UNIVERSITÄT BAYREUTH

An Optimal Control Approach to Human-Computer Interaction

Masterarbeit

von

Florian Fischer

FAKULTÄT FÜR MATHEMATIK, PHYSIK UND INFORMATIK
MATHEMATISCHES INSTITUT

Datum: 12. September 2019

Betreuung:
Prof. Dr. Lars Grüne
Prof. Dr. Jörg Müller

Abstract

In this thesis, we investigate the possibilities and limitations of the Linear-Quadratic Regulator (LQR) for modeling interactions between humans and computers, focusing on one of the most frequently used tasks: pointing with a mouse device.
We assume that users behave optimally with respect to some cost function, whose underlying assumptions are explained and motivated in detail. Moreover, the LQR framework allows us to model the underlying biomechanical apparatus through linear system dynamics. We therefore give an overview of both motion dynamics and optimization-based models and subsequently compose our proposed 2OL-LQR model as a combination of second-order lag dynamics, positional error penalization, and jerk minimization. The concrete design of the objective function is carried out in an iterative process, which makes it possible to address problems that have arisen during the implementation of different cost structures.
In order to determine the suitability of our model, we try to reproduce experimentally observed user trajectories as accurately as possible. For this purpose, our 2OL-LQR model is provided with some easily interpretable parameters, which are for a given user trajectory again optimized within an outer loop based on the least squares method. We show that our model approximates the observed user behavior significantly better than the conventional minimum-jerk and second-order models. In addition, the optimal parameters allow us to characterize users by properties and strategies and provide a deeper insight into the variability of mouse movements.
Finally, we present a computationally less-demanding method for testing the applicability of mathematical models to user trajectories as well as an extension of our proposed model that, in particular, takes reaction times into account.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

"There is no reason anyone would want a computer in their home." – Ken Olsen, co-founder of the *Digital Equipment Corporation* and himself owner of a personal computer at home at that time, hardly could have been more wrong when he spoke in 1977 about the future of his industry. Computers, notebooks, game consoles, tablets, smartphones, smartwatches, smartglasses, smart vending machines, and probably soon brain-computer interfaces: There is an ever-increasing number of machines that make everyday life easier for us. According to the *Global Digital Report 2019* [15], there are 5.112 billion unique mobile users worldwide nowadays, that is, two thirds of the world's population. Moreover, the current growth rate is still 2%. However, not only the number of users but also their usage behavior is taking on astonishing dimensions: In the second and third quarter of 2018, the average internet user spent 6 hours and 42 minutes online each day. In countries like Brazil or the Philippines it was even well over nine hours a day.
The number of mouse clicks, screen touches, or words interchanged with Alexa or Siri increases immensely. Even though these interactions with devices and artificial intelligence have become self-evident, a scientific understanding of the complex processes of our nervous and muscular systems is essential. Since the devices are supposed to assist us in various situations, a clear, simple, and – above all – intuitive use is extremely important. The **human-computer interaction (HCI)** research area, whose origins date back to the early 1980s, has set itself precisely this task.

However, while most improvements in this area result from observations and experiments, it is our aim to classify and analyze fundamental properties of the interaction between humans and machines using suitable mathematical models. For this reason, our methods and models essentially stem from **control theory**, which emerged as an engineering discipline and today is assigned to applied mathematics. The origins of this field of research reach back to the irrigation systems in ancient Mesopotamia and the aqueducts of the ancient Romans, which were based on controlling the water level [21]. However, the first comprehensive mathematical description of the control of relatively complex systems took place towards the end of the 19th century in the course of the further development of the steam engine. The associated

work “On Governors” [41] published by James Clerk Maxwell in 1868 counts as the birth of control theory as we know it today. Moreover, the research of manned flight, culminating in December 1903 with the Wright brothers’ first successful tests lasting about one minute, was based on the use of control models that were groundbreaking for the further development in this area [54].
Since then control theory has become an integral part of our everyday lives: Whether in heating systems, refrigerators, toilets, cars, spacecrafts, robots, or more abstract systems such as the sustainable use of natural resources – methods of control theory have become indispensable and are constantly adapted to countless new fields.
All these areas of application have one fundamental aspiration in common: They wish to influence a “system” in one way or another in order to reach a specific objective. In the above mentioned examples these objectives are, e.g., constant water level, constant operating speed of the steam engine, steady flight altitude, smooth and precise movements of robotic arms, efficient chemical reactions in polymer electrolyte membrane (PEM) fuel cells in order to extend their service life [34], or the long-term preservation of energy sources and valuable raw materials.

In this thesis, we focus on the control of a device that is fundamental for HCI: the **mouse pointer**.
In Chapter 2, we give a brief insight into the basic concepts and statements of modern control theory such as **stability** and **controllability**, first in the time-continuous, then in the time-discrete case. An overview of the most important models of human motor control, including quantitative models such as **Fitts’ law** as well as dynamic models such as **2OL-Eq**, can be found in Chapter 3. In addition, an introduction to **optimal control problems** is given, together with a first optimization-based model, the **Minimum-Jerk (MinJerk) model**. Section 4.1 contains a more mathematical part dealing with optimal control in **linear-quadratic problems**, while in Section 4.2 we apply this scheme to so-called **pointing tasks** by making suitable assumptions about user dynamics and defining reasonable objectives. The proposed **2OL-LQR model** is then solved analytically, and in Section 4.3 generalized to arbitrary dimensions and **via-point tasks**.
Finally, we use this model to reproduce experimentally observed user trajectories. Both the used data set and the complete **2OL-LQR algorithm**, which makes use of the method of least squares, are described in Chapter 5. In Chapter 6, the suggested variant of our algorithm is motivated by an iterative design process and applied to a variety of user trajectories. The results are presented and compared to those of 2OL-Eq and MinJerk. An extension of our algorithm, developed specifically for trajectories including reaction times, is introduced in Chapter 7, and the final conclusion is to be found in Chapter 8.

Chapter 2

Control Theoretic Basics

In this chapter, our aim is to introduce and explain the basic concept of control theory. The following statements are mainly based on the standard work “Mathematical Control Theory” by Eduardo D. Sontag [66], “Linear Optimal Control Systems” by Huibert Kwakernaak and Raphael Sivan [38], and the lecture notes “Mathematische Kontrolltheorie” (in German) by Lars Grüne [30]. Since all control models we present throughout this thesis are based on linear system dynamics, we restrict our demonstration of the fundamental concept of control theory to the linear case.

2.1 Continuous Time

We first assume an (uncontrolled) system given by the linear and autonomous ordinary differential equation

$$\dot{x}(t) = Ax(t), \quad t \in \mathbb{R}. \tag{2.1a}$$

Here, $x(t) \in \mathbb{R}^l$ ($l \in \mathbb{N}$) denotes the **state** of the system at time t, incorporating all relevant information about the system we wish to learn (and also affect later), and $A \in \mathbb{R}^{l \times l}$ is a matrix determining the development of the state x, i.e., the continuous state function $x\colon \mathbb{R} \longrightarrow \mathbb{R}^l$. Given an initial state $x_0 \in \mathbb{R}^l$ at an initial time $t_0 \in \mathbb{R}$, i.e.,

$$x(t_0) = x_0, \tag{2.1b}$$

the system (2.1a) has a unique solution, which can be explicitly specified:

Theorem 2.1. *[38, Theorem 1.1 and Theorem 1.4]*
The unique solution to the ***initial value problem*** (2.1) *with fixed* $t_0 \in \mathbb{R}$ *and* $x_0 \in \mathbb{R}^l$ *is given by*

$$\begin{aligned} x\colon \mathbb{R} &\longrightarrow \mathbb{R}^l \\ t &\mapsto e^{A(t-t_0)}x_0 \end{aligned} \tag{2.2}$$

and denoted by $x(t; t_0, x_0)$ *in the following.*

In particular, given the initial condition (2.1b), the states $x(t;t_0,x_0)$ are uniquely determined not only for all future times $t > t_0$, but also for all past times $t < t_0$. Such systems are thus called *time-reversible.* However, since we are usually interested in the development of x over a future period of time starting from t_0, from now on we will restrict x to either a bounded interval $I := [t_0, t_f]$ with final time $t_f \in \mathbb{R}$, $t_f > t_0$, or a half-bounded interval $I := [t_0, \infty[$ (in this case we use $t_f := \infty$), i.e., we rather mean $x|_I$ when writing x in the following.

In many fields of research, the development of the considered systems is typically not predetermined, but can be affected by some "controller". To take this scope of action into account, we define a **control function** $u\colon I \longrightarrow \mathbb{R}^m$ ($m \in \mathbb{N}$). We only require u to be **piecewise continuous**, i.e.,

$$\begin{gathered}\forall\, a, b \in \mathbb{R} \text{ with } [a,b] \subset I \;\; \exists\, n \in \mathbb{N},\; a = t_0 < t_1 < \ldots < t_n = b\colon \\ u|_{]t_{i-1},t_i[} \text{ is continuous and bounded } \forall\, i \in \{1,\ldots,n\},\end{gathered} \tag{2.3}$$

and denote the set of functions $u\colon I \longrightarrow \mathbb{R}^m$ satisfying (2.3) by $\mathcal{U}$. The resulting **linear control system** is then given by

$$\dot{x}(t) = Ax(t) + Bu(t), \quad t \in I, \tag{2.4a}$$

where the matrix $A \in \mathbb{R}^{l\times l}$ describes how the system evolves without applied control and the matrix $B \in \mathbb{R}^{l\times m}$ incorporates the additional effect of the control $u(t)$.
Together with the initial condition

$$x(t_0) = x_0, \tag{2.4b}$$

analogously to the uncontrolled case, we have a unique solution:

Theorem 2.2. *[38, Theorems 1.3 and 1.4]*
The unique continuous solution to the initial value problem (2.4) *for fixed* $t_0 \in I$, $x_0 \in \mathbb{R}^l$, *and* $u \in \mathcal{U}$ *is given by*

$$\begin{aligned} x\colon I &\longrightarrow \mathbb{R}^l \\ t &\mapsto e^{A(t-t_0)}x_0 + \int_{t_0}^{t} e^{A(t-s)}Bu(s)ds. \end{aligned} \tag{2.5}$$

This ***solution trajectory*** *is denoted by* $x(t;t_0,x_0,u)$ *in the following.*

The fact that x needs to be assumed continuous to guarantee uniqueness results from the permission of discontinuities in the control function u. Moreover, analogously to Theorem 2.1, a unique continuous extension of x to $\mathbb{R}$ exists for any given piecewise continuous control function $u\colon \mathbb{R} \longrightarrow \mathbb{R}^m$.
There are a few interesting properties of the solution formula (2.5) we would like to examine:

First, the second, control-dependent term appears in addition to the solution of the uncontrolled initial value problem (2.1) given by (2.2), and is notably independent of the initial value x_0. In other words, (2.5) can always be divided into the solutions for two special cases of the linear control system (2.4): Setting $u \equiv 0$ naturally results in only the first term and setting $x_0 = 0$ yields exactly the second term. Thus, the system's own evolvement (depending on the initial value x_0) and the effect of the applied control can be considered separately.
Second, taking the solution $\hat{x} := x(\tau; t_0, x_0)$ of (2.4) at an arbitrary time $\tau \in [t_0, t_f[$ as initial state for the new initial value problem consisting of (2.1a) and $x(\tau) = \hat{x}$ instead of (2.1b) results in the same solution trajectory. This observation can be explained by the linearity of the integral and the exponential identity $e^y e^z = e^{y+z}$ applied to (2.5), and it shows that the solution is not only unique on I, but also that there are no "solution branches" along the way.
Third, the linear control system is not only time-reversible given t_0, x_0, and u, but the solution trajectories (2.5) are also independent of the initial time t_0 in the sense that replacing t_0 and t_f by $t_0 + \tau$ and $t_f + \tau$ with arbitrary $\tau \in \mathbb{R}$ in (2.4), i.e., in particular starting at the same initial state x_0 at another initial time $t_0 + \tau$, and applying the same control, i.e., $u(\cdot - \tau)$ rather than $u(\cdot)$, leads to the same solution trajectory (formally, with domain shifted by τ). This follows directly from the form of the solution formula (in particular, no absolute times t but only relative times $t - t_0$ are considered).
Hence, we can assume $t_0 = 0$ without loss of generality and denote the solution of (2.1) by $x(t; x_0) := x(t; 0, x_0)$ and the solution of (2.4) by $x(t; x_0, u) := x(t; 0, x_0, u)$ in the following.[1]

In the following, we want to apply this control framework to pointing tasks. Since the users aim to reach a specific target, it is of interest to us under which conditions this can be achieved within our linear control system (2.4). For this reason, we need a few definitions.

Definition 2.3 (Stability Concepts). *Let $f\colon \mathbb{R}^l \longrightarrow \mathbb{R}^l$ be Lipschitz continuous and consider the autonomous differential equation $\dot{x}(t) = f(x(t))$, $t \geq 0$, together with some initial condition $x(0) = x_0 \in \mathbb{R}^l$. Let $\|\cdot\|$ be an arbitrary[2] norm on $\mathbb{R}^l$.*

*(a) $x^* \in \mathbb{R}^l$ is called **equilibrium** of this differential equation $:\Longleftrightarrow f(x^*) = 0$.*

*(b) An equilibrium x^* is called **stable** $:\Longleftrightarrow \forall\, \epsilon > 0\ \exists\, \delta > 0\colon \|x(t; x_0) - x^*\| < \epsilon\ \forall\, t \geq 0$ holds for all solutions $x(\cdot; x_0)$ of the above initial value problem with $\|x_0 - x^*\| < \delta$.*

*(c) An equilibrium x^* is called **locally asymptotically stable** $:\Longleftrightarrow x^*$ is stable and $\exists\, \delta > 0$ such that $\lim_{t\to\infty} x(t; x_0) = x^*$ holds for all $x_0 \in \mathbb{R}^l$ with $\|x_0 - x^*\| < \delta$.*

*(d) An equilibrium x^* is called **asymptotically stable** $:\Longleftrightarrow x^*$ is stable and in addition $\lim_{t\to\infty} x(t; x_0) = x^*$ holds for all $x_0 \in \mathbb{R}^l$.*

[1]For the generalized case of non-linear differential equations or control systems we use the same notation.

[2]Note that on finite-dimensional vector spaces such as $\mathbb{R}^l$ all norms are *equivalent*, i.e., for any two norms $\|\cdot\|_\alpha$, $\|\cdot\|_\beta$ there exist positive constants $c_1, c_2 > 0$ such that $c_1\|x\|_\alpha \leq \|x\|_\beta \leq c_2\|x\|_\alpha$ holds for any $x \in \mathbb{R}^l$. In particular, all stability definitions are independent of the norm used.

Equilibria are therefore exactly the states in which the system remains after reaching them; in particular, the solution to the initial value problem with $x(0) = x^*$, where x^* denotes an equilibrium of the respective differential equation, is $x \equiv x^*$. Note that in our linear case, i.e., $f(x) := Ax$, the set of equilibria is exactly the kernel of A. In particular, $x^* = 0$ is always an equilibrium.
While stability claims that solutions remain arbitrarily close to the equilibrium for all times if only the initial value is close enough, asymptotic stability additionally implies convergence towards this equilibrium (in the local version again only for initial values in a sufficiently small neighborhood of the equilibrium).[3] Fortunately, for the linear case there is an easily verifiable criterion for stability:

Lemma 2.4. *[38, Theorems 1.13 and 1.14]*
The equilibria of the autonomous linear differential equation (2.1a) *are*

(a) stable $\iff \mathrm{Re}(\lambda_i) \leq 0$ *for all eigenvalues* λ_i *of* A*, and for all eigenvalues with* $\mathrm{Re}(\lambda_i) = 0$ *the algebraic and the geometric multiplicity coincide, and*

(b) asymptotically stable $\iff \mathrm{Re}(\lambda_i) < 0$ *for all eigenvalues* λ_i *of* A.

In the latter case, we not only have asymptotic stability, but in fact ***exponential stability****, i.e.,*

$$\exists\, c, \sigma > 0\colon \quad \|x(t; x_0)\| < ce^{-\sigma t}\|x_0\| \quad \forall\, t \geq 0,\ x_0 \in \mathbb{R}^l. \tag{2.6}$$

In particular, any equilibrium $x^* \neq 0$ can at most be stable since 0 must be an eigenvalue of A for its existence. Conversely, if the real parts of all eigenvalues of A have negative sign, the solution of (2.1) thus converges exponentially fast to $x^* = 0$ for *any* initial state $x_0 \in \mathbb{R}^l$. The fact that this is not the case for most applications is exactly the reason for introducing the control option in (2.4): Even though the system might not reach its steady-state by itself, the controller has the opportunity to influence the states appropriately.
The most far-reaching impact a controller might have can be formalized as follows:

Definition 2.5 (Complete Controllability)**.** *Consider the linear control system* (2.4a) *with* $I := [0, \infty[$.

