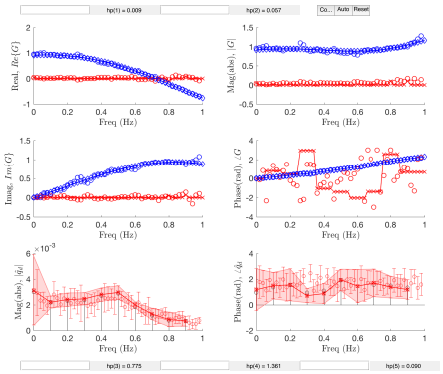


(e) 0.5s preview (sweptsines)

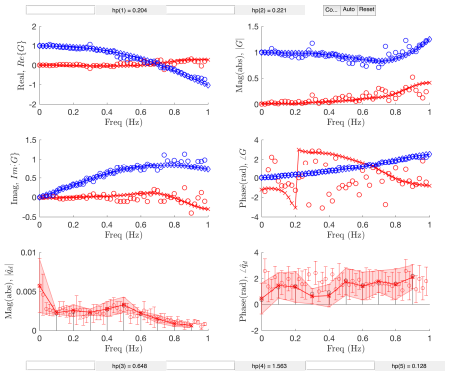


(f) No preview (sweptsines)

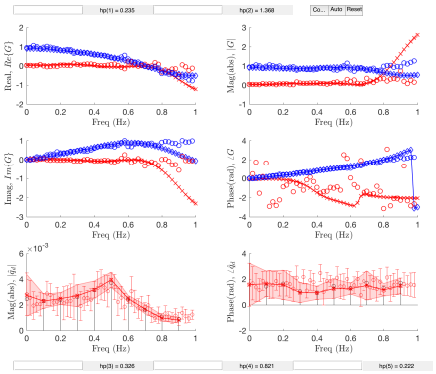
Figure B.14: Comparison of SISO Forward models $e \rightarrow r_h$ using sumsines/sweptsines training data. ( ): experimental data, ( ): estimated data, ( ): Least-square estimate.



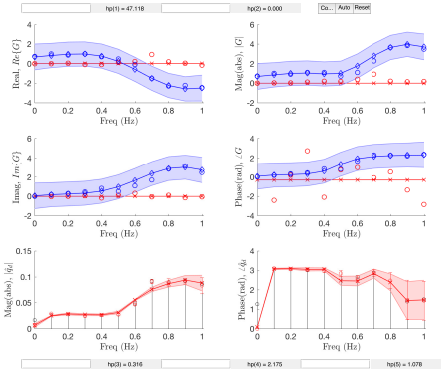
(a) 1s preview (sumsines)



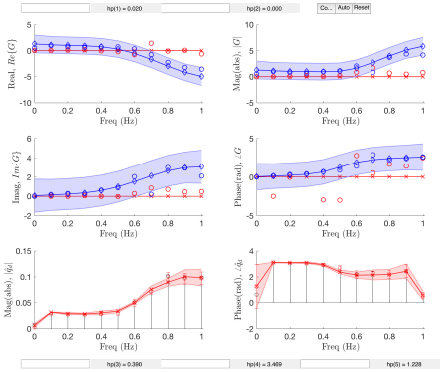
(b) 0.5s preview (sumsines)



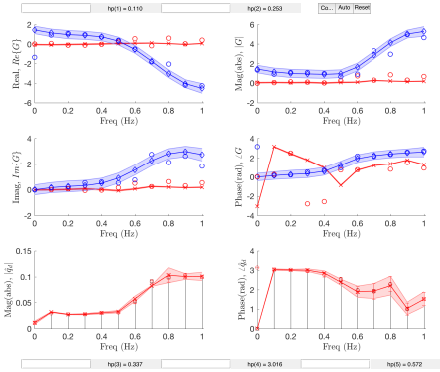
(c) No preview (sumsines)



(d) 1s preview (sweptsines)

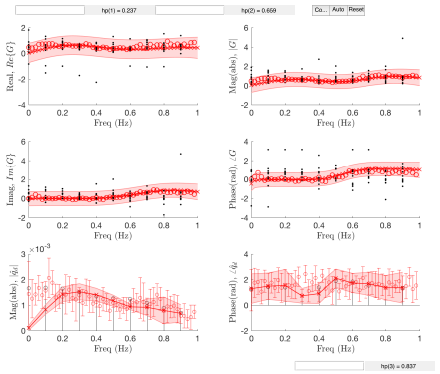


(e) 0.5s preview (sweptsines)