(a) The ***reachable set at time*** $t \geq 0$ *is given by* $\mathcal{R}^t := \{x(t; 0, u) \mid u \in \mathcal{U}\}$.

(b) The system is ***completely controllable in time*** $t > 0$ *if* $\mathcal{R}^t = \mathbb{R}^l$ *holds.*

(c) The system is ***completely controllable*** *if* $\mathcal{R}^t = \mathbb{R}^l$ *holds for all times* $t > 0$.

[3]Note that asymptotic stability only makes sense for at least half-bounded domains, i.e., $I := [t_0, \infty[$.

It can be shown that the reachable sets $\mathcal{R}^t$ coincide for all times $t > 0$, i.e., the states that can be reached by starting at $t_0 = 0$ in $x_0 = 0$ and applying some admissible control $u \in \mathcal{U}$ up to time $t > 0$ can also be reached at any other positive time $t > 0$ by applying another control $\tilde{u} \in \mathcal{U}$. This immediately implies the equivalence of (b) and (c). Moreover, if complete controllability holds, the solution formula (2.5) suggests that the analogous reachable sets for starting at time $t_0 = 0$ in any $x_0 \in \mathbb{R}^l$, $\mathcal{R}^t_{x_0} := \{x(t; x_0, u) \mid u \in \mathcal{U}\}$ for $t \geq 0$, satisfy

$$\mathcal{R}^t_{x_0} \overset{(2.5)}{=} x(t; x_0, 0) + \mathcal{R}^t = x(t; x_0, 0) + \mathbb{R}^l = \mathbb{R}^l, \quad t > 0. \tag{2.7}$$

This means that in case of complete controllability any state $x \in \mathbb{R}^l$ can be reached from any initial state $x_0 \in \mathbb{R}^l$ in arbitrarily short time by applying an appropriate control $u \in \mathcal{U}$. The following theorem provides an easy-to-check criterion for complete controllability of a system.

Theorem 2.6 (Kalman-Criterion). *[66, Definition 3.3.1 and Lemma 3.3.2]*
The linear control system (2.4a) *is completely controllable if and only if*

$$\operatorname{rank}(\begin{bmatrix} B & AB & \cdots & A^{l-1}B \end{bmatrix}) = l. \tag{2.8}$$

However, in many cases complete controllability might be too demanding. Instead, one might only be interested in being able to *stabilize* the system by applying appropriate controls in the sense that the solution of the resulting linear control system $\dot{x} = Ax + Bu$ converges to an equilibrium x^* of this control system[4] which is possibly not stable for the uncontrolled system $\dot{x} = Ax$ (or not even an equilibrium).

For that purpose, it is reasonable to regard the control function u not as a function of time as before, but as a function of state, i.e., we consider $u(x(t))$ rather than $u(t)$.
The question whether controls are planned ahead for the whole time span or event-dependently chosen at each time step is the basis for one of the most fundamental differentiation of control theoretic models [17]. In the former case, the whole "control plan" is determined a priori by the control function $u \colon I \longrightarrow \mathbb{R}^m$ mapping any considered time $t \in I$ to the control applied at this time. However, solving the control system numerically might already lead to slightly different states x than expected analytically due to rounding errors. A control function u that is explicitly time- and not state-dependent has not taken these deviations into account and thus cannot compensate appropriately, which might lead to a propagation of error. In contrast to such **open-loop** control models, **closed-loop** models assume that some information about the current state x is fed back to the controller, which can condition its control on this observation. The function that maps any state[5] x to the corresponding control u, i.e., $F \colon \mathbb{R}^l \longrightarrow \mathbb{R}^m,\ x \mapsto u := F(x)$, is called **feedback function**. Another advantage of such

[4]A more precise formulation of this concept of equilibrium is given in Definition 2.7.

[5]In the more general case of *imperfect observation*, F only maps an observable *output* $y(x)$ that depends on the unobservable state x to the corresponding control u.

models is thus, besides the ability to react to external deviations, the generally applicable control instruction that only depends on the current state and not on the realized point in time.

For these reasons we assume our model to be closed-loop. Consistent with our linear control system we additionally assume the feedback function F to be also linear, i.e., $u(t) = Fx(t)$ holds for all $t \in I$ and some matrix $F \in \mathbb{R}^{m\times l}$.
Under these assumptions the stabilization problem can be described as follows:

Definition 2.7. *The* ***(asymptotic) stabilization problem*** *for the linear control system* (2.4a) *is to find a linear feedback function/matrix F, so that the equilibria of the resulting* ***closed-loop system***

$$\dot{x}(t) = (A + BF)x(t) \tag{2.9}$$

are (asymptotically) stable.

As we will see, complete controllability is not necessary for the solvability of this problem. To this end, we need another representation of the linear control system (2.4a):

Lemma 2.8. *[66, Lemmas 3.3.3 and 3.3.4]*
Let the system (2.4a) *be not completely controllable, i.e.,* $\sigma := \operatorname{rank}(\begin{bmatrix} B & AB & \cdots & A^{l-1}B\end{bmatrix}) < l$. *Then $T \in \mathbb{R}^{l\times l}$ invertible exists, so that $\hat{A} := T^{-1}AT$ and $\hat{B} := T^{-1}B$ are of the form*

$$\hat{A} = \begin{bmatrix} A_1 & A_2 \\ 0 & A_3 \end{bmatrix}, \quad \hat{B} = \begin{bmatrix} B_1 \\ 0 \end{bmatrix}, \tag{2.10}$$

with $A_1 \in \mathbb{R}^{\sigma\times\sigma}$, $A_2 \in \mathbb{R}^{\sigma\times(l-\sigma)}$, $A_3 \in \mathbb{R}^{(l-\sigma)\times(l-\sigma)}$, and $B_1 \in \mathbb{R}^{\sigma\times m}$, and the system $\dot{y} = A_1y + B_1u$ is completely controllable.

In particular, coordinate transformation with T, i.e., $z(t) := Tx(t)$, yields the transformed linear control system

$$\dot{z}_1 = A_1z_1 + A_2z_2 + B_1u \tag{2.11a}$$
$$\dot{z}_2 = A_3z_2 \tag{2.11b}$$

with $z_1 \in \mathbb{R}^{\sigma}$, $z_2 \in \mathbb{R}^{l-\sigma}$, where z_2 is the state part that cannot be influenced by the controller.
With this lemma, we can state the main result of this chapter:

Theorem 2.9 (Pole-Shifting Theorem)**.** *[66, Theorem 13 and Corollary 5.8.8]*
The asymptotic stabilization problem for (2.4a) *is solvable if and only if one of the following conditions holds:*

(a) (2.4a) *is completely controllable.*

(b) Considering the transformed linear system (2.11)*, all eigenvalues of* A_3 *have negative real part.*

The stabilization problem for (2.4a) *is solvable if and only if (a) or the following condition holds:*

(c) Considering the transformed linear system (2.11)*, all eigenvalues of* A_3 *have non-positive real part and for all eigenvalues with real part equal to zero the algebraic and the geometric multiplicity coincide.*

In summary: If the linear control system (2.4a) is completely controllable, i.e., every state $x \in \mathbb{R}^l$ can be reached (even in arbitrarily short time) by applying appropriate controls $u(t)$, a feedback matrix F can be found so that $x^* = 0$ is asymptotically stable for $\dot{x} = (A+BF)x$. However, complete controllability is sufficient, but not necessary. With the above defined transformation of systems that are not completely controllable into their controllable part (2.11a) and their uncontrollable part (2.11b), it actually suffices that $z_2^* = 0$ is asymptotically stable for (2.11b) respectively that the uncontrollable part z_2^* of every equilibrium x^* of $\dot{x} = (A + BF)x$ is stable for (2.11b), which, according to Lemma 2.4, is equivalent to condition *(b)* respectively to condition *(c)* in Theorem 2.9.

While all previous statements related to linear systems in continuous time, we transfer them to time-discrete systems in the following section. The reason for this is the discrete form of the experimental data we aim to simulate later, which suggests the usage of models in discrete time.

2.2 Discrete Time

The idea behind time-discrete control systems is to sample the system's output at a given sequence of usually equidistant times. Here, the constant distance $h > 0$ between two consecutive times is called **sampling time**.
To be precise, the considered[6] time span $I = [0, t_f]$ is divided into $N-1$ subintervals with boundary points $\{(n-1)h \mid n \in \{1, \dots, N\}\}$, where $N := \max\{n \in \mathbb{N} \mid (n-1)h \leq t_f\}$.[7]
These boundary points are exactly the times at which we want to evaluate the state of the system.
Furthermore, we assume that the controls are constant during those intervals and denote the resulting **control sequence** by $(u_n)_{n\in\{1,\dots,N-1\}}$, i.e., $u_n = u|_{[(n-1)h,nh[}$ applies for all $n \in \{1, \dots, N-1\}$.[8] This is an important assumption for the subsequent considerations,

[6]The half-bounded case $I := [0, \infty[$ works analogously with boundary points $\{(n-1)h \mid n \in \mathbb{N}\}$.

[7]For reasons of simplification, we assume that $\frac{t_f}{h} \in \mathbb{N}$, i.e., $(N-1)h = t_f$ holds. This can be ensured through either choosing h appropriately or restricting I to $[0, (N-1)h]$.

[8]Note that the control values at the boundary points are irrelevant in the time-continuous variant since they are null-sets and thus can be chosen arbitrarily. Here, we decided to concatenate u right-sided continuous.

which implies that changes in control can only be made each h time units. However, they may still depend on the immediately observed state at the respective time.
Returning to linear control systems, Theorem 2.2 and the second subsequent remark instantly imply the following time-discrete update formula:

Corollary 2.10. *The time-continuous linear control system with initial condition*

$$\begin{aligned} \dot{x}(t) = \tilde{A}x(t) + \tilde{B}u(t), \quad t \in I := [0, t_f]\,, \\ x(0) = x_0, \end{aligned} \tag{2.12}$$

with $\tilde{A} \in \mathbb{R}^{l \times l}$, $\tilde{B} \in \mathbb{R}^{l \times m}$, $x(t) \in \mathbb{R}^l$ and $u(t) \in \mathbb{R}^m$ for all $t \in I$, $t_f \in \,]0, \infty[$, and $x_0 \in \mathbb{R}^l$ can be transferred into the respective ***time-discrete linear control system with sampling time*** *$h > 0$ and initial condition*

$$\begin{aligned} x_{n+1} = Ax_n + Bu_n, \quad n \in \{1, \dots, N-1\}, \\ x_1 = \bar{x}_1, \end{aligned} \tag{2.13}$$

with

$$A := e^{\tilde{A}h}, \quad B := \int_0^h e^{\tilde{A}(h-\tau)}\tilde{B}\; d\tau, \tag{2.14}$$

where $N := \frac{t_f}{h} + 1 \in \mathbb{N}$.
This means that for any consistent initial conditions, i.e., $\bar{x}_1 = x_0$, and any control function u that is constant on each sampling interval together with the corresponding time-discrete control sequence $u_n := u|_{[(n-1)h, nh[}$ for all $n \in \{1, \dots, N-1\}$, the unique solutions of (2.12) *and* (2.13)*, which we denote by $x(t; x_0, u)$ and $(x_n^{\bar{x}_1, (u_1, \dots, u_{n-1})})_{n \in \{1, \dots, N\}}$, coincide, i.e.,*

$$x((n-1)h; x_0, u) = x_n^{\bar{x}_1, (u_1, \dots, u_{n-1})} \tag{2.15}$$

holds for all $n \in \{1, \dots, N\}$.

With this, we can now investigate how the above statements can be transferred to time-discrete systems. In order to avoid confusion, we use the notation of Corollary 2.10 in the following, i.e., the matrices of time-continuous systems are denoted by $\tilde{A}$ and $\tilde{B}$ (although denoted by A and B in previous equations for reasons of simplicity) while A and B are now reserved for the time-discrete variants.

Remark 2.11.

(a) According to the first remark after Theorem 2.2, setting $u \equiv 0$ in the solution formula (2.5) *of the time-continuous linear control system yields the solution of the time-continuous uncontrolled system* (2.1)*. Analogously, the uncontrolled system in discrete time is given by* (2.13) *with $u_n = 0$ for all $n \in \{1, \dots, N-1\}$.*

(b) *Similar stability criteria as in Lemma 2.4 can be found for the* ***linear difference equation*** $x_{n+1} = Ax_n$, $n \in \mathbb{N}$:
An ***equilibrium*** $x^* \in \mathbb{R}$ *of this equation, i.e.,* $Ax^* = x^*$, *is*

- *stable* $\iff |\lambda_i| \leq 1$ *for all eigenvalues* λ_i *of* A*, and for all eigenvalues with* $|\lambda_i| = 1$ *the algebraic and the geometric multiplicity coincide, and*
- *asymptotically stable* $\iff |\lambda_i| < 1$ *for all eigenvalues* λ_i *of* A.

This follows directly from (2.14) *and some matrix exponential properties.*
Note that analogously to the time-continuous case an equilibrium $x^* \neq 0$ *can be stable at most since it needs to be an eigenvector of* A *corresponding to the eigenvalue* 1.

(c) *Theorem 2.6 also holds for the time-discrete case, but with* ***time-discrete complete controllability*** *in the sense that for the reachable sets at time steps* $n \in \{0, \dots, N-1\}$, $\mathcal{R}^{(n-1)h} := \{x_n^{0,u} \mid u = (u_k)_{k=1,\dots,n-1} \subset \mathbb{R}^m\}$, *the equality* $\mathcal{R}^{(n-1)h} = \mathbb{R}^l$ *only needs to apply for* $n > l$ *in general. In other words, even with complete controllability of a time-discrete system it might take up to* l *time steps, with* l *denoting the state dimension of the system, to reach any arbitrary state* $x \in \mathbb{R}^l$.

(d) *Lemma 2.8 can be directly transferred to time-discrete systems of the form* (2.13). *In particular, coordinate transformation with* T*, i.e.,* $z^n := Tx_n$*, yields the transformed linear control system*

$$z_1^{n+1} = A_1 z_1^n + A_2 z_2^n + B_1 u_n \tag{2.16a}$$
$$z_2^{n+1} = A_3 z_2^n \tag{2.16b}$$

with $z_1^n \in \mathbb{R}^\sigma$, $z_2^n \in \mathbb{R}^{l-\sigma}$ *for all* $n \in \{1, \dots, N\}$*, where* z_2^n *are the parts that cannot be affected by the controller.*

(e) *Hence, the time-discrete version of Theorem 2.9 reads as follows: The* ***time-discrete (asymptotic) stabilization problem*** *for* (2.13)*, i.e., to find a matrix* $F \in \mathbb{R}^{m \times l}$*, so that the equilibria* x^* *of the resulting* ***time-discrete closed-loop system***

$$x_{n+1} = (A + BF)x_n, \tag{2.17}$$

are (asymptotically) stable, is solvable if and only if either time-discrete complete controllability holds or, considering the transformed linear system (2.16)*, all eigenvalues* λ_i *of* A_3 *satisfy* $|\lambda_i| \leq 1$ *(for stability) respective* $|\lambda_i| < 1$ *(for asymptotic stability).*

We would like to close this section with a simple example to illustrate the close relationship between time-continuous and time-discrete linear control systems, before we continue with typical models from the field of human motor control in the next chapter.

Example 2.12. *Consider the time-continuous linear control system*

$$\dot{x}(t) = \tilde{A}x(t) + \tilde{B}u(t), \quad t \in I := [0, \infty[\,, \tag{2.18}$$

with $x\colon I \longrightarrow \mathbb{R}^2$ *and* $u\colon I \longrightarrow \mathbb{R}$, *where*

$$\tilde{A} := \begin{bmatrix} -1 & 0 \\ 0 & 1-\rho \end{bmatrix}, \quad \tilde{B} := \begin{bmatrix} -1 \\ -\epsilon \end{bmatrix}, \tag{2.19}$$

for given $\rho, \epsilon \in \mathbb{R}$.

According to Corollary 2.10, the respective time-discrete linear control system with sampling time $h > 0$ *is given by*

$$x_{n+1} = Ax_n + Bu_n, \quad n \in \{1, \ldots, N-1\}, \tag{2.20}$$

with

$$A = \begin{bmatrix} e^{-h} & 0 \\ 0 & e^{(1-\rho)h} \end{bmatrix}, \quad B = \begin{cases} \begin{bmatrix} e^{-h} - 1 \\ -\epsilon h \end{bmatrix}, & \text{if } \rho = 1. \\ \begin{bmatrix} e^{-h} - 1 \\ \frac{\epsilon}{1-\rho}(1 - e^{(1-\rho)h}) \end{bmatrix}, & \text{if } \rho \neq 1. \end{cases} \tag{2.21}$$

The following therefore holds for $x^* = [0,0]^\top$:

$$\begin{aligned} & x^* \text{ is asymptotically stable equilibrium of } \dot{x}(t) = \tilde{A}x(t) \\ \overset{2.4}{\iff} & \operatorname{Re}(\tilde{\lambda}_i) < 0 \text{ for all eigenvalues } \tilde{\lambda}_i \text{ of } \tilde{A} \\ \iff & \rho > 1 \\ \iff & |\lambda_i| < 1 \text{ for all eigenvalues } \lambda_i \text{ of } A \\ \overset{2.11(b)}{\iff} & x^* \text{ is asymptotically stable equilibrium of } x_{n+1} = Ax_n. \end{aligned} \tag{2.22}$$

Analogous statements apply to stability if "<" is replaced by "≤".
In particular, both $\operatorname{Re}(\tilde{\lambda}_2) = 0$ *and* $|\lambda_2| = 1$ *hold for the eigenvalue* $\tilde{\lambda}_2 = 1 - \rho$ *of* $\tilde{A}$ *and the eigenvalue* $\lambda_2 = e^{(1-\rho)h}$ *of* A, *respectively, if and only if* $\rho = 1$ *holds (note that* $\operatorname{Re}(\tilde{\lambda}_1) < 0$ *and* $|\lambda_1| < 1$ *always apply). In this case, the eigenvectors* $x^* = [0, c]^\top$, $c \in \mathbb{R}$, *of both* $\tilde{A}$ *(corresponding to* $\tilde{\lambda}_2 = 0$*) and* A *(corresponding to* $\lambda_2 = 1$*) are, in addition to* $x^* = [0,0]^\top$, *stable equilibria of both uncontrolled systems.*
Conversely, if $\rho < 1$ *applies, neither of these two properties is fulfilled, i.e., both systems are unstable.*

For statements on complete controllability, we need to consider

$$\begin{bmatrix}\tilde{B} & \tilde{A}\tilde{B}\end{bmatrix} = \begin{bmatrix}-1 & 1 \\ -\epsilon & \epsilon(\rho-1)\end{bmatrix} \tag{2.23}$$

respectively

$$\begin{bmatrix}B & AB\end{bmatrix} = \begin{cases}\begin{bmatrix}e^{-h}-1 & e^{-h}(e^{-h}-1) \\ -\epsilon h & -\epsilon h e^{(1-\rho)h}\end{bmatrix}, & \text{if } \rho = 1. \\ \begin{bmatrix}e^{-h}-1 & e^{-h}(e^{-h}-1) \\ \frac{\epsilon}{1-\rho}(1-e^{(1-\rho)h}) & \frac{\epsilon}{1-\rho}e^{(1-\rho)h}(1-e^{(1-\rho)h})\end{bmatrix}, & \text{if } \rho \neq 1.\end{cases} \tag{2.24}$$

According to the Kalman-Criterion (Theorem 2.6), the following applies:

$$\begin{aligned}(2.18) \text{ is completely controllable} &\overset{2.6}{\iff} \operatorname{rank}(\begin{bmatrix}\tilde{B} & \tilde{A}\tilde{B}\end{bmatrix}) = 2 \\ &\overset{(*)}{\iff} \epsilon \neq \epsilon(\rho-1) \iff \epsilon \neq 0 \wedge \rho \neq 2 \\ &\overset{(*)}{\iff} \operatorname{rank}(\begin{bmatrix}B & AB\end{bmatrix}) = 2 \overset{2.6}{\iff} (2.20) \text{ is completely controllable.}\end{aligned} \tag{2.25}$$

Note that in our case with one-dimensional control u, $(*)$ already follows from the structure of $\tilde{A}$: Since $\tilde{A}$ (and thus also A) is diagonal, the two columns of $\begin{bmatrix}B & AB\end{bmatrix}$ respective $\begin{bmatrix}\tilde{B} & \tilde{A}\tilde{B}\end{bmatrix}$ are linearly dependent if and only if $\tilde{B}$ respective B has a zero entry ($\iff \epsilon = 0$) or $\tilde{A}$ respective A has the same diagonal entries ($\iff \rho = 2$).

Complete controllability is sufficient, but not necessary to solve the (asymptotic) stabilization problems for both systems:
If $\epsilon = 0$, $\tilde{A}$ and $\tilde{B}$ (respectively A and B) are already of the form (2.10) *with $A_3 = 1-\rho$ (respectively $A_3 = e^{(1-\rho)h}$) and it can easily be shown that any further coordinate transformation preserving this form cannot change the value of A_3, i.e., the asymptotic stabilization problem is solvable for* (2.18) *according to Theorem 2.9 and for* (2.20) *according to Remark 2.11(e) if and only if $\rho > 1$ applies (for the stabilization problem the same applies with $\rho \geq 1$).*
If $\rho = 2$, both (2.18) *and* (2.20) *can be transformed into a linear control system of the form* (2.11) *with $A_3 = -1$ respectively $A_3 = e^{-h}$, i.e., the (asymptotic) stabilization problem is solvable for both systems, which was to be expected since the respective uncontrolled systems were already asymptotically stable in this case according to* (2.22).

However, even if the time-discrete system is completely controllable, not every $x \in \mathbb{R}^2$ must be reachable within one step: Setting, e.g., $\rho := 3$, $\epsilon := 2$, $\bar{x}_1 := [0,0]^\top$, and $h := 0.2$ makes it impossible to obtain $x_2 := [1,1]^\top$ (i.e., $\mathcal{R}^h \neq \mathbb{R}^2$), since

$$x_2 = \begin{bmatrix}e^{-0.2}-1 \\ e^{-0.4}-1\end{bmatrix} u_1 \tag{2.26}$$

needs to hold for any arbitrary $u_1 \in \mathbb{R}$.

Chapter 3

Human Motor Control Models

In this chapter, we give a brief overview of the most important models of human motor control. Beginning with Fitts' law, we present typical closed-loop models with up to two integrators and distinguish these models from approaches based on an optimal control perspective.

3.1 Fitts' law

One of the most famous and, above all, basic models of human movements is **Fitts' law**. In the 1950s, Paul M. Fitts was the first to establish a relationship between the difficulty of an aimed movement and its duration, which was confirmed in various experiments [22]. Using conventional notation from information theory, he defined the *Index of Difficulty (ID)* of an arbitrary pointing task as

$$\mathrm{ID} := \log_2\left(\frac{2D}{W}\right), \tag{3.1}$$

where D denotes the Euclidean distance between the starting point and the center of some target box and W denotes the width of this target box.[1] This index can be interpreted as the *information content* that needs to be processed to complete the task. Since the ID is specified in bits due to the use of the binary logarithm, both doubling the distance D and halving the width W increase the index of difficulty by exactly 1 bit ($\log_2(2c) = \log_2(2) + \log_2(c) = 1 + \log_2(c)$). Using $2D$ instead of D also has a simple reason: $\mathrm{ID} \geq 0$ holds as long as $D \geq \frac{W}{2}$, i.e., as long as the initial point is not within the target box (note that for rectangles the Euclidean distance between target center and boundary points is at least $\frac{W}{2}$).

[1]Originally, Fitts' law was based on (highly related) one-dimensional tapping and transfer tasks. However, for reasons of consistency we try to use the terminology of *pointing tasks*, which is explained in detail in Section 4.2.1. In particular, we assume that some pointer needs to be moved from a fixed initial point to some rectangular target box.

Fitts' observation is the following: Across a range of values, the ID is directly proportional to the average *movement time (MT)* in a consecutive series of trials, or in other words, the *Index of Performance (IP)* given by

$$\mathrm{IP} := \frac{\mathrm{ID}}{\mathrm{MT}}, \tag{3.2}$$

which expresses the average rate of information processed per second, is constant within different trials.
In particular, movement times which are linear in the ID imply that pointing tasks with virtually no distance to cover are executed in arbitrarily small time on average. However, the target click required at the end of the movement implies additional dwell time [40], which should be independent of the task difficulty.
Hence, Fitts proposed the following affine map with intercept a and slope b [22]:

$$\mathrm{MT} = a + b\,\mathrm{ID} = a + b\log_2\left(\frac{2D}{W}\right). \tag{3.3}$$

This simple but fundamental relation between two decisive factors, namely the distance D and the target width W, which are combined into one ID parameter, and the average movement duration MT was verified in many experimental observations and several linear motor control models (see next section).

Over the years, many variants of this approach have been proposed, the most well-known being the **Wellford** [69] and the **Shannon** [40] **formulation**, suggesting

$$\mathrm{ID} := \log_2\left(\frac{D}{W} + \frac{1}{2}\right) \tag{3.4}$$

respective

$$\mathrm{ID} := \log_2\left(\frac{D}{W} + 1\right). \tag{3.5}$$

Note that, in contrast to the former two variants, the Shannon formulation guarantees $\mathrm{ID} \geq 0$ for any $D \geq 0$ and $W > 0$, which could, e.g., be advantageous for on-line prediction of the *remaining* movement time in movements towards the center of a target box.
An alternative to these variations of Fitts' law which works without logarithm was proposed by **Kvålseth** [37], suggesting the following relationship:

$$\mathrm{MT} = a\left(\frac{D}{W}\right)^{b}. \tag{3.6}$$

This was later taken up by **Meyer** and further developed into his **stochastic optimized submovement model** [46, 45], which proposes

$$\mathrm{MT} = a + b\left(\frac{D}{W}\right)^{\frac{1}{N}}, \tag{3.7}$$

where N denotes the maximum number of submovements that are assumed to occur during the execution of a specific task.[2]

However, while all these laws describe a general relationship between task description and user performance, namely a linear or affine link between the task difficulty and the movement duration, none of these models makes assumptions about underlying causal relationships that could explain these observations. In fact, neither the (individual) task understanding of users nor the realization of desired arm and hand movements through muscle activation processes based on this understanding is addressed by such *quantitative* models. While we find that these models provide an important initial insight into the accomplishment of pointing tasks, in the following we will focus on *qualitative* motor control models, which put the main emphasis on the kinematics and dynamics of human movements.

3.2 Models of Motion Dynamics

3.2.1 1OL-Eq

As discussed in Section 2.1, we mainly distinguish open-loop models, which assume a complete pre-defined control plan that is executed straightforward, from closed-loop models, which allow users to adjust their controls to dynamically observed system outputs on a moment-to-moment basis.
An early attempt to model the dynamics of hand movements was made by Crossman and Goodeve in the early 1980s [14]. They used an first-order approach, i.e., only position and velocity terms were incorporated into their model. In addition, they assumed that users permanently monitor their own position and the position of the target center and internally calculate the remaining distance to determine their velocity.
This approach can be formalized in terms of the previous chapter as follows, regarding the time-continuous case first: The states $x(t) \in \mathbb{R}^l$ correspond to the positions at times $t \in I$, in particular l denotes the dimension of the task (e.g., one-, two- or three-dimensional pointing tasks might be considered). The controls[3] $u(t) \in \mathbb{R}^l$ denote the observed position of the target center. For the pointing task defined and used in the following chapters, targets are assumed to be fixed during tasks, leading to a constant "control" u. However, in theory the model could be applied to target tracking tasks as well.
The main assumption of this model is that the velocity is directly proportional to the observed error vector $e(t) := u(t) - x(t)$. Thus, the resulting movement is given by the first-order

[2]The underlying assumption of corrective movements based on visual feedback is essentially from Crossman and Goodeve [14] and Keele [33].

[3]In order to point out similarities and differences between all the models presented, we try, whenever possible, to use the same notation. In this model, however, $u(t)$ is to be understood rather as an input signal that influences human behavior than as a control option that can be freely chosen.

differential equation

$$\dot{x}(t) = ke(t) = ku(t) - kx(t) \tag{1OL-Eq}$$

with positive constant $k > 0$. This equation obviously takes the form of a linear control system (2.4a) with $A := -k\mathrm{Id}_l$ and $B := k\mathrm{Id}_l$, where Id_l denotes the l-by-l identity matrix. However, the pre-determined choice of u makes the use of this control form and the investigation on associated properties such as complete controllability less natural. Following the argumentation of Crossman and Goodeve [14], we assume that the current position is always observed only relative to the target center, i.e., rather $x_{\mathrm{rel}}(t) := x(t) - u(t)$ than $x(t)$ denotes the respective state of the system. Note that $e(t) = -x_{\mathrm{rel}}(t)$ holds, and thus (1OL-Eq) is equivalent to $\dot{x}(t) = -kx_{\mathrm{rel}}(t)$.

For tasks with constant target position T, i.e., with constant control $u \equiv T$, $\dot{x}_{\mathrm{rel}}(t) = \dot{x}(t)$ holds and thus implies

$$\dot{x}_{\mathrm{rel}}(t) = -kx_{\mathrm{rel}}(t), \tag{3.8}$$

which is of the form (2.1a) with $A := -k\mathrm{Id}_l$. In particular, the only eigenvalue of A is $\lambda = -k < 0$. Hence, Lemma 2.4 yields asymptotic stability of the equilibrium $x^*_{\mathrm{rel}} = 0$, i.e., the solution of (3.8) satisfies $\lim_{t\to\infty} x_{\mathrm{rel}}(t; x_{\mathrm{rel},0}) = 0$ for all initial values $x_{\mathrm{rel},0} \in \mathbb{R}^l$.
In terms of absolute coordinates, the definition of $x_{\mathrm{rel}}(t)$ implies that for any initial value $x(0) = x_0 \in \mathbb{R}^l$, the solution trajectory of (1OL-Eq) gets arbitrarily close to the target center, i.e.,

$$\lim_{t\to\infty} x(t; x_0, u \equiv T) = T. \tag{3.9}$$

Since this is achieved by shifting the equilibrium of a simple *first-order lag* ("1OL") to $x^* = T$, a general approach often called "equilibrium-control" in literature, we denote this model – consistent with the following model names – by **1OL-Eq**.
Depending on what is regarded as part of the system itself and what is regarded as control, this model can be considered as open-loop ((1OL-Eq) with control u), as closed-loop ((1OL-Eq) with control $u - x$), and even as uncontrolled ((3.8)). While in the first case, the constant target coordinate $u \equiv T$ is seen as pre-planned control, from a closed-loop perspective the users are assumed to apply the feedback control $\tilde{u}(t) := u(t) - x(t) = T - x(t)$.