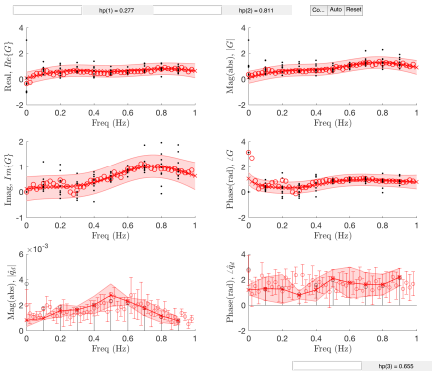


(f) No preview (sweptsines)

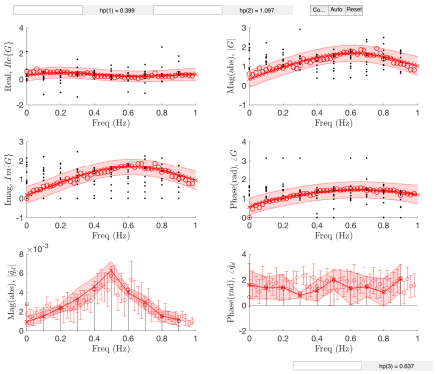
Figure B.15: Comparison of MISO Forward models $(q_d, q) \to r_h$ using sumsines/sweptsines training data. (—x—): $q_d \to r_h$ (fwd); (—o—): $q \to r_h$ (fwd), (●): Least-square estimate $q_d \to r_h$, (●): Least-square estimate $q \to r_h$.



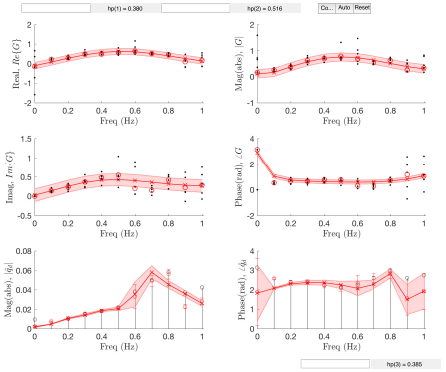
(a) 1s preview (sumsines)



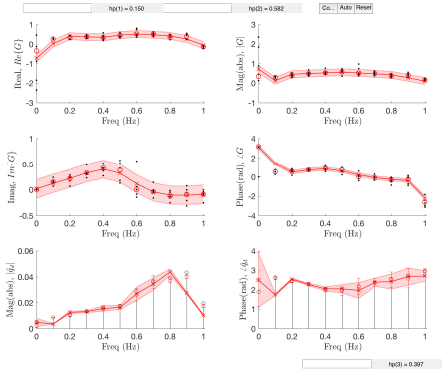
(b) 0.5s preview (sumsines)



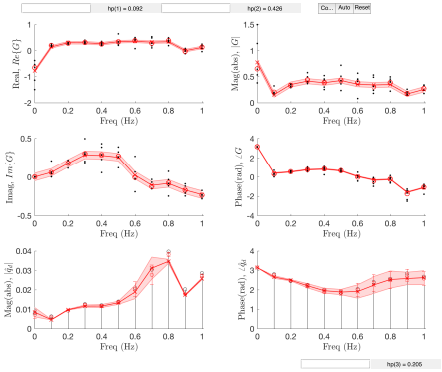
(c) No preview (sumsines)



(d) 1s preview (sweptsines)

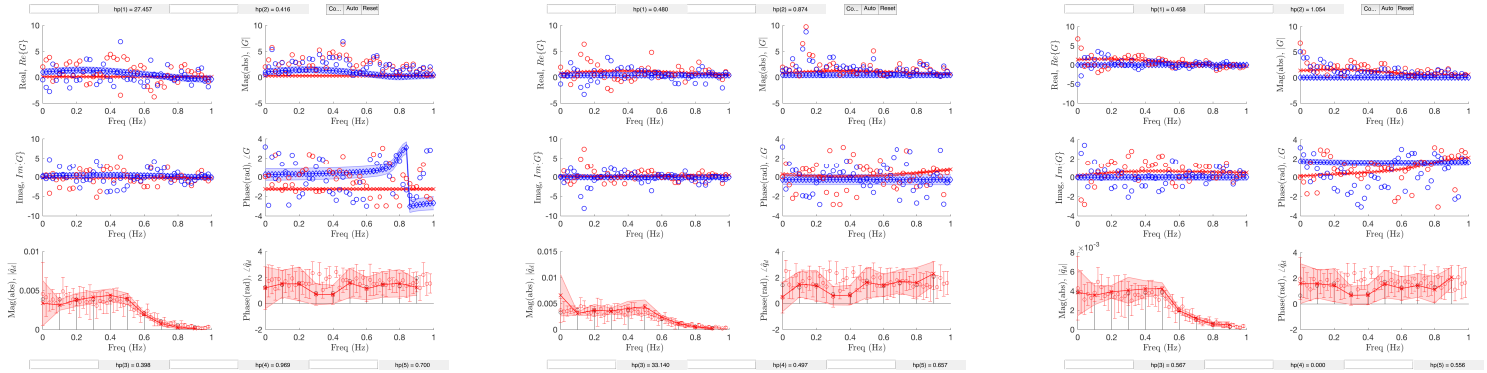


(e) 0.5s preview (sweptsines)



(f) No preview (sweptsines)

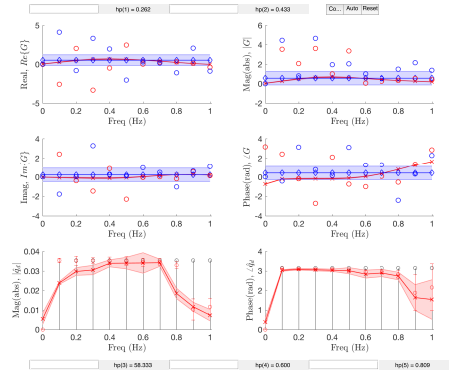
Figure B.16: Comparison of SISO Inverse models $r_h \to e$ using sumsines/sweptsines training data. (—○—): experimental data, (—x—): estimated data, (●): Least-square estimate.



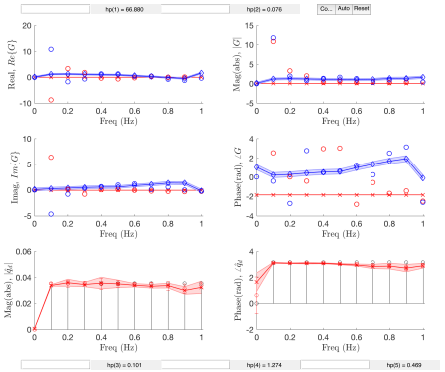
(a) 1s preview (sumsines)

(b) 0.5s preview (sumsines)

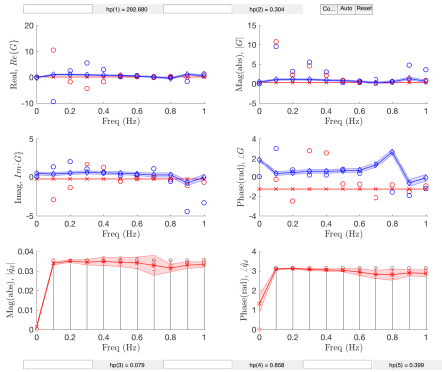
(c) No preview (sumsines)



(d) 1s preview (sweptsines)



(e) 0.5s preview (sweptsines)



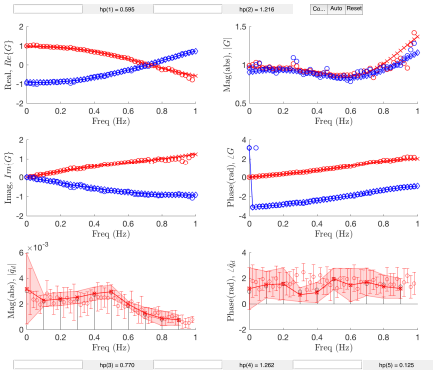
(f) No preview (sweptsines)

Figure B.17: Comparison of MISO Inverse models $(r_h, q) \to q_d$ using sumsines/sweptsines training data. (—x—): $r_h \to q_d$ (inv); (—o—): $q \to q_d$ (inv), (●): Least-square estimate $r_h \to q_d$, (●): Least-square estimate $q \to q_d$.

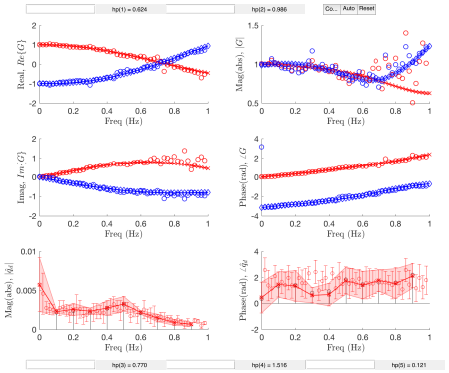
The scattering effect of the LSQ fits is because the underlying functions are not well defined. Consider the forward modeling equation for $(q_d, q) \to r_h$,

$$r_h(\omega) = g_1(\omega)q_d(\omega) + g_2(\omega)q(\omega) \quad q_d(\omega) = \frac{1}{g_1(\omega)}r_h(\omega) + \frac{-g_2(\omega)}{g_1(\omega)}q(\omega) \tag{B.1}$$

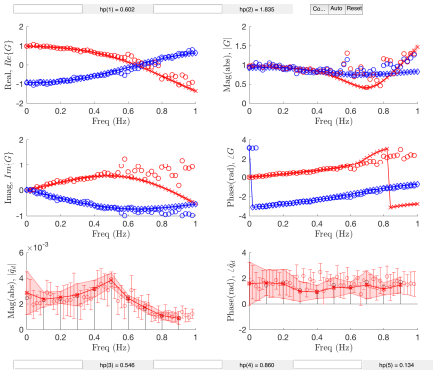
but, we see that $g_1(\omega) \approx 0$ for most frequencies $\omega \in [0, 1]$ Hz from Fig. B.15, resulting in large sensitivity of the inverse model estimates to noise.