In addition to convergence, we wish to briefly show that 1OL-Eq is not only suitable for reaching constant targets but also in accordance with Fitts' law, a result presented by Crossman and Goodeve [14]. First, we obtain

$$x_{\mathrm{rel}}(t) = e^{-kt} x_{\mathrm{rel},0} \tag{3.10}$$

from the solution formula for linear autonomous differential equations (2.2) applied to (3.8), which for the one-dimensional case[4], i.e., $x(t), u(t) \in \mathbb{R}$, equals

$$\ln\left(\frac{x_{\mathrm{rel}}(t)}{x_{\mathrm{rel},0}}\right) = -kt. \tag{3.11}$$

Evaluating this equation at the time τ at which the rectangular target box with width W is reached first, i.e., $x_{\mathrm{rel}}(\tau) = \frac{W}{2}$ holds if $x_0 \geq T$ and $x_{\mathrm{rel}}(\tau) = -\frac{W}{2}$ holds if $x_0 < T$, yields

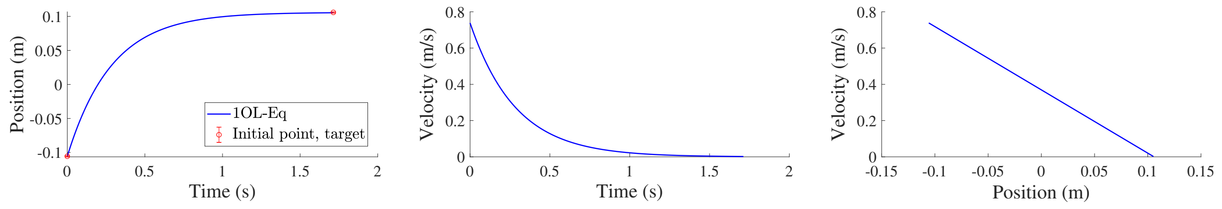
$$\ln\left(\frac{W}{2D}\right) = -k\tau, \tag{3.12}$$

where the equivalence between the absolute value of $x_{\mathrm{rel},0}$ and the distance D was used. Basic logarithm rules thus imply

$$\tau = \tilde{k}\log_2\left(\frac{2D}{W}\right), \tag{3.13}$$

where $\tilde{k} := \frac{\ln 2}{k}$, i.e., the movement time at which the target box is reached first coincides with the prediction of Fitts' law (3.3).
However, pointing tasks typically include the instruction of clicking at the target, which requires additional time. In contrast to (3.3), where this is modeled through the additive constant a, such a target-independent delay is not part of 1OL-Eq. Moreover, early investigations on Fitts' law already suggested that (at least the originally proposed) logarithmic relation only holds for a range of IDs (see [40] for a good overview). In very simple tasks, e.g., it is sometimes argued that users rather tend to rely on pre-determined motor templates, which leads to a discrepancy between the effects of distance and width [25]. Similar discrepancies were also observed in some tapping tasks with low target accuracy constraints [70]. In each of these cases the duration of the experimentally observed trajectories differed relatively clearly from the prediction of Fitts' law. We therefore expect 1OL-Eq not to capture such movements reasonably well.
Furthermore, the velocity of the 1OL-Eq trajectory is proportional to the remaining distance to the target by definition. This necessarily implies that the initial velocity can be generated all of a sudden (requiring infinite force) and that velocity decreases linearly as the target is approached, which can be seen in Figure 3.1. Both characteristics do not reflect typical human behavior, as demonstrated in various pointing task experiments (see, e.g., [39], [50] or our results presented in Chapter 6).
Therefore, it is obvious to consider higher-order models that offer more freedom for the modeling process.

[4]The general case works analogously, using the absolute values of the vectors $x_{\mathrm{rel}}(t)$ and $x_{\mathrm{rel},0}$.



(a) *Position Time Series* **(b)** *Velocity Time Series* **(c)** *Phase Space Plot*

Figure 3.1: *Characteristic features of 1OL-Eq trajectories are both* **(b)** *a high initial velocity and* **(c)** *a velocity profile that is linear in the remaining distance to the target.*

3.2.2 2OL-Eq

In this section, an adaption of the above described 1OL-Eq model to second-order dynamics is presented. Here, the control u (and by equating it with the target center position T, as above, also the positional error) is proportional to the second time derivative of position, i.e., to the acceleration, rather than to the first time derivative, i.e., to the velocity. Formally, such a model would be given by

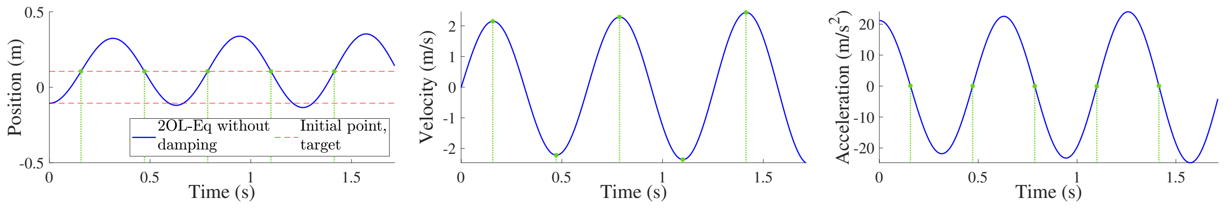
$$\ddot{x}(t) = ke(t) = ku(t) - kx(t) \tag{3.14}$$

with *spring constant* $k > 0$. Analogously to the above discussion, for constant target T this corresponds to the linear differential equation system

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} 0 & \mathrm{ID}_l \\ -k\mathrm{ID}_l & 0 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}, \tag{3.15}$$

where z_1 is the position relative to the target and z_2 is the velocity vector. Since the right-side matrix has eigenvalues $i\sqrt{k}$ and $-i\sqrt{k}$, whose real parts are both equal to zero, Lemma 2.4 yields stability but not asymptotic stability. This means that it is not possible to find an admissible positive maximum distance between starting point and target so that the solutions of the respective initial value problems incorporating the differential equation (3.14) with $u \equiv T$ converge to the target. The reason for this is quite intuitive: The acceleration proportional to the positional error only forces a reduction of the speed change instead of the speed itself as the target is approached, leading to oscillatory behavior (see Figure 3.2). Hence, we need to incorporate a *damping* term in the second-order dynamics (3.14), which takes the current velocity into account.

This is consistent with the widespread interpretation of the biomechanical apparatus as a **spring-mass-damper system** following Granit [29] and Crossman and Goodeve [14]: As depicted in Figure 3.3, in this model the limb's motion is assumed to arise from the interaction between applied force (resulting from an α-γ-coactivation loop of the agonist muscle) and a combination of a spring and a dashpot (representing the antagonist muscle). The mass that characterizes the inertia of the limb can be neglected because it only occurs relative to the



(a) *Position Time Series* **(b)** *Velocity Time Series* **(c)** *Acceleration Time Series*

Figure 3.2: *Each time the target is reached (green lines), the second-order dynamics* (3.14) *ensure that the acceleration equals zero but not the velocity. This means that the target is always passed, resulting in the characteristic oscillatory behavior.*

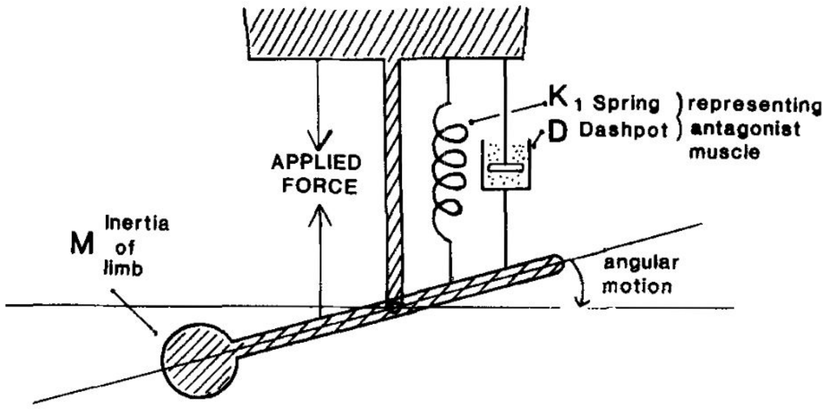


Figure 3.3: *Interpretation of the biomechanical apparatus as a spring-mass-damper system following Granit [29] and Crossman and Goodeve [14]. 2OL-Eq and 2OL-LQR use a linearized version thereof as a simple model of motion dynamics.*

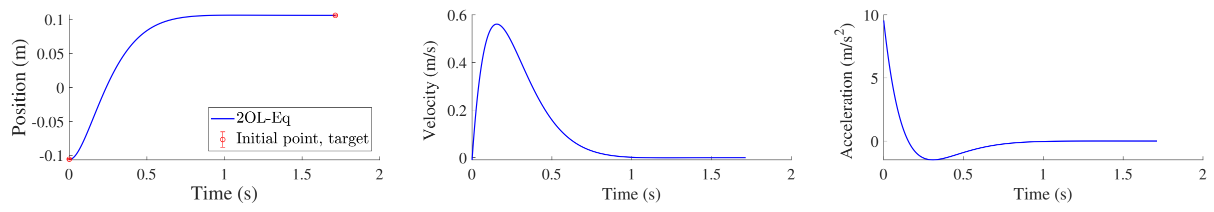
spring stiffness and damping constants. Moreover, we argue that the limb motions required for our pointing tasks are small relative to the length of the limb, so that the angular motion can be linearized. In summary, this leads to the **2OL-Eq** system dynamics

$$\ddot{x}(t) = ke(t) - d\dot{x}(t) = ku(t) - kx(t) - d\dot{x}(t) \tag{2OL-Eq}$$

with spring constant $k > 0$ and damping factor $d > 0$, where x again denotes the (absolute) pointer position and u denotes the target center position, each in Cartesian coordinates. Again, for $u \equiv T$ this equals

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} 0 & \mathrm{ID}_l \\ -k\mathrm{ID}_l & -d\mathrm{ID}_l \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}, \tag{3.16}$$

where z_1 denotes the relative position vector and z_2 denotes the velocity vector. The right-side matrix now has eigenvalues $\lambda_1 = \frac{-d+\sqrt{d^2-4k}}{2}$, $\lambda_2 = \frac{-d-\sqrt{d^2-4k}}{2}$, where the real



(a) *Position Time Series* **(b)** *Velocity Time Series* **(c)** *Acceleration Time Series*

Figure 3.4: *Both* **(b)** *an asymmetric velocity profile and* **(c)** *an instantaneous positive acceleration at the beginning of the movement are characteristic of 2OL-Eq trajectories.*

part of λ_2 is trivially negative due to $d > 0$, and the same holds for λ_1 since $k > 0$ implies $\mathrm{Re}(\lambda_1) < \frac{-d+\sqrt{d^2}}{2} = 0$. Hence, according to Lemma 2.4, the additional damping term $-d\dot{x}$ in (2OL-Eq) ensures asymptotic stability.
In addition, it can be shown that 2OL-Eq trajectories are also consistent with Fitts' law [11].

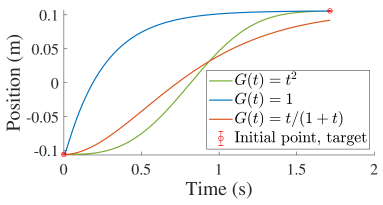
We would like to point out that one of the major drawbacks of 1OL-Eq was the unrealistic velocity profile including a suddenly occurring initial velocity and a permanent decrease in speed, which both cannot be explained physically. Solutions of 2OL-Eq conversely exhibit velocity profiles whose maximum is only reached after an initial increasing phase followed by a main decreasing phase; in particular, the velocity time series has only one local maximum. However, the distance to the target now being proportional to the acceleration instead of the velocity (apart from damping) has a few impacts on the acceleration profile: First, it leads to a jump in the acceleration profile at the beginning of the movement, requiring infinitely fast neuron firing in the muscle activation process, which is physically impossible. Second, the acceleration strongly decreases at first[5], which results in an asymmetric velocity profile, as shown in Figure 3.4. This is contrary to experimental observations, which suggest that typical user trajectories exhibit more symmetric and bell-shaped velocity profiles [48, 50].[6]
A few modifications and extensions concerning these drawbacks are briefly addressed in the following section.

3.2.3 Other Models

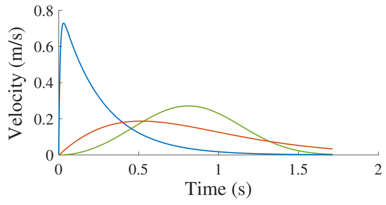
One approach that manages to overcome the previously discussed problems while maintaining the spring-mass-damper system as underlying model, is called **Vector Integration To Endpoint (VITE)** [8].

[5]As soon as the acceleration becomes negative the resulting decrease in velocity has an opposite effect on the acceleration due to (2OL-Eq). Therefore, a slight increase of the acceleration (for sufficient damping towards zero) occurs with time.

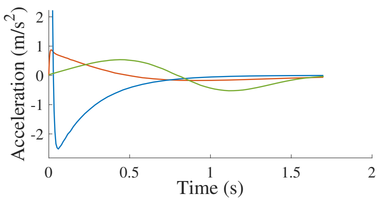
[6]A more detailed comparison between 2OL-Eq trajectories and user trajectories as well as trajectories of other models can be found in Section 6.2.



(a) *Position Time Series*



(b) *Velocity Time Series*



(c) *Acceleration Time Series*

Figure 3.5: *In the VITE model, choosing the GO-Function $G(t)$ allows the generation of different trajectories. While faster-than-linear GO-Functions (green) exhibit a bell-shaped velocity profile and thus lead to more symmetric trajectories, slower-than-linear GO-Functions (red) increase asymmetry (note that in our case the target has not yet been reached). Choosing a linear GO function (blue) is essentially the same as using 2OL-Eq (in the depicted case, the acceleration initially takes values up to* 95*).* [$k = 450$, $d = 120$]

The idea is to dissolve the proportionality between the acceleration $\ddot{x}$ and the distance to the target $u - x$ in (2OL-Eq) by not changing the right side of the equation but rather the meaning of $\ddot{x}$. With the position and the velocity at time t denoted by $p(t)$ and $v(t)$, respectively, (2OL-Eq) becomes equal to

$$\dot{p}(t) = v(t), \tag{3.17a}$$
$$\dot{v}(t) = ku(t) - kp(t) - dv(t). \tag{3.17b}$$

While (3.17b) remains unaffected, VITE allows to modify the relation between v and p by multiplying the positive part[7] of the velocity, $v^+(t) := \max\{0, v(t)\}$, with a "GO"-Function $G(t)$ before integrating it:

$$\begin{aligned} \dot{p}(t) &= G(t)v^+(t), \\ \dot{v}(t) &= ku(t) - kp(t) - dv(t). \end{aligned} \tag{VITE}$$

Note that while (3.17) is equivalent to (3.16) with $z_1 = p - u$ (using $u \equiv T$) and $z_2 = v$, (VITE) cannot be transformed into a linear system of differential equations due to the general non-linearity of G (which was just desired).
While faster-than-linear GO-Functions, i.e., $G(t) := ct^\xi$ with $\xi > 1$ and $c > 0$, lead to more symmetric velocity profiles, slower-than-linear GO-Functions, i.e., $G(t) := c\frac{t}{1+t}$, $c > 0$, provide a slower decrease in speed and thus strengthen asymmetry (see Figure 3.5). Even step response can be modeled by setting $G(t) := 0$ for all $t \in [0, \delta[$ and $G(t) := 1$ for all $t \geq \delta$ using some threshold $\delta > 0$.
However, while we have to admit that nearly any qualitative behavior that could be observed in user trajectories can be modeled with this enormous tool box, there is not much insight

[7]Since VITE models the agonist's and the antanogist's activity separately, it is reasonable to process only positive "input signals" v^+ in each case.

into causal relations between observed phenomena provided from such a generic model. In particular, v and $\dot{v}$ lose their simple physical interpretation as velocity and acceleration, which in turn leads to less informative value of the underlying second-order lag equation.

Another variation of 2OL-Eq is **McRuer's Crossover model** [42, 43, 44], which was originally developed for aircraft simulation. In order to be able to track moving targets, it does not only take the positional error but also *lead*, i.e., the change of the positional error, into account.
Conversely, a **Bang-Bang model** [1] only makes use of controls from the boundary of a compact constraint set, i.e., only "extreme" controls are applied. For example, one might use the sequential combination of maximum acceleration and maximum deceleration (both determined by a gain parameter) such that the target is reached without terminal velocity in minimum time [50].
Costello's Surge model [12] combines the latter two models as follows: The Bang-Bang model is applied until the remaining distance to the target reaches a certain threshold. From this point on, the Crossover model is used to achieve the required target accuracy through smooth movements instead of numerous ever smaller acceleration impulses.
Finally, **Plamodon's Kinematic Theory** [55, 56] describes rapid arm movements through the interaction of two muscle groups (agonist and antagonist), whose impulse responses are assumed to follow log-normal distributions.

However, in the following section we present a wider approach which allows to implement not only specific biomechanical dynamics but also some internal objectives of users.