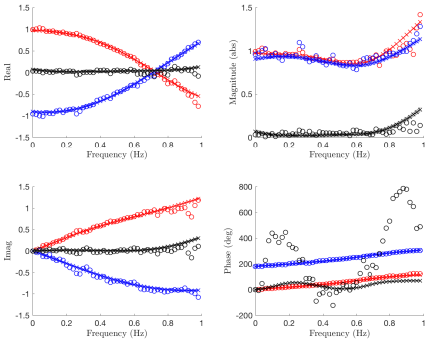
(a) 1s preview (sumsines)



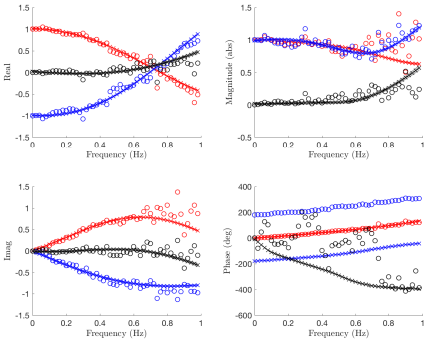
(b) 0.5s preview (sumsines)



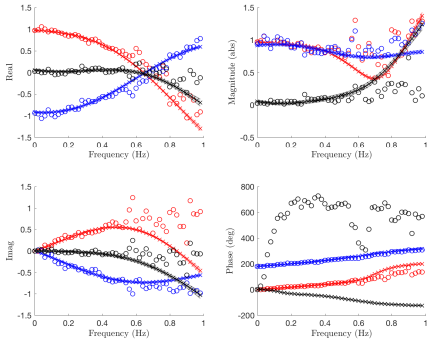
(c) No preview (sumsines)



(d) 1s preview (sumsines)

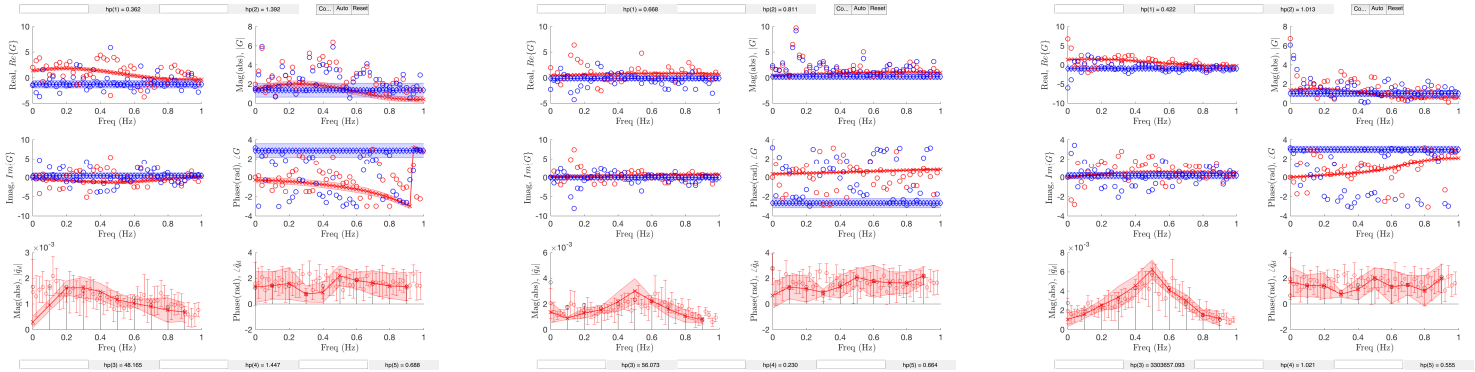


(e) 0.5s preview (sumsines)



(f) No preview (sumsines)

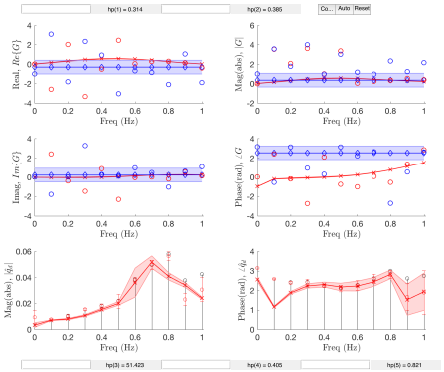
Figure B.18: Comparison of MISO Forward models $(q_d, e) \to r_h$ using sumsines/sweptsines training data. (—x—): $q_d \to r_h$, (g_1); (—x—): $e \to r_h$, (g_2), (—x—): $(g_1 + g_2)$, (●): Least-square estimate $q_d \to r_h$, (●): Least-square estimate $e \to r_h$, (—x—): $(g_1 + g_2)$, (●): Least-square estimate $g_1 + g_2$.



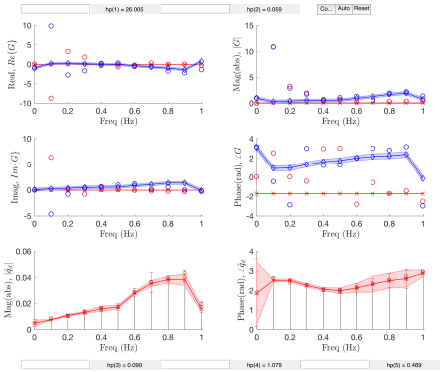
(a) 1s preview (sumsines)

(b) 0.5s preview (sumsines)

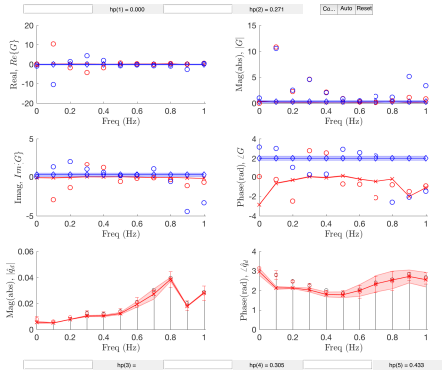
(c) No preview (sumsines)



(d) 1s preview (sweptsines)



(e) 0.5s preview (sweptsines)



(f) No preview (sweptsines)

Figure B.19: Comparison of MISO Inverse models $(r_h, q) \to e$ using sumsines/sweptsines training data. (—x—): $r_h \to e$ (inv); (—o—): $q \to e$ (inv), (●): Least-square estimate $r_h \to e$, (●): Least-square estimate $q \to e$.

The scattering effect of the LSQ fits is because the underlying functions are not well defined. Consider the forward modeling equation for $(q_d, e) \to r_h$,

$$r_h(\omega) = g_1(\omega)q_d(\omega) + g_2(\omega)e(\omega) \quad e(\omega) = \frac{1}{g_1(\omega) + g_2(\omega)}r_h(\omega) + \frac{-g_1(\omega)}{g_1(\omega) + g_2(\omega)}q(\omega) \tag{B.2}$$

but, we see that $g_1(\omega) + g_2(\omega) \approx 0$ for most frequencies $\omega \in [0, 1]$ Hz from Fig. B.18, resulting in large sensitivity of the inverse model estimates to noise.

BIBLIOGRAPHY

[1] Pieter Abbeel and Andrew Y Ng. "Apprenticeship learning via inverse reinforcement learning". In: *Proceedings of the twenty-first international conference on Machine learning*. ACM. 2004, p. 1.

[2] Jake K Aggarwal and Sangho Park. "Human motion: Modeling and recognition of actions and interactions". In: *3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004. Proceedings. 2nd International Symposium on*. IEEE. 2004, pp. 640–647.

[3] S Reza Ahmadzadeh, Roshni Kaushik, and Sonia Chernova. "Trajectory learning from demonstration with canal surfaces: A parameter-free approach". In: *Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on*. IEEE. 2016, pp. 544–549.

[4] Toshikazu Akita et al. "Hybrid system modeling of human driver in the vehicle following task". In: *SICE, 2007 Annual Conference*. IEEE. 2007, pp. 1122–1127.

[5] R Wade Allen and Henry R Jex. "An experimental investigation of compensatory and pursuit tracking displays with rate and acceleration control dynamics and a disturbance input(Pilot performance using compensatory and pursuit tracking displays with rate and acceleration control dynamics and disturbance input)". In: (1968).

[6] Seifemichael B Amsalu et al. "Driver intention estimation via discrete hidden Markov model". In: *Systems, Man, and Cybernetics (SMC), 2017 IEEE International Conference on*. IEEE. 2017, pp. 2712–2717.

[7] Suguru Arimoto, Sadao Kawamura, and Fumio Miyazaki. "Bettering operation of robots by learning". In: *Journal of Field Robotics* 1.2 (1984), pp. 123–140.