3.3 Optimization-Based Models

The decisive question that remains open in qualitative models such as Fitts' law is that of the cause of the determined movement duration. While the models of motion dynamics presented in Section 3.2 make more or less detailed assumptions about the biomechanical apparatus, which considerably limits the set of possible movements, the following approach is **optimization-based**:
Given a specific task whose requirements are well-known a priori, users are assumed to derive and internalize main objectives from this task description in order to decide for one of the (often infinitely many) possible "execution strategies" based on these objectives, which is then implemented as a "control law". This can be formalized as follows:

Definition 3.1 (Finite-Horizon Optimal Control Problems). *[30, Def. 6.1 and Remark 6.2] Let $l, m \in \mathbb{N}$. Consider the time interval $I := [0, t_f]$, $t_f > 0$, and the corresponding sampling times with sampling rate $h > 0$, i.e., $t_n := (n-1)h$, $n \in \{1, \dots, N\}$, with $t_f = (N-1)h$. Let the **system dynamics** $f\colon \mathbb{R}^l \times \mathbb{R}^m \longrightarrow \mathbb{R}^l$ be continuous and Lipschitz continuous with respect to its first argument, and consider a continuous **cost function** $g\colon \mathbb{R}^l \times \mathbb{R}^m \longrightarrow \mathbb{R}_{\geq 0}$ and a continuous **terminal cost function** $g_N\colon \mathbb{R}^l \longrightarrow \mathbb{R}_{\geq 0}$.*

(a) The ***time-continuous optimal control problem*** *is given by*

$$\begin{gathered}\textit{Minimize}\quad J(x,u) := g_N(x(t_f)) + \int_0^{t_f} g(x(t),u(t))\ \mathrm{d}t \\ \textit{with respect to } u \in \mathcal{U} \textit{ for all } x_0 \in \mathbb{R}^l,\end{gathered} \tag{3.18}$$

where $x\colon I \longrightarrow \mathbb{R}^l$ is the unique[8] *solution to*

$$\begin{aligned}\dot{x}(t) &= f(x(t),u(t)),\ \ t \in I,\\ x(0) &= x_0.\end{aligned}$$

(b) The ***time-discrete optimal control problem*** *is given by*

$$\begin{gathered}\textit{Minimize}\quad J(x,u) := g_N(x_N) + \sum_{n=1}^{N-1} g(x_n,u_n) \\ \textit{with respect to } u := (u_n)_{n\in\{1,\dots,N-1\}} \subset \mathbb{R}^m \textit{ for all } \bar{x}_1 \in \mathbb{R}^l,\end{gathered} \tag{3.19}$$

where $x := (x_n)_{n\in\{1,\dots,N\}} \subset \mathbb{R}^l$ satisfies

$$\begin{aligned}x_{n+1} &= f(x_n,u_n),\ \ n \in \{1,\dots,N-1\},\\ x_1 &= \bar{x}_1.\end{aligned}$$

In each case, J is called ***objective function*** *or* ***total cost function*** *of the optimization problem.*

Analogously, infinite-horizon optimal control problems can be defined by replacing t_f respective N with "∞" and omit the terminal costs g_N[9]. Moreover, we assumed autonomous system dynamics and cost functions; in general, f, g, and g_N all might explicitly depend on the time t as well.

We would like to explicitly point out that these two variants are not equivalent in general: Even with system dynamics f that fit to each other (Corollary 2.10 describes the "correct" transformation in the linear case), the time-discrete variant assumes the control to be constant between two consecutive sampling times while the set of admissible control functions $\mathcal{U}$ consists of all piecewise continuous functions $u\colon I \longrightarrow \mathbb{R}^m$. Interestingly, this restriction to finitely many changes of control in the time-discrete setting is the reason for the observed time step constraint in case of time-discrete complete controllability, which was described in Remark 2.11.

Regardless of which of these two variants is used, the assumed task objectives of users need

[8]Under the given conditions it can be shown that such a unique solution exists [66, Proposition C.3.8].

[9]Note that with infinite horizon, the total cost function J might assume infinite values, probably even in the infimum, both in the time-continuous and the time-discrete variant.

to be expressed through the cost functions g and g_N, since it is the accumulation of g over the entire movement time and the penalization of the final state through g_N that determines the objective function J and thus the optimal choice of u. For our pointing tasks, which are defined in detail later, some of these objectives might be, e.g., maximum accuracy, minimum time, minimum overshoot, minimum effort, or shortest path.
Naturally, all these objectives can be arbitrarily combined together, as we will do with some of them in our model presented in Chapter 4. However, we explicitly point out that the assumed objectives define the framework for later interpretations: Model trajectories may approximate given user trajectories very well, but if the meaning of the underlying optimization structure is unclear or implausible, little insight can be gained into the actual process of task execution.

In many optimization problems (e.g., convex optimization problems, which also includes our model), the optimal solution is unique. Since human behavior usually differs both among IDs and participants (and even among multiple trials of the same participant for the same task [50]), the different solutions only can, if at all, be assumed optimal with respect to different cost functions.[10] For this reason, it has proven useful not to consider the complete objective function as "universally valid" but to insert some parameters $\lambda \in \Lambda$ which need to be specified to uniquely determine the optimal control problem (and thus its solution). This means that instead of a fixed cost function g an entire family of cost functions $(g_\lambda)_{\lambda \in \Lambda}$ is considered to obtain more degrees of freedom (the same holds for the terminal cost function g_N). Such parameters λ might affect the objectives themselves as well as their respective weights when using a combination of them as cost function.

The question of how and, above all, how many parameters should be included in the objective function is much discussed. On the one hand, enough parameters are required to model the just described diverse behavior sufficiently. On the other hand, too many degrees of freedom for the choice of objectives might lead to redundant or uninterpretable parameters, whose impact on the resulting trajectory becomes unclear.
This is, e.g., the case in Ziebart's *inverse optimal control* model [72], which uses a machine learning approach to fit a generic cost matrix with 36 parameters[11] to a dataset of pointing task trials. While this seems to be very useful for his purpose of predicting mouse movements on-line, it is not our preferred approach as little insight is gained into users' intents and internal processes.
Instead, we focus on optimization models whose assumptions on the objectives are plausible and meaningful, since the results can thus be better interpreted.

[10]At least within the same task and participant, it would be more reasonable to explain this *movement variability* through disturbances such as sensory input noise and motor control noise [68]. However, modeling these disturbances would require a stochastic extension of our deterministic model, which is beyond the scope of this thesis.
[11]A symmetric 8×8-matrix penalizing all combinations of position and its first three derivatives, each in two dimensions, is used.

A relatively simple model is presented in the following section and a wide-used approach to such optimization problems, on which our model will be based, is described in detail in the next chapter. Finally, details on the *parameter fitting process* used for the approximation of user trajectories are given in Chapter 5.

3.3.1 Minimum-Jerk Model

One of the most famous optimization models for pointing tasks is the **Minimum-Jerk model (MinJerk)** introduced by Flash and Hogan [23].
Here, the only objective humans are assumed to aim for is maximum smoothness of the movement from the initial to the target position. Flash and Hogan motivate this assumption by "learning and practice" effects leading to smoother and more graceful movements.
In terms of motor control, smoothness is related to *jerk*, which is the third time derivative of the position. With the position at time $t \geq 0$ denoted by

$$p(t) := \begin{bmatrix} p_1(t) & \dots & p_l(t) \end{bmatrix}^\top \in \mathbb{R}^l, \tag{3.20}$$

where l is the dimension of the pointing task, i.e., l provides information if the hand is supposed to move on the line ($l = 1$), in the plane ($l = 2$) or in the space ($l = 3$), the jerk at time $t \geq 0$ is given by the vector

$$\dddot{p}(t) := \begin{bmatrix} \dddot{p}_1 & \dots & \dddot{p}_l \end{bmatrix}^\top := \begin{bmatrix} \frac{\mathrm{d}^3 p_1}{\mathrm{d}t^3}(t) & \dots & \frac{\mathrm{d}^3 p_l}{\mathrm{d}t^3}(t) \end{bmatrix}^\top \in \mathbb{R}^l. \tag{3.21}$$

Minimizing the time integral of the squared Euclidean norm of the jerk over the entire movement, i.e., minimizing

$$J(p) := \frac{1}{2} \int_0^{t_f} \|\dddot{p}(t)\|_2^2 \, \mathrm{d}t = \sum_{i=1}^{l} \frac{1}{2} \int_0^{t_f} (\dddot{p}_i(t))^2 \, \mathrm{d}t \tag{3.22a}$$

with respect to $p \colon I \longrightarrow \mathbb{R}^l$, $I := [0, t_f]$, where p satisfies

$$\begin{bmatrix} p(0) \\ \dot{p}(0) \\ \ddot{p}(0) \end{bmatrix} = \begin{bmatrix} \bar{p}_0 \\ \bar{v}_0 \\ \bar{a}_0 \end{bmatrix} \in \mathbb{R}^{3l}, \quad \begin{bmatrix} p(t_f) \\ \dot{p}(t_f) \\ \ddot{p}(t_f) \end{bmatrix} = \begin{bmatrix} \bar{p}_{t_f} \\ \bar{v}_{t_f} \\ \bar{a}_{t_f} \end{bmatrix} \in \mathbb{R}^{3l} \tag{3.22b}$$

for given initial and terminal positions $\bar{p}_0$ and $\bar{p}_{t_f}$, velocities $\bar{v}_0$ and $\bar{v}_{t_f}$, and accelerations $\bar{a}_0$ and $\bar{a}_{t_f}$, each in $\mathbb{R}^l$, then leads to the motion trajectory with maximum smoothness.
Note that both squaring the norm of $\dddot{p}(t)$ and multiplying it with the prefactor $\frac{1}{2}$ does not affect the minimizer of the optimization problem (only the value it takes), but simplifies the further analysis. Moreover, the fact that each of the l components of $\dddot{p}(t)$ occurs in exactly one addend in (3.22a) immediately implies that the jerk can be minimized separately in each dimension.

Following the *fundamental lemma of calculus of variations*, the necessary condition for $x = (x_1, \dots, x_l)\colon I \longrightarrow \mathbb{R}^l$ minimizing

$$J(x) := \int_0^{t_f} g(t, x_1, \dots, x_l, \dot{x}_1, \dots, x_l^{(r)})\ \mathrm{d}t \tag{3.23}$$

under some boundary conditions, where $x_1, \dots, x_l$ are assumed to have continuous derivatives up to the $2r$-th order and g needs to be twice continuously differentiable with respect to each of its $l(r+1)+1$ arguments, is that

$$\frac{\partial g}{\partial x_i} - \frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial g}{\partial \dot{x}_i} + \dots + (-1)^r \frac{\mathrm{d}^\mathrm{r}}{\mathrm{d}t^r}\frac{\partial g}{\partial(x_i^{(r)})} = 0, \quad i \in \{1, \dots, l\}, \tag{3.24}$$

holds for all $t \in I$, where the arguments were omitted for the sake of clarity. This system of equations is a special case of the *Euler-Equation* [13, Chapter IV., §3].
In our case of minimizing (3.22a) under the initial and terminal conditions (3.22b), i.e., using $g(t, p_1, \dots, p_l, \dot{p}_1, \dots, \dddot{p}_l) := \frac{1}{2}\sum_{i=1}^{l}(\dddot{p}_i(t))^2$, this leads to

$$-\frac{\mathrm{d}^3}{\mathrm{d}t^3}\frac{\partial g}{\partial \dddot{p}_i}(t) = -\frac{\mathrm{d}^3 \dddot{p}_i}{\mathrm{d}t^3}(t) = -p_i^{(6)}(t) = 0 \tag{3.25}$$

for all $t \in I$ and each dimension $i \in \{1, \dots, l\}$. Therefore only functions p that consist of polynomials of degree $\kappa \leq 5$ in each dimension, i.e., $p_i(t) := \sum_{j=0}^{5} c_j^{(i)} t^j$ with $c_j^{(i)} \in \mathbb{R}$ for all $j \in \{0, \dots, \kappa\}$ holds for each $i \in \{1, \dots, l\}$, remain possible solutions to our optimization problem (3.22). Moreover, the six boundary conditions from (3.22b) are already sufficient to clearly determine these polynomials. Since g is convex in each argument, the corresponding p is indeed the unique solution to (3.22) (see [60, Theorem 2.9.1 and the following remark] for details).

In the following, we would like to briefly derive these polynomials given such initial and terminal conditions. For this purpose, we express the general polynomials in dimensionless coordinates, i.e., we use $\tau(t) := \frac{t}{t_f} \in [0,1]$ for all $t \in I$, which leads to

$$p_i(t) = \sum_{j=0}^{5} \tilde{c}_j^{(i)} (\tau(t))^j, \quad i \in \{1, \dots, l\}, \tag{3.26}$$

with $\tilde{c}_j^{(i)} := c_j^{(i)} (t_f)^j$.
The first and the second derivative of these polynomials with respect to t are given by

$$\dot{p}_i(t) = \frac{1}{t_f}\sum_{j=1}^{5} j \tilde{c}_j^{(i)} (\tau(t))^{j-1}, \quad i \in \{1, \dots, l\}, \tag{3.27}$$

and

$$\ddot{p}_i(t) = \frac{1}{(t_f)^2} \sum_{j=2}^{5} j(j-1)\tilde{c}_j^{(i)} (\tau(t))^{j-2}, \quad i \in \{1, \dots, l\}. \tag{3.28}$$

Because $p_i(0) = \tilde{c}_0^{(i)}$ holds in (3.26), the initial position constraints from (3.22b) immediately imply $\tilde{c}_0^{(i)} = \bar{p}_0^{(i)}$ for all $i \in \{1, \dots, l\}$. As can be derived from (3.27) and (3.28), analogously $\dot{p}_i(0) = \frac{\tilde{c}_1^{(i)}}{t_f}$ and $\ddot{p}_i(0) = \frac{2\tilde{c}_2^{(i)}}{(t_f)^2}$ apply. Hence, the initial velocity and acceleration conditions from (3.22b) yield $\tilde{c}_1^{(i)} = \bar{v}_0^{(i)} t_f$ respective $\tilde{c}_2^{(i)} = \bar{a}_0^{(i)} \frac{(t_f)^2}{2}$ for all $i \in \{1, \dots, l\}$.

Even though the initial values thus need to be rescaled by t_f to yield the corresponding prefactors $\tilde{c}$, the implementation of the terminal conditions becomes much easier through the use of these dimensionless times τ. In particular, the following equations result directly from inserting $\tau(t_f) = 1$ into (3.26), (3.27), and (3.28) for all $i \in \{1, \dots, l\}$:

$$p_i(t_f) = \sum_{j=0}^{5} \tilde{c}_j^{(i)}, \quad \dot{p}_i(t_f) = \sum_{j=1}^{5} \frac{j}{t_f} \tilde{c}_j^{(i)}, \quad \ddot{p}_i(t_f) = \sum_{j=2}^{5} \frac{j(j-1)}{(t_f)^2} \tilde{c}_j^{(i)}. \tag{3.29}$$

Together with the terminal conditions from (3.22b), this immediately implies that $\tilde{c}_3^{(i)}$, $\tilde{c}_4^{(i)}$, and $\tilde{c}_5^{(i)}$ can be computed by solving the system of linear equations

$$\begin{pmatrix} 1 & 1 & 1 \\ 3 & 4 & 5 \\ 6 & 12 & 20 \end{pmatrix} \begin{pmatrix} \tilde{c}_3^{(i)} \\ \tilde{c}_4^{(i)} \\ \tilde{c}_5^{(i)} \end{pmatrix} = \begin{pmatrix} \bar{p}_{t_f}^{(i)} - \tilde{c}_0^{(i)} - \tilde{c}_1^{(i)} - \tilde{c}_2^{(i)} \\ \bar{v}_{t_f}^{(i)} t_f - \tilde{c}_1^{(i)} - 2\tilde{c}_2^{(i)} \\ \bar{a}_{t_f}^{(i)} (t_f)^2 - 2\tilde{c}_2^{(i)} \end{pmatrix}, \quad i \in \{1, \dots, l\}. \tag{3.30}$$

In summary, this leads to the following corollary:

Corollary 3.2 (Minimum-Jerk Polynomial). *Let $t_f > 0$, $I := [0, t_f]$, and let $l \in \mathbb{N}$ denote the state dimension. The unique function $p = (p_1, \dots, p_l) \colon I \longrightarrow \mathbb{R}^l$ which minimizes*

$$J(p) := \frac{1}{2} \int_0^{t_f} \|\dddot{p}(t)\|_2^2 \, \mathrm{d}t$$

under the additional constraints

$$\begin{bmatrix} p(0) \\ \dot{p}(0) \\ \ddot{p}(0) \end{bmatrix} = \begin{bmatrix} \bar{p}_0 \\ \bar{v}_0 \\ \bar{a}_0 \end{bmatrix} \in \mathbb{R}^{3l}, \qquad \begin{bmatrix} p(t_f) \\ \dot{p}(t_f) \\ \ddot{p}(t_f) \end{bmatrix} = \begin{bmatrix} \bar{p}_{t_f} \\ \bar{v}_{t_f} \\ \bar{a}_{t_f} \end{bmatrix} \in \mathbb{R}^{3l},$$

is in each dimension given by the fifth-degree polynomial

$$p_i(t) = \sum_{j=0}^{5} \tilde{c}_j^{(i)} (\tau(t))^j, \quad i \in \{1, \dots, l\},$$

with $\tau(t) := \frac{t}{t_f}$ *and*

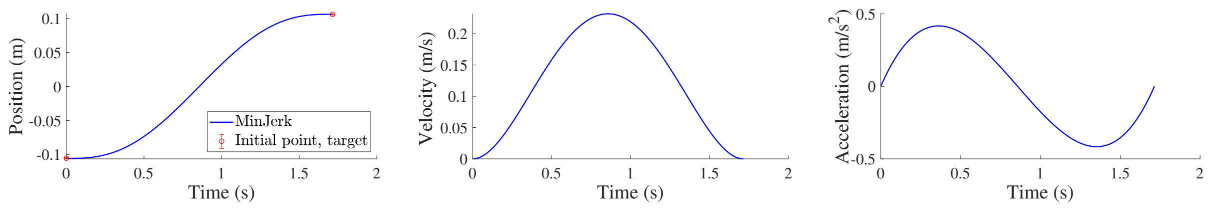
$$\begin{bmatrix} \tilde{c}_0^{(i)} \\ \tilde{c}_1^{(i)} \\ \tilde{c}_2^{(i)} \\ \tilde{c}_3^{(i)} \\ \tilde{c}_4^{(i)} \\ \tilde{c}_5^{(i)} \end{bmatrix} := \begin{bmatrix} \bar{p}_0^{(i)} \\ \bar{v}_0^{(i)} t_f \\ \bar{a}_0^{(i)} \frac{(t_f)^2}{2} \\ 10(\bar{p}_{t_f}^{(i)} - \bar{p}_0^{(i)}) - 4(\bar{v}_{t_f}^{(i)} - \bar{v}_0^{(i)})t_f + (\bar{a}_{t_f}^{(i)} - \bar{a}_0^{(i)})\frac{(t_f)^2}{2} - 10\bar{v}_0^{(i)} t_f - \bar{a}_0^{(i)}(t_f)^2 \\ -15(\bar{p}_{t_f}^{(i)} - \bar{p}_0^{(i)}) + 7(\bar{v}_{t_f}^{(i)} - \bar{v}_0^{(i)})t_f - (\bar{a}_{t_f}^{(i)} - \bar{a}_0^{(i)})(t_f)^2 + 15\bar{v}_0^{(i)} t_f + \bar{a}_0^{(i)}\frac{(t_f)^2}{2} \\ 6(\bar{p}_{t_f}^{(i)} - \bar{p}_0^{(i)}) - 3(\bar{v}_{t_f}^{(i)} - \bar{v}_0^{(i)})t_f + (\bar{a}_{t_f}^{(i)} - \bar{a}_0^{(i)})\frac{(t_f)^2}{2} - 6\bar{v}_0^{(i)} t_f \end{bmatrix}.$$

Flash and Hogan, e.g., derived the minimum-jerk polynomial for two-dimensional arm movements, i.e., $l = 2$, assuming zero velocity and acceleration in each dimension both at the beginning and at the end of the movement, i.e., $\bar{v}_0 = \bar{v}_{t_f} = \bar{a}_0 = \bar{a}_{t_f} = 0$. In this case, $\bar{p}_0$ and $\bar{p}_{t_f}$ are the only free parameters, which considerably simplifies the direct solution formulas for $p_1(t)$ respective $p_2(t)$ (see [23]).
While the solution trajectories of this model are analyzed and compared to those of other models in Section 6.2 (Figure 3.6 already gives a first impression), we want to point out that given these initial and terminal constraints the solution trajectory is clearly determined. This particularly means that there are no undetermined parameters that could explain different behavior among different users.

One of the many extensions of this widely acknowledged MinJerk model is the **Dynamic MinJerk model** introduced by Hoff and Arbib [32]. Here, the minimum-jerk trajectory from Corollary 3.2 is computed bit by bit using an equivalent linear control system of the form $\dot{x}(t) = Ax(t) + Bu(t)$, where the states $x(t)$ incorporate the current position, velocity, and acceleration, and the controls $u(t)$ denote the currently perceived target position. Hence, this model can be described as "minimum-jerk dynamics with equilibrium-control" (see Section 3.2 for a more detailed discussion on this term). In particular, the continuous (or time-discrete) re-computation of the remaining trajectory allows to take into account both moving targets and sensory input effects on the perception of the target position. Moreover, perturbations of the own position, velocity, and acceleration during the movement are compensated because the minimum-jerk trajectory is permanently (or repeatedly) adjusted to the currently observed state $x(t)$.

Furthermore, it is possible to adapt the minimum-jerk principle to *via-point tasks*, i.e., to pointing tasks with multiple targets that must be reached in a predefined order (see Section 4.3.2). A detailed derivation based on analytical methods for *optimal control problems with interior point equality constraints*, which are, e.g., described in [7, Section 3.5], can be found in [23, Appendix C].

In Chapter 6.2, we will compare our model, which still needs to be defined, with MinJerk among others. We implemented all models to be compared in discrete time and beginning



(a) *Position Time Series* **(b)** *Velocity Time Series* **(c)** *Acceleration Time Series*

Figure 3.6: *Typical characteristics of the minimum-jerk trajectory with identical initial and terminal velocity and acceleration constraints are the completely symmetric position, velocity and acceleration profiles.*

with the next chapter we will mainly switch to the time-discrete case in our analysis, too. For reasons of consistency we thus briefly introduce the respective time-discrete notation for MinJerk in one spatial dimension[12]:
As described in Section 2.2 in detail, the interval $I := [0, t_f]$ is partitioned into $\tilde{N} - 1 \in \mathbb{N}$ equidistant subintervals of length h. The final time t_f is thus given by $t_f = t_{\tilde{N}} = (\tilde{N} - 1)h$. The polynomial is then evaluated once at the beginning of each subinterval and at the and of the movement, i.e., at the times $t_n = (n-1)h$, $n \in \{1, \dots, \tilde{N}\}$.
Hence, the position at the time step $n \in \{1, \dots, \tilde{N}\}$ is given by

$$p_n^{\text{MinJerk}} := p((n-1)h) = \sum_{j=0}^{5} \tilde{c}_j \left(\frac{(n-1)h}{t_f} \right)^j = \sum_{j=0}^{5} \tilde{c}_j \left(\frac{n-1}{\tilde{N}-1} \right)^j \tag{3.31}$$

with $\tilde{c}_j$, $j \in \{0, \dots, 5\}$, from Corollary 3.2.

[12]As derived above, the polynomials for further spatial dimensions are independent of each other and have exactly the same form.

Chapter 4

2OL-LQR

In this chapter, we introduce our **2OL-LQR model**, which combines parts of models presented in the preceding sections. However, at its core, our model is based on a well explored optimization technique for a certain type of optimal control problems – the **linear-quadratic regulator (LQR)**, which we need to define and analyze first. In the following section, we describe appropriate modifications of this concept for our purpose of simulating pointing tasks, resulting in the basic structure of our proposed model. Finally, we briefly explain how to extend this approach to more general via-point tasks.

4.1 Linear-Quadratic Regulator (LQR)

In the following, we consider a special case of the time-discrete optimal control problem introduced in Definition 3.1:

Definition 4.1 (Linear-Quadratic Optimal Control Problem).
Consider arbitrary ***system dynamics matrices*** $A \in \mathbb{R}^{l \times l}$, $B \in \mathbb{R}^{l \times m}$ *and an arbitrary finite number of discrete time steps* $N \in \mathbb{N}$.
Let the ***state cost matrices*** $Q_n \in \mathbb{R}^{l \times l}$, $n \in \{1, \ldots, N\}$, *be symmetric and positive semidefinite and the* ***control cost matrices*** $R_n \in \mathbb{R}^{m \times m}$, $n \in \{1, \ldots, N-1\}$, *be symmetric and positive definite.*
The ***linear-quadratic optimal control problem with horizon*** N *is given by*

$$\begin{gathered} \textit{Minimize} \quad J_N(x,u) := \sum_{n=1}^{N} x_n^\top Q_n x_n + \sum_{n=1}^{N-1} u_n^\top R_n u_n \\ \textit{with respect to } u := (u_n)_{n \in \{1,\ldots,N-1\}} \subset \mathbb{R}^m \textit{ for all } \bar{x}_1 \in \mathbb{R}^l, \end{gathered} \tag{4.1a}$$

where $x := (x_n)_{n \in \{1,\ldots,N\}} \subset \mathbb{R}^l$ *satisfies*

$$\begin{gathered} x_{n+1} = Ax_n + Bu_n, \quad n \in \{1, \ldots, N-1\}, \\ x_1 = \bar{x}_1. \end{gathered} \tag{4.1b}$$

Here, the states x_n, $n \in \{1, \dots, N\}$, which evolve according to the linear system dynamics (4.1b), are penalized quadratically by the respective state cost matrix $Q_n \in \mathbb{R}^{l \times l}$. The controls u_n, $n \in \{1, \dots, N-1\}$, which uniquely determine this state sequence, are simultaneously penalized quadratically by the respective control cost matrix R_n. Additionally, it would be possible to define cost matrices M_n penalizing combinations of state and cost vectors, i.e., to add terms of the form $2x_n^\top M_n u_n$ to the objective function $J_N(x, u)$. We decided to abandon this generalization, since such combined cost terms cannot be reasonably interpreted in our model.
Note that $J_N(x, u) \geq 0$ holds because of the positive definiteness of the matrices R_n and the positive semidefiniteness of the matrices Q_n.

The solution to this optimal control problem can be given in general terms, which is the main result of this section:

Theorem 4.2 (Linear-Quadratic Regulator (LQR)).
The unique solution $u^* = (u_n^*)_{n \in \{1,\dots,N-1\}}$ to the time-discrete linear-quadratic optimal control problem with horizon N (4.1) is given by

$$\begin{aligned} u_n^* &:= -K_n x_n, \quad n \in \{1, \dots, N-1\}, \\ K_n &= (R_n + B^\top S_{n+1} B)^{-1} B^\top S_{n+1} A, \quad n \in \{1, \dots, N-1\}, \end{aligned} \tag{4.2}$$

where the symmetric matrices $S_n \in \mathbb{R}^{l \times l}$ can be determined backwards in time by solving the ***Discrete Riccati Equations****:*

$$\begin{gathered} S_n = Q_n + A^\top S_{n+1} A - A^\top S_{n+1} B (R_n + B^\top S_{n+1} B)^{-1} B^\top S_{n+1} A, \quad n \in \{1, \dots, N-1\}, \\ S_N = Q_N. \end{gathered} \tag{4.3}$$

In particular, the ***optimal feedback gain*** *matrices $K_n \in \mathbb{R}^{m \times l}$ only depend on the system matrices A, B, Q_n, R_n, which define the underlying system dynamics and the considered objective function, i.e., they can be computed a priori without any information about the states x_n and even independent of the initial state $\bar{x}_1$.*

Proof. (Large parts of this proof are essentially based on [7].)
Since the solution trajectory $x = (x_n)_{n \in \{1,\dots,N\}}$ of (4.1) must satisfy the system dynamics equation (4.1b), which is equivalent to

$$\begin{gathered} f(x_n, u_n) - x_{n+1} = 0, \quad f(\hat{x}, \hat{u}) := A\hat{x} + B\hat{u}, \quad n \in \{1, \dots, N-1\}, \\ \tilde{f}(x_1) = 0, \quad \tilde{f}(\hat{x}) := \bar{x}_1 - \hat{x}, \end{gathered}$$

adding arbitrary multiplies of $(f(x_n, u_n) - x_{n+1})$ and $\tilde{f}(x_1)$ to the objective function $J_N(x, u)$

cannot change its value for such trajectories.[1] Hence, we can minimize

$$\begin{aligned} J_N^\lambda(x,u) &:= \frac{1}{2}J_N(x,u) + \lambda_1^\top \tilde{f}(x_1) + \sum_{n=1}^{N-1} \lambda_{n+1}^\top (f(x_n,u_n) - x_{n+1}) = \\ &= \frac{1}{2}\sum_{n=1}^{N} x_n^\top Q_n x_n + \frac{1}{2}\sum_{n=1}^{N-1} u_n^\top R_n u_n + \lambda_1^\top \tilde{f}(x_1) + \sum_{n=1}^{N-1} \lambda_n^\top (f(x_n,u_n) - x_{n+1}) \end{aligned}$$

under the constraint (4.1b), where $\lambda = (\lambda_n)_{n\in\{1,\dots,N\}} \subset \mathbb{R}^l$ denotes an (initially) arbitrary cofactor sequence of so-called **Lagrange multipliers**, just as well. Note that the multiplication of the objective function $J_N(x,u)$ by $\frac{1}{2}$ in order to simplify the following calculations also cannot change the optimal control u^* (but the respective optimal value, of course).
The modified objective function $J_N^\lambda(x,u)$ together with (4.1b) thus represents the same optimization problem as $J_N(x,u)$ together with (4.1b).
Using the total differential[2] of J_N^λ, i.e.,

$$\mathrm{d}J_N^\lambda = \sum_{n=1}^{N} \frac{\partial J_N^\lambda}{\partial x_n}\mathrm{d}x_n + \sum_{n=1}^{N-1} \frac{\partial J_N^\lambda}{\partial u_n}\mathrm{d}u_n, \tag{4.4}$$

we can take the perspective that it is only u which should affect J_N^λ since the corresponding states x_n are for any fixed u given by the constraints (4.1b) (instead of the other way around).
In our case, the partial derivatives of J_N^λ are given by

$$\frac{\partial J_N^\lambda}{\partial x_1}(x,u) = \frac{1}{2}\frac{\partial J_N}{\partial x_1}(x,u) + \lambda_1^\top \frac{\partial \tilde{f}}{\partial \hat{x}}(x_1) + \lambda_2^\top \frac{\partial f}{\partial \hat{x}}(x_1,u_1) = Q_1 x_1 + A^\top \lambda_2 - \lambda_1, \tag{4.5}$$

$$\frac{\partial J_N^\lambda}{\partial x_n}(x,u) = \frac{1}{2}\frac{\partial J_N}{\partial x_n}(x,u) + \lambda_{n+1}^\top \frac{\partial f}{\partial \hat{x}}(x_n,u_n) - \lambda_n = Q_n x_n + A^\top \lambda_{n+1} - \lambda_n, \quad n \in \{2,\dots,N-1\}, \tag{4.6}$$

$$\frac{\partial J_N^\lambda}{\partial x_N}(x,u) = \frac{1}{2}\frac{\partial J_N}{\partial x_N}(x,u) - \lambda_N = Q_N x_N - \lambda_N, \tag{4.7}$$

$$\frac{\partial J_N^\lambda}{\partial u_n}(x,u) = \frac{1}{2}\frac{\partial J_N}{\partial u_n}(x,u) + \lambda_{n+1}^\top \frac{\partial f}{\partial \hat{u}}(x_n,u_n) = R_n u_n + B^\top \lambda_{n+1}, \quad n \in \{1,\dots,N-1\}. \tag{4.8}$$

In particular, $\lambda_1 = \frac{\partial J_N^\lambda}{\partial \bar{x}_1}$ is the rate of change of J_N^λ as a function of the initial constraint $\bar{x}_1$. Analogously, the effect of infinitesimally small changes in the system dynamics function f on J_N^λ is expressed by λ_n, $n \in \{2,\dots,N\}$.

[1]Note that in the considered case the cost matrices Q_n, R_n might change at every time step, while the system dynamics matrices A, B are assumed constant. The general case can be derived analogously by allowing a different function f_n for every $n \in \{1,\dots,N-1\}$.

[2]This commonly used notation is mathematically correct, when, e.g., regarding $\mathrm{d}x$ and $\mathrm{d}u$ as differential forms. For further details, see, e.g., [28, Section 2.2.2.1].

Because λ was arbitrary, we are free to choose. In order to express the determination of x_n by (4.1b) for any fixed u through the Lagrange parameters, we equate (4.5), (4.6), and (4.7) with zero, i.e., we set

$$\frac{\partial J_N^\lambda}{\partial x_n}(x,u) = 0, \quad n \in \{1,\dots,N\},$$

which yields

$$\begin{aligned} \lambda_n &:= \frac{1}{2}\frac{\partial J_N}{\partial x_n}(x,u) + \lambda_{n+1}^\top \frac{\partial f}{\partial \hat{x}}(x_n,u_n) = Q_n x_n + A^\top \lambda_{n+1}, \quad n \in \{1,\dots,N-1\}, \\ \lambda_N &:= \frac{1}{2}\frac{\partial J_N}{\partial x_N}(x,u) = Q_N x_N. \end{aligned} \tag{4.9}$$

The definition of the total differential (4.4) thus yields

$$\mathrm{d}J_N^\lambda = \sum_{n=1}^{N-1} \frac{\partial J_N^\lambda}{\partial u_n}\mathrm{d}u_n, \tag{4.10}$$

i.e., now it is only u that has an impact on the values of J_N^λ. In fact, we have chosen the Lagrange multipliers λ_n appropriately depending on the states x_n in order to eliminate the effect of infinitesimally small changes in x_n assuming (4.1b) applies, i.e., we have "compensated" the *overall* effect of changes in x_n on the objective function J_N, that is, the direct effect (via the corresponding stage costs) as well as the indirect effect (via the system dynamics constraints).