[8] A Terry Bahill and Jack D McDonald. "Smooth pursuit eye movements in response to predictable target motions". In: *Vision research* 23.12 (1983), pp. 1573–1583.

[9] N. Banka et al. "Iterative Machine Learning for Precision Trajectory Tracking with Series Elastic Actuators". In: *IEEE 15th International Workshop on Advanced Motion Control (AMC), Tokyo, Japan, March 9-11, 2018*. Accepted. 2018.

[10] Nathan Banka et al. "Iterative machine learning for precision trajectory tracking with series elastic actuators". In: *2018 IEEE 15th International Workshop on Advanced Motion Control (AMC)* (2018), pp. 234–239.

[11] George a. Bekey. "The Human Operator as a Sampled-Data System". In: *IRE Transactions on Human Factors in Electronics* HFE-3.2 (1962), pp. 511–536. ISSN: 0099-4561. DOI: 10.1109/THFE2.1962.4503341.

[12] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library.* " O'Reilly Media, Inc.", 2008.

[13] Stephen Butterworth. "On the theory of filter amplifiers". In: *Wireless Engineer* 7.6 (1930), pp. 536–541.

[14] D. De Carli et al. "Measuring intent in human-robot cooperative manipulation". In: *2009 IEEE International Workshop on Haptic Audio visual Environments and Games*. 2009, pp. 159–163.

[15] Ozkan Celik and Seniz Ertugrul. "Predictive human operator model to be utilized as a controller using linear, neuro-fuzzy and fuzzy-ARX modeling techniques". In: *Engineering Applications of Artificial Intelligence* 23.4 (2010), pp. 595–603.

[16] Chi-Tsong Chen. *Linear system theory and design*. Oxford University Press, Inc., 1998.

[17] Garrett M Clayton et al. "A review of feedforward control approaches in nanopositioning for high-speed SPM". In: *Journal of dynamic systems, measurement, and control* 131.6 (2009), p. 061101.

[18] David J Cole. "A path-following driver–vehicle model with neuromuscular dynamics, including measured and simulated responses to a step in steering angle overlay". In: *Vehicle System Dynamics* 50.4 (2012), pp. 573–596.

[19] R.G. Costello. "The Surge Model of the Well-Trained Human Operator in Simple Manual Control". In: *IEEE Transactions on Man-Machine Systems* 9.1 (1968), pp. 2–9. ISSN: 0536-1540. DOI: 10.1109/TMMS.1968.300028.

[20] Santosh Devasia. "Output tracking with nonhyperbolic and near nonhyperbolic internal dynamics: Helicopter hover control". In: *Journal of guidance, control, and dynamics* 20.3 (1997), pp. 573–580.

[21] Santosh Devasia, Degang Chen, and Brad Paden. "Nonlinear inversion-based output tracking". In: *Automatic Control, IEEE Transactions on* 41.7 (1996), pp. 930–942.

[22] Brad E Dicianno, Rory A Cooper, and John Coltellaro. "Joystick control for powered mobility: current state of technology and future directions". In: *Physical medicine and rehabilitation clinics of North America* 21.1 (2010), pp. 79–86.

[23] John M Dolan, Mark B Friedman, and Mark L Nagurka. "Dynamic and loaded impedance components in the maintenance of human arm posture". In: *IEEE Transactions on Systems, Man, and Cybernetics* 23.3 (1993), pp. 698–709.

[24] Frank M Drop et al. "Identification of the feedforward component in manual control with predictable target signals". In: *Cybernetics, IEEE Transactions on* 43.6 (2013), pp. 1936–1949.

[25] Sheikh Muhamad Hafiz Fahami et al. "Modeling and simulation of vehicle steer by wire system". In: *Humanities, Science and Engineering Research (SHUSER), 2012 IEEE Symposium on*. IEEE. 2012, pp. 765–770.

[26] Ronald A Hess. "Effects of time delays on systems subject to manual control". In: *Journal of Guidance, Control, and Dynamics* 7.4 (1984), pp. 416–421.

[27] Karin Hollerbach and H Kazerooni. "Modeling human arm movements constrained by robotic systems". In: *Proceedings of the Winter Annual Meeting of the American Society of Mechanical Engineering (Advances in Robotics)*. 1992, pp. 19–24.

[28] Auke Jan Ijspeert et al. "Dynamical movement primitives: learning attractor models for motor behaviors". In: *Neural computation* 25.2 (2013), pp. 328–373.

[29] K. E. Kaplan. "Improving inclusion segmentation task performance through human-intent based human-robot collaboration". In: *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 2016, pp. 623–624.