The remaining necessary condition for u^* being a local minimum of the considered optimization problem (4.1) thus is

$$\frac{\partial J_N^\lambda}{\partial u_n}(x,u^*) = 0, \quad n \in \{1,\dots,N-1\}. \tag{4.11}$$

Using (4.8), this is equivalent to

$$u_n^* = -R_n^{-1}B^\top \lambda_{n+1}, \quad n \in \{1,\dots,N-1\}. \tag{4.12}$$

Moreover, the assumed positive definiteness of R_n, $n \in \{1,\dots,N-1\}$, directly implies that the sufficient condition for u^* being a local minimum of (4.1) is satisfied:

$$\frac{\partial^2 J_N^\lambda}{\partial u_n^2}(x,u^*) = R_n > 0, \quad n \in \{1,\dots,N-1\}. \tag{4.13}$$

Thus, the optimal trajectory is given by

$$\begin{aligned} x_{n+1} &= Ax_n - BR_n^{-1}B^\top \lambda_{n+1}, \quad n \in \{1,\dots,N-1\}, \\ x_1 &= \bar{x}_1, \end{aligned} \tag{4.14}$$

with

$$\begin{aligned}\lambda_n &= Q_n x_n + A^\top \lambda_{n+1}, \quad n \in \{1, \dots, N-1\},\\ \lambda_N &= Q_N x_N.\end{aligned} \tag{4.15}$$

In this form, both iteration formulas each require the complete solution of the other formula since they are coupled and need to be solved from different time directions. Fortunately, it is possible to decouple them, as we show for the rest of the proof.
To this end, we assume that there exist matrices $S_n \in \mathbb{R}^{l \times l}$, $n \in \{1, \dots, N\}$, so that

$$\lambda_n = S_n x_n, \quad n \in \{1, \dots, N\}. \tag{4.16}$$

holds. First, we use this assumption to specify u_n independent of λ_n:
Inserting

$$\lambda_{n+1} \overset{(4.16)}{=} S_{n+1} x_{n+1} \overset{(4.1b)}{=} S_{n+1}(Ax_n + Bu_n), \quad n \in \{1, \dots, N-1\} \tag{4.17}$$

into (4.12) yields for all $n \in \{1, \dots, N-1\}$

$$\begin{aligned} u_n^* &= -R_n^{-1} B^\top S_{n+1}(Ax_n + Bu_n)\\ \iff u_n^* &= -(R_n + B^\top S_{n+1} B)^{-1} B^\top S_{n+1} A x_n. \end{aligned} \tag{4.18}$$

Defining

$$K_n := (R_n + B^\top S_{n+1} B)^{-1} B^\top S_{n+1} A x_n, \quad n \in \{1, \dots, N-1\},$$

yields the first part of the solution.

Hence, for all $n \in \{1, \dots, N-1\}$

$$\begin{aligned} S_n x_n &\overset{(4.16)}{=} \lambda_n \overset{(4.15)}{=} Q_n x_n + A^\top \lambda_{n+1} \overset{(4.16)}{=} Q_n x_n + A^\top S_{n+1} x_{n+1} \overset{(4.1b)}{=}\\ &\overset{(4.1b)}{=} Q_n x_n + A^\top S_{n+1}(Ax_n + Bu_n^*) \overset{(4.18)}{=}\\ &\overset{(4.18)}{=} Q_n x_n + A^\top S_{n+1}(Ax_n - B(R_n + B^\top S_{n+1} B)^{-1} B^\top S_{n+1} A x_n)\\ \iff &\Big(S_n - Q_n - A^\top S_{n+1} A + A^\top S_{n+1} B(R_n + B^\top S_{n+1} B)^{-1} B^\top S_{n+1} A\Big) x_n = 0 \end{aligned}$$

needs to hold for the states x_n of the trajectory resulting from the optimal control u^*, which is satisfied if

$$S_n = Q_n + A^\top S_{n+1} A - A^\top S_{n+1} B(R_n + B^\top S_{n+1} B)^{-1} B^\top S_{n+1} A \tag{4.19}$$

holds for all $n \in \{1, \dots, N-1\}$.
Analogously, for $n = N$

$$S_N x_N \overset{(4.16)}{=} \lambda_N \overset{(4.15)}{=} Q_N x_N$$

holds if[3]

$$S_N = Q_N. \tag{4.20}$$

In particular, (4.16) holds with these matrices S_n, which are independent of all states x_n, the initial state $\bar{x}_1$, and all controls u_n but only depend on the "general setup" of the optimization problem, i.e., on the system dynamics matrices A, B and the time-varying cost matrices Q_n, R_n. Thus, the Discrete Riccati-Equations (4.19) only need to be solved once a priori backwards in time and the solution matrices S_n determine the unique solution of the optimal control problem u^* through (4.18).
Moreover, all S_n are symmetric since all Q_n and R_n are assumed to be symmetric and the terms in the recursion formula (4.19) including S_{n+1} are thus symmetric by backward induction and construction, using that sums and inverses preserve symmetry.
This finally concludes the proof. □

Theorem 4.2 contains some important statements:
First, the unique minimum of $J_N(x, u)$ with respect to the linear system dynamics constraints is assumed for any initial state x_0 in the respective optimal control sequence u^*.
Second, this optimal control can be specified directly in a **linear feedback** form, i.e.,

$$u_n^* = -K_n x_n, \quad n \in \{1, \dots, N-1\}, \tag{4.21}$$

where the optimal feedback gain matrices K_n can be computed once a priori by solving the Discrete Riccati Equations backwards in time. With these K_n, starting in x_0, the **optimal trajectory** can be derived by alternately computing the respective optimal feedback u_n^* from (4.21) and the consecutive state x_{n+1} from the system dynamics (4.1b), both of which involve only matrix-vector multiplications.

In addition to Theorem 4.2, we want to give an explicit formula for the minimum value that the objective function reaches given some initial state x_0. To this end, we need to extend Definition 4.1:

Definition 4.3. *In the case of Definition 4.1, the* ***partially cutoff (linear-quadratic) optimal control problem with horizon*** $K \in \{1, \dots, N\}$ *is given by*

$$\text{Minimize} \quad J_N^K(x^{K-1}, u^{K-1}) := \sum_{n=N-K+1}^{N} x_n^\top Q_n x_n + \sum_{n=N-K+1}^{N-1} u_n^\top R_n u_n \tag{4.22a}$$
$$\text{with respect to } u^{K-1} = (u_n)_{n \in \{N-K+1, \dots, N-1\}} \subset \mathbb{R}^m \text{ for all } \bar{x}_{N-K+1} \in \mathbb{R}^l,$$

[3]In fact, it can be shown that (4.19) and (4.20) are even necessary conditions for the respectively derived equations (because they need to hold for arbitrary $x_n \in \mathbb{R}^l$ due to Bellman's principle of optimality).

where $x^{K-1} := (x_n)_{n\in\{N-K+1,\dots,N\}} \subset \mathbb{R}^l$ *satisfies*

$$\begin{aligned} x_{n+1} &= Ax_n + Bu_n, \quad n \in \{N-K+1,\dots,N-1\}, \\ x_{N-K+1} &= \bar{x}_{N-K+1}. \end{aligned} \tag{4.22b}$$

The respective ***optimal value function*** *is given by*

$$V_N^K(\bar{x}_{N-K+1}) := \min_{u^{K-1}\subset\mathbb{R}^m} J_N^K(x^{K-1}, u^{K-1}), \quad \bar{x}_{N-K+1} \in \mathbb{R}^l. \tag{4.23}$$

The partially cutoff optimal control problem with horizon $K \in \{1,\dots,N\}$ and initial state $\bar{x}_{N-K+1} := x_{N-K+1}^{\bar{x}_1,(u_1,\dots,u_{N-K})}$, where $x^{\bar{x}_1,(u_1,\dots,u_{N-K})}$ denotes the solution of (4.1b) up to time step $N-K+1$ with initial state $\bar{x}_1$ and applied control $\bar{u} := (u_n)_{n\in\{1,\dots,N-K\}}$, thus exactly corresponds to the original optimal control problem with horizon N and initial state $\bar{x}_1$ from Definition 4.1, where the (not necessarily optimal) first $N-K$ controls are determined by $\bar{u}$ and only the remaining $K-1$ controls in u^{K-1} are to be optimized.
Note that the minimum in (4.23) is assumed according to Theorem 4.2, which can be applied because each partially cutoff optimal control problem (4.22) can be brought into a separate complete optimal control problem with horizon K of the form (4.1) by shifting indexes and using $\bar{x}_{N-K+1}$ as initial state $\bar{x}_1$. Particularly, in case of $K = N$, (4.22) corresponds exactly to (4.1), i.e., $J_N^N \equiv J_N$ holds with $x^N \equiv x$ and $u^N \equiv u$.

Following **Bellman's principle of optimality** [66, Lemma 8.1.5], the optimal value function of the optimal control problem with horizon N (4.1),

$$V_N^N(\bar{x}_1) = \min_{u\subset\mathbb{R}^m} J_N(x,u), \tag{4.24}$$

for each $\bar{x}_1 \in \mathbb{R}^l$ satisfies

$$\begin{aligned} V_N^N(\bar{x}_1) &= \min_{\tilde{u}\in\mathbb{R}^m} \left[\bar{x}_1^\top Q_1 \bar{x}_1 + \tilde{u}^\top R_1 \tilde{u} + V_N^{N-1}(x_2^{\bar{x}_1,(\tilde{u})})\right] = \\ &= \bar{x}_1^\top Q_1 \bar{x}_1 + (\tilde{u}^*)^\top R_1 \tilde{u}^* + V_N^{N-1}(x_2^{\bar{x}_1,(\tilde{u}^*)}), \end{aligned} \tag{4.25}$$

where $V_N^{N-1}(x_2^{\bar{x}_1,(\tilde{u})})$ denotes the minimum value of the partially cutoff optimal control problem with horizon $N-1$ starting in $x_2^{\bar{x}_1,(\tilde{u})} = A\bar{x}_1 + B\tilde{u}$ and $\tilde{u}^*$ is the first control of the optimal control sequence $u^* = (u_n^*)_{n\in\{1,\dots,N-1\}}$ of (4.1).
More general, the optimal value function for the partially cutoff optimal control problem (4.22) with horizon $K \in \{2,\dots,N\}$, i.e., beginning at $N-K+1 \in \{1,\dots,N-1\}$, which is defined by (4.23), satisfies

$$\begin{aligned} V_N^K(\bar{x}_{N-K+1}) &= \min_{\tilde{u}\in\mathbb{R}^m} \left[\bar{x}_{N-K+1}^\top Q_{N-K+1} \bar{x}_{N-K+1} + \tilde{u}^\top R_{N-K+1} \tilde{u} + V_N^{K-1}(x_{N-K+2}^{\bar{x}_{N-K+1},(\tilde{u})})\right] = \\ &= \bar{x}_{N-K+1}^\top Q_{N-K+1} \bar{x}_{N-K+1} + (\tilde{u}^*)^\top R_{N-K+1} \tilde{u}^* + V_N^{K-1}(x_{N-K+2}^{\bar{x}_{N-K+1},(\tilde{u}^*)}), \end{aligned} \tag{4.26}$$

where $V_N^{K-1}(x_{N-K+2}^{\bar{x}_{N-K+1},(\tilde{u})})$ denotes the minimum value of the partially cutoff optimal control problem with horizon $K-1$ starting in $x_{N-K+2}^{\bar{x}_{N-K+1},(\tilde{u})} = A\bar{x}_{N-K+1} + B\tilde{u}$ and $\tilde{u}^*$ is the first control of the optimal control sequence $u^{K-1,*} = (u_n^*)_{n\in\{N-K+1,\ldots,N-1\}} \subset \mathbb{R}^m$ of (4.22).[4]
Again, the fact that the partially cutoff optimal control problem with horizon K can be regarded as a separate complete optimal control problem with horizon K (see discussion above) together with the linear feedback form (4.21) of the optimal control sequence u^* from Theorem 4.2 implies that in (4.26)

$$\tilde{u}^* = -K_{N-K+1}\bar{x}_{N-K+1} \tag{4.27}$$

holds for some $K_{N-K+1} \in \mathbb{R}^{m\times l}$ independent of $\bar{x}_{N-K+1}$, even if $\bar{x}_{N-K+1}$ cannot be reached at time step $N-K+1$ by starting in some $\bar{x}_1 \in \mathbb{R}^l$ and applying the first $N-K$ controls of the sequence $u^* = (u_n^*)_{n\in\{1,\ldots,N-1\}}$ that is optimal for horizon N.
However, if this is the case, i.e., $\bar{x}_{N-K+1} = x_{N-K+1}^{\bar{x}_1,(u_1^*,\ldots,u_{N-K}^*)}$, then obviously $\tilde{u}^* = u_{N-K+1}^*$ holds, i.e. the single-step optimal control coincides with that of the complete trajectory. Repeated use of this observation immediately implies that "tails of optimal trajectories are again optimal".
Finally,

$$V_N^1(\bar{x}_N) = \bar{x}_N^\top Q_N \bar{x}_N \tag{4.28}$$

holds for any $\bar{x}_N \in \mathbb{R}^l$ by definition, especially for $\bar{x}_N = x_N^{\bar{x}_1,(u_1,\ldots,u_{N-1})}$ with some initial state $\bar{x}_1 \in \mathbb{R}^l$ and some control sequence $\bar{u} := (u_n)_{n\in\{1,\ldots,N-1\}} \subset \mathbb{R}^m$.
Based on this principle, we can prove the following statement:

Theorem 4.4. *For any initial state $\bar{x}_1 \in \mathbb{R}^l$, the optimal value of the objective function J of the optimal control problem* (4.1) *is given by*

$$V_N^N(\bar{x}_1) = J_N(x, u^*) = \bar{x}_1^\top S_1 \bar{x}_1 \tag{4.29}$$

with optimal control sequence $u^* = (u_n^*)_{n\in\{1,\ldots,N-1\}} \subset \mathbb{R}^m$ from (4.2) *and with S_1 from the solution to the Discrete Riccati Equations* (4.3).