[30] Mitsuo Kawato. "Feedback-error-learning neural network for supervised motor learning". In: *Advanced neural computers* 6.3 (1990), pp. 365–372.

[31] D.L. Kleinman, S. Baron, and W.H. Levison. "An optimal control model of human response part I: Theory and validation". In: *Automatica* 6.3 (1970), pp. 357–369. ISSN: 00051098. DOI: 10.1016/0005-1098(70)90051-8.

[32] John Lataire and Tianshi Chen. "Transfer function and transient estimation by Gaussian process regression in the frequency domain". In: *Automatica* 72 (2016), pp. 217–229.

[33] Y. Long et al. "Online sparse Gaussian process based human motion intent learning for an electrically actuated lower extremity exoskeleton". In: *2017 International Conference on Rehabilitation Robotics (ICORR)*. 2017, pp. 919–924.

[34] Charles C Macadam. "Understanding and modeling the human driver". In: *Vehicle System Dynamics* 40.1-3 (2003), pp. 101–134.

[35] Adamantia S Mamma-Graham. "An intermittent predictive control approach to modelling sustained human motor control". PhD thesis. University of Glasgow, 2014.

[36] D McRuer and D H Weir. "Theory of manual vehicular control." In: *Ergonomics* 12.4 (1969), pp. 599–633. ISSN: 0536-1540. DOI: 10.1109/TMMS.1969.299930.

[37] DT McRuer and Dunstan Graham. "Pilot-vehicle control system analysis". In: *Guidance and control* (1964), pp. 603–621.

[38] D.T. McRuer and Ezra Krendel. *Human Pilot Dynamics in Compensatory systems*. 1965, p. 194. ISBN: AFFDL TR 65-15. URL: http://contrails.iit.edu/DigitalCollection/1965/AFFDLTR65-015.pdf.

[39] Duane T McRuer and Henry R Jex. "A review of quasi-linear pilot models". In: *IEEE Transactions on Human Factors in Electronics* 3 (1967), pp. 231–249.

[40] Duane T. McRuer and H.R. Hr Jex. "A Review of Quasi-Linear Pilot Models". In: *IEEE Transactions on Human Factors in Electronics* HFE-8.3 (1967), pp. 231–249. ISSN: 0096-249X. DOI: 10.1109/THFE.1967.234304. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1698271$\backslash$nhttp://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1698271.

[41] Duane T. McRuer and Ezra S. Krendel. *Mathematical Models of Human Pilot Behavior*. Tech. rep. January. 1974.

[42] Duane T. McRuer and Ezra S. Krendel. "The human operator as a servo system element". In: *Journal of the Franklin Institute* 267.5 (1959), pp. 381–403. ISSN: 00160032. DOI: 10.1016/0016-0032(59)90091-2. URL: http://linkinghub.elsevier.com/retrieve/pii/0016003259900912.

[43] Duane T. McRuer et al. "Human Pilot Dynamics in Compensatory Systems, Theory, Models, and Experiments with Controlled Element and Forcing Function Variations". In: *Control* (1965), p. 215.

[44] Bernard Michini et al. "Bayesian nonparametric reward learning from demonstration". In: *IEEE Transactions on Robotics* 31.2 (2015), pp. 369–386.

[45] Mohammad Najafi, Kim Adams, and Mahdi Tavakoli. "Robotic Learning From Demonstration of Therapist's Time-Varying Assistance to a Patient in Trajectory-Following Tasks". In: *International Conference on Rahabilitation Robotics (ICORR)* (2017), pp. 888–894.

[46] Duy Nguyen-Tuong et al. "Learning inverse dynamics: a comparison". In: *European Symposium on Artificial Neural Networks*. EPFL-CONF-175477. 2008.

[47] Gianluigi Pillonetto et al. "Kernel methods in system identification, machine learning and function estimation: A survey". In: *Automatica* 50.3 (2014), pp. 657–682.

[48] Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*. Vol. 1. MIT press Cambridge, 2006.

[49] Harish Chaandar Ravichandar and Ashwin P Dani. "Human Intention Inference Using Expectation-Maximization Algorithm With Online Model Learning." In: *IEEE Trans. Automation Science and Engineering* 14.2 (2017), pp. 855–868.

[50] Jonathan Realmuto, Rahul B Warrier, and Santosh Devasia. "Data-Inferred Personalized Human-Robot Models for Iterative Collaborative Output Tracking". In: *Journal of Intelligent & Robotic Systems* (2017), pp. 1–17.

[51] Jonathan Realmuto, Rahul B Warrier, and Santosh Devasia. "Iterative learning control for human-robot collaborative output tracking". In: *Mechatronic and Embedded Systems and Applications (MESA), 2016 12th IEEE/ASME International Conference on*. IEEE. 2016, pp. 1–6.

[52] Leonel Rozo et al. "Learning physical collaborative robot behaviors from human demonstrations". In: *IEEE Transactions on Robotics* 32.3 (2016), pp. 513–527.

[53] Xiaogang Ruan et al. "On-line adaptive control for inverted pendulum balancing based on feedback-error-learning". In: *Neurocomputing* 70.4 (2007), pp. 770–776.

[54] Stefan Schaal. "Is imitation learning the route to humanoid robots?" In: *Trends in cognitive sciences* 3.6 (1999), pp. 233–242.

[55] B Schèolkopf, Christopher JC Burges, and Alexander J Smola. *Advances in kernel methods support vector learning*. 1999.

[56] J.W. Senders. "The Human Operator as a Monitor and Controller of Multidegree of Freedom Systems". In: *IEEE Transactions on Human Factors in Electronics* HFE-5.1 (1964), pp. 2–5. ISSN: 0096-249X. DOI: 10.1109/THFE.1964.231647.

[57] Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*. MIT press, 1998.

[58] Szuchi Tien, Qingze Zou, and Santosh Devasia. "Iterative control of dynamics-coupling-caused errors in piezoscanners during high-speed AFM operation". In: *IEEE Transactions on Control Systems Technology* 13.6 (2005), pp. 921–931.

[59] Stephen Timoshenko. *Strength of material, Part1*. 1930.

[60] Jur Van Den Berg et al. "Superhuman performance of surgical tasks by robots using iterative learning from human-guided demonstrations". In: *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE. 2010, pp. 2074–2081.

[61] Gautham Vasan and Patrick M. Pilarski. "Learning from Demonstration: Teaching a Myoelectric Prosthesis with an Intact Limb via Reinforcement Learning". In: *International Conference on Rahabilitation Robotics (ICORR)* (2017), pp. 888–894.

[62] Sethu Vijayakumar, Aaron D'souza, and Stefan Schaal. "Incremental online learning in high dimensions". In: *Neural computation* 17.12 (2005), pp. 2602–2634.

[63] Steven Eric Vozar. "A framework for improving the speed and performance of teleoperated mobile manipulators". PhD thesis. US Army, 2013.

[64] Ronald E Walpole et al. *Probability and statistics for engineers and scientists*. Vol. 5. Macmillan New York, 1993.

[65] Zhikun Wang et al. "Probabilistic movement modeling for intention inference in human–robot interaction". In: *The International Journal of Robotics Research* 32.7 (2013), pp. 841–858.

[66] Rahul B Warrier and Santosh Devasia. "Data-based Iterative Human-in-the-loop Robot-Learning for Output Tracking". In: *IFAC-PapersOnLine* 50.1 (2017), pp. 12113–12118.

[67] Rahul B Warrier and Santosh Devasia. "Inferring Intent for Novice Human-in-the-Loop Iterative Learning Control". In: *IEEE Transactions on Control Systems Technology* (2016).

[68] Rahul B Warrier and Santosh Devasia. "Inverse control for inferring intent in novice human-in-the-loop iterative learning". In: *American Control Conference (ACC), 2016*. IEEE. 2016, pp. 2148–2154.

[69] Rahul B Warrier and Santosh Devasia. "Iterative learning from novice human demonstrations for output tracking". In: *IEEE Transactions on Human-Machine Systems* 46.4 (2016), pp. 510–521.

[70] Rahul B Warrier and Santosh Devasia. "Kernel-based human-dynamics inversion for precision robot motion-primitives". In: *Intelligent Robots and Systems, 2018 IEEE/RSJ International Conference on*. IEEE. 2018, (accepted, in print).

[71] Rahul B Warrier and Santosh Devasia. "Kernel-based intent estimation to improve robot learning from human-in-the-loop demonstrations". In: (2018), (submitting, in review).

[72] R. J. Wasicko, D. T. McRuer, and R. E. Magdaleno. *Human pilot dynamic response in single-loop systems with compensatory and pusuit displays*. Tech. rep. DTIC Document, 1967.

[73] Jarrett Webb and James Ashley. *Beginning Kinect Programming with the Microsoft Kinect SDK*. Apress, 2012.

[74] Bo Yu. "Interaction dynamics in oscillator and human-in-the-loop systems". PhD thesis. The University of Michigan, 2014.

[75] Zhibin Yu and Minho Lee. "Human motion based intent recognition using a deep dynamic neural model". In: *Robotics and Autonomous Systems* 71.SI (2015), 134–149.

[76] Qingze Zou and Santosh Devasia. "Preview-based stable-inversion for output tracking of linear systems". In: *Journal of dynamic systems, measurement, and control* 121.4 (1999), pp. 625–630.