Proof. The first equality directly follows from Theorem 4.2 and Definition 4.3. We prove the second equality (or rather $V_N^N(\bar{x}_1) = \bar{x}_1^\top S_1 \bar{x}_1$) by induction with respect to $K \in \{1,\ldots,N\}$. As can be easily seen by definition,

$$V_N^1(\bar{x}_N) = \bar{x}_N^\top Q_N \bar{x}_N = \bar{x}_N^\top S_N \bar{x}_N$$

holds for any $\bar{x}_N \in \mathbb{R}^l$.
We assume that for some arbitrary but fixed $K \in \{1,\ldots,N-1\}$

$$V_N^K(\bar{x}_{N-K+1}) = \bar{x}_{N-K+1}^\top S_{N-K+1} \bar{x}_{N-K+1} \tag{4.30}$$

[4]An alternative proof of Theorem 4.2, which does entirely without Lagrange multipliers, is mainly based on this recursive formula (4.26) and can, e.g., be found in [38, Section 6.4.3].

holds for any $\bar{x}_{N-K+1} \in \mathbb{R}^l$ with symmetric $S_{N-K+1} \in \mathbb{R}^{l\times l}$ from the solution to the Discrete Riccati Equations.
Then for $\tilde{u}^* \in \mathbb{R}^m$ first control of the optimal control sequence $u^{K,*} = (u_n^*)_{n\in\{N-K,\dots,N-1\}}$ regarding $J_N^{K+1}(x^K, u^K)$ with $x_{N-K} = \bar{x}_{N-K}$, i.e.,

$$\tilde{u}^* = -K_{N-K}\bar{x}_{N-K} \tag{4.31}$$

instead of (4.27), the following holds for any $\bar{x}_{N-K} \in \mathbb{R}^l$:

$$\begin{aligned}
V_N^{K+1}(\bar{x}_{N-K}) &\overset{(4.26)}{=} \bar{x}_{N-K}^\top Q_{N-K}\bar{x}_{N-K} + (\tilde{u}^*)^\top R_{N-K}\tilde{u}^* + V_N^K\left(x_{N-K+1}^{\bar{x}_{N-K},(\tilde{u}^*)}\right) \overset{(4.30)}{=} \\
&\overset{(4.30)}{=} \bar{x}_{N-K}^\top Q_{N-K}\bar{x}_{N-K} + (\tilde{u}^*)^\top R_{N-K}\tilde{u}^* + \left(x_{N-K+1}^{\bar{x}_{N-K},(\tilde{u}^*)}\right)^\top S_{N-K+1}\left(x_{N-K+1}^{\bar{x}_{N-K},(\tilde{u}^*)}\right) \overset{(4.1b)}{=} \\
&\overset{(4.1b)}{=} \bar{x}_{N-K}^\top Q_{N-K}\bar{x}_{N-K} + (\tilde{u}^*)^\top R_{N-K}\tilde{u}^* + \left(\bar{x}_{N-K}^\top A^\top + (\tilde{u}^*)^\top B^\top\right) S_{N-K+1}(A\bar{x}_{N-K} + B\tilde{u}^*) \overset{(4.31)}{=} \\
&\overset{(4.31)}{=} \bar{x}_{N-K}^\top \Big(Q_{N-K} + A^\top S_{N-K+1}A + K_{N-K}^\top(R_{N-K} + B^\top S_{N-K+1}B)K_{N-K} - \\
&\qquad - A^\top S_{N-K+1}BK_{N-K} - K_{N-K}^\top B^\top S_{N-K+1}A\Big)\bar{x}_{N-K} \overset{(4.2),(*)}{=} \\
&\overset{(4.2),(*)}{=} \bar{x}_{N-K}^\top \Big(Q_{N-K} + A^\top S_{N-K+1}A + A^\top S_{N-K+1}B(R_{N-K} + B^\top S_{N-K+1}B)^{-1}B^\top S_{N-K+1}A - \\
&\qquad -2A^\top S_{N-K+1}B(R_{N-K} + B^\top S_{N-K+1}B)^{-1}B^\top S_{N-K+1}A\Big)\bar{x}_{N-K} = \\
&= \bar{x}_{N-K}^\top \Big(Q_{N-K} + A^\top S_{N-K+1}A - \\
&\qquad - A^\top S_{N-K+1}B(R_{N-K} + B^\top S_{N-K+1}B)^{-1}B^\top S_{N-K+1}A\Big)\bar{x}_{N-K} \overset{(4.3)}{=} \\
&\overset{(4.3)}{=} \bar{x}_{N-K}^\top S_{N-K}\bar{x}_{N-K}.
\end{aligned}$$

In $(*)$, the symmetry of S_{N-K+1} and the resulting symmetry of $(R_{N-K} + B^\top S_{N-K+1}B)$, which implies $(R_{N-K} + B^\top S_{N-K+1}B)^{-\top} = (R_{N-K} + B^\top S_{N-K+1}B)^{-1}$, were used.
Hence, the induction hypothesis (4.30) is true for all $K \in \{1,\dots,N\}$.
The last case $K = N$ thus yields

$$V_N^N(\bar{x}_1) = \bar{x}_1^\top S_1\bar{x}_1 \tag{4.32}$$

for all $\bar{x}_1 \in \mathbb{R}^l$, which proves the claim. □

However, it is important to emphasize that the definition of the optimal controls u_n^* requires perfect observation of the states x_n. Even though the linear feedback form allows to compute the optimal feedback gain matrices K_n without any information about the states x_n, at every time step n the state x_n resulting from the previous state x_{n-1} and the last applied control u_{n-1}^* needs to be available to proceed with the computation of next optimal control $u_n^* = -K_nx_n$. While, technically, in the LQR model only information about the system dynamics matrices A and B are required to compute the following state for any chosen

control, this actually means a perfect and immediate observation, which at least in human actions is never given. Instead, it would be more convenient to distinguish between the system state $x_n \in \mathbb{R}^l$ and the **system output** $y_n \in \mathbb{R}^s$, which, e.g., might also be linear in the state, i.e., $y_n = Cx_n$ holds for any $C \in \mathbb{R}^{s\times l}$, and to force the feedback control $u_n \in \mathbb{R}^m$ to depend on y_n rather than x_n.
Moreover, different types of noise could easily be added, leading to the stochastic extension of the LQR scheme, the **Linear-Quadratic Gaussian Controller (LQG)** (see, e.g., [2], [65], or [7, Chapter 14]), which is able to model optimal behavior under uncertainty. However, stochastic processes are beyond the scope of this thesis.

4.2 Modifications and Application to Pointing Tasks

In this section, we want to explain the modifications to the LQR scheme that we make in order to use it for human-computer interaction tasks. For this reason, we first need to define what exactly we mean by a "pointing task", which is exemplarily used in this thesis.

4.2.1 Pointing Tasks

Pointing is a very frequently used movement of both humans and animals. Already thousands of years ago, pointing to objects or living beings was used to draw each other's attention to dangers or to communicate with each other. Since the beginning of the digital age, the field of application has changed considerably: Nowadays, we constantly move our fingers to point to a specific display area on our smartphone or tablet and make a specific request, such as "Open this website" or "Send the message", to articulate our needs. While the types of devices we interact with are becoming more diverse as digitalization progresses, the standard device for communicating with computers is currently still the mouse.
Here, users move the physical mouse device, which is placed on the flat two-dimensional surface of a pad or a table, in some direction. Optical sensors and accelerometers within the device then measure the acceleration in short time intervals, i.e., with high frequency, and transmit these information to the computer. Some filters might be applied to smoothen the data and to get rid of noise before a tranfer function maps the observed physical acceleration to the acceleration of the virtual mouse cursor, i.e., it defines the transformation between physical and virtual coordinates.[5] Finally, two integrators lead to the new mouse pointer position, which the user can observe from the display. The entire interaction process is shown graphically in Figure 4.1.

Since it is hardly possible to give a clear definition of what is regarded as a coherent pointing task in the everyday use of computer devices, the term originates mainly from experiments

[5]Often, a simple *control-display gain*, i.e., a linear transfer function leading to a constant ratio of virtual mouse cursor movements to physical mouse device movements, is used.

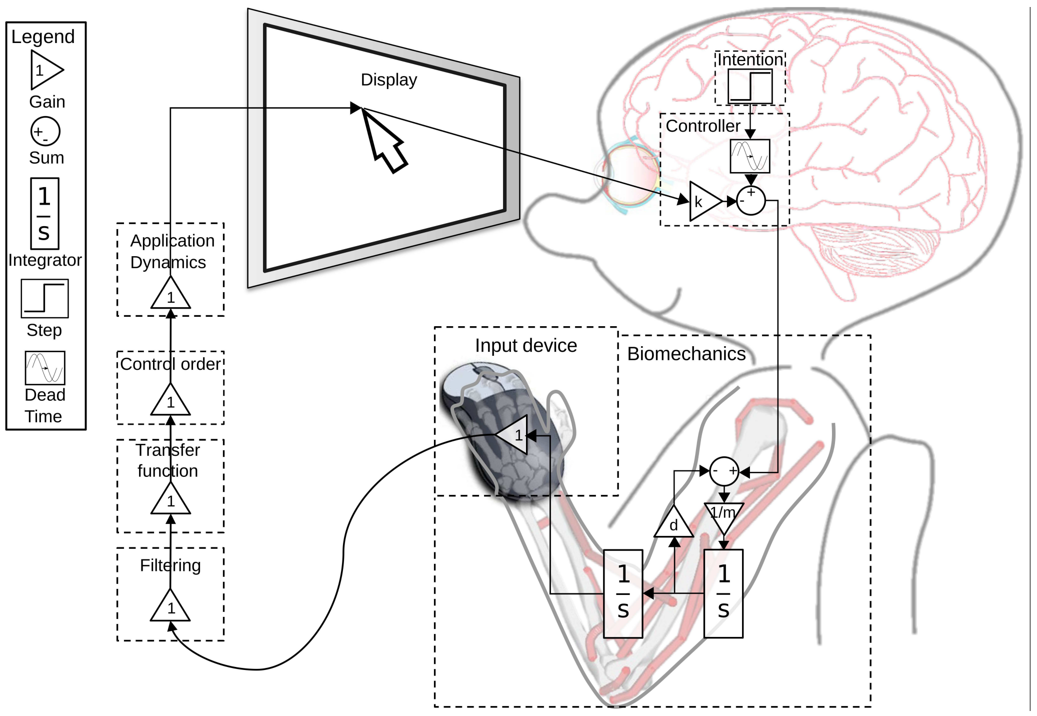


Figure 4.1: *Graphical representation of the human-computer interaction process using our pointing task as an example [50]. Here, the interpretation of the biomechanical apparatus largely corresponds to that of the 2OL-Eq model.*

in which human pointing behavior is investigated. Here, clear task instructions are required to be able to interpret the results reasonably.
Two main types of such (experimental) **pointing tasks** need to be distinguished:
In **isolated** pointing tasks, the mouse rests at an initial position at the beginning of the task. The target area, towards which the user is asked to move the mouse, only appears at the beginning of the task, i.e., the user is not aware of the target position before he is asked to move, which explicitly allows to take **reaction time** effects into account.
In contrast to this there are **reciprocal** pointing tasks [22]: Here, users are asked to move from an initial area to the displayed target area and back again, several times in quick succession. Thus, at least after an initial "training phase" of multiple trials, both targets are well-known and therefore the required movement can be optimized. In particular, *choice reaction time* [19, 53] in the sense of a delay due to a previously unknown target configuration which can only be observed through received input signals should not exist in reciprocal pointing tasks. However, it is still unclear how perceived sensory inputs are exactly processed and combined with pre-planned movement strategies or templates [52, 10], which complicates the interpretation of the nevertheless existing delays at the beginning of movements.

Typically, a pointing task needs to be terminated by a click within the target area, which requires extra effort and time. Therefore, we distinguish between the main **surge phase** of

a movement and the subsequent **correction phase**.
In addition, pointing tasks can be differentiated according to the dimension used. Everyday pointing tasks typically allow **two-dimensional** movements, e.g., moving the mouse device on the surface of a table or moving the finger on the display of a smartphone. However, both **one-** and **three-dimensional** tasks are also conceivable: While air pointing in the three-dimensional space (e.g., in connection with virtual reality) could serve as a useful input facility, in some experiments it might be reasonable to transfer only a one-dimensional component of the mouse device measurements to the displayed mouse cursor, i.e., it is physically possible to move in two orthogonal directions, but only one direction is used.

In our case, the experimental data we later use for model comparisons is based on a one-dimensional reciprocal pointing task. Here, users are asked to move to the target, which is displayed at a horizontal distance of D meters, as accurate and fast as possible and click on it. The accuracy criterion is met if and only if they click within the rectangular target box, which has a width of W meters. At the time of the click the new target box of the same width, which surrounds the initial position, is highlighted and the entire distance needs to be covered backwards with the same requirements. This reciprocal task is repeated several times in a row, i.e., without any break. More details on the experiment can be found in Section 5.2.
However, in the following we refer to a single movement between initial and target area (or vice versa) when using the terms "pointing task" and "trial". In addition, we formulate all statements explicitly for the one-dimensional (1D) case, while the necessary adjustments for the two-dimensional (2D) and three-dimensional (3D) case are attached in Section 4.3.1.

4.2.2 System Dynamics

In our model, the biomechanical apparatus of the limb is assumed to be the *spring-mass-damper system* depicted in Figure 3.3. Here, the combination of an active agonist muscle, which is assumed to directly realize the applied force, and a passive antagonist muscle modeled by a spring and a dashpot, which compensates the direct force, leads to angular motions of the limb. Again, the assumption of small movements relative to the length of the limb allows us to linearize the model and thus feature straight motions.

Note that in the previously presented 2OL-Eq, which is described by (2OL-Eq) with $u \equiv T$, this fixed "control" u implies that the acceleration is proportional to the remaining distance to target $T - x(t)$ with factor $k > 0$, apart from the additional damping term $-d\dot{x}(t)$. Regarding our model, we want to generalize this approach by omitting the "equilibrium-control" assumption, which states that the applied force necessarily corresponds to kT. More precisely, the underlying system dynamics of our model is the second-order lag equation

$$\ddot{p}(t) = u(t) - kp(t) - d\dot{p}(t), \tag{2OL}$$

where $p(t)$ denotes the position of the mouse pointer and $u(t)$ denotes the applied control, each at time $t \in I := [0, t_f]$. This allows us to model different behavior through different controls u, with $u(t) := kT$ corresponding to 2OL-Eq again. Since $u(t)$ rather than $ku(t)$ is included as applied force term, this model is often referred to as *direct force control* rather than *equilibrium control* (like the 2OL-Eq model).
Note that in our case $p(t) \in \mathbb{R}$ and thus $u(t) \in \mathbb{R}$ applies since 1D pointing tasks are regarded.

In order to transform this system into a first-order linear control system of the form (2.4), we must include both the position $p(t)$ and the velocity $v(t) := \dot{p}(t)$ in the states $x(t)$ of the resulting system, i.e.,

$$x(t) := \begin{bmatrix} p(t) \\ v(t) \end{bmatrix} \in \mathbb{R}^2, \quad t \in I. \tag{4.33}$$

Then, (2OL) is equivalent to

$$\dot{x}(t) = \tilde{A}x(t) + \tilde{B}u(t), \quad t \in I, \tag{4.34a}$$

with

$$\tilde{A} := \begin{bmatrix} 0 & 1 \\ -k & -d \end{bmatrix} \in \mathbb{R}^{2\times 2}, \quad \tilde{B} := \begin{bmatrix} 0 \\ 1 \end{bmatrix} \in \mathbb{R}^{2\times 1}. \tag{4.34b}$$

Because

$$\operatorname{rank}(\begin{bmatrix} \tilde{B} & \tilde{A}\tilde{B} \end{bmatrix}) = \operatorname{rank}\left(\begin{bmatrix} 0 & 1 \\ 1 & -d \end{bmatrix} \right) = 2$$

holds for each $d > 0$ (even for $d \in \mathbb{R}$), this system is completely controllable according to Theorem 2.6, i.e., starting at time $t_0 = 0$ in some arbitrary state $x_0 \in \mathbb{R}^2$ (due to (2.7)), any state $\bar{x} \in \mathbb{R}^2$ can be reached in arbitrary time $\tau > 0$ by applying the appropriate piecewise continuous control function $u \in \mathcal{U}$. In particular, the asymptotic stabilization problem for (4.34) is solvable according to the Pole-Shifting Theorem (Theorem 2.9).

However, we need to slightly modify (4.34) for some technical reasons: We additionally incorporate the constant[6] target position $T \in \mathbb{R}$ in the state vectors to be able to compute the distance to the target later. Precisely, we use the states

$$x(t) := \begin{bmatrix} p(t) \\ v(t) \\ T \end{bmatrix} \in \mathbb{R}^3, \quad t \in I, \tag{4.35}$$

[6]For pointing tasks with moving targets a non-constant target position function $T(t)$, which is solution to the linear differential equation $\dot{T} = A_T T$ with some $A_T \in \mathbb{R}$, could be incorporated analogously.

rather than those defined in (4.33) and the respective system dynamics

$$\dot{x}(t) = \tilde{A}x(t) + \tilde{B}u(t), \quad t \in I, \tag{4.36a}$$

with

$$\tilde{A} := \begin{bmatrix} 0 & 1 & 0 \\ -k & -d & 0 \\ 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 3}, \quad \tilde{B} := \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \in \mathbb{R}^{3\times 1}, \tag{4.36b}$$

which result from (4.34) with additional zero row in $\tilde{A}$ and $\tilde{B}$ due to $\dot{T} \equiv 0$ and additional zero column in $\tilde{A}$, since $x(t)$ is not directly affected by T.
For the sake of completeness, we note that complete controllability does not hold for (4.36) anymore because of $\operatorname{rank}(\begin{bmatrix} \tilde{B} & \tilde{A}\tilde{B} & \tilde{A}^2\tilde{B} \end{bmatrix}) = 2 < 3$. However, this was to be expected since the third component of any state $x(t)$ necessarily corresponds to the fixed target position T and thus cannot be controlled. Nevertheless, the stabilization problem for (4.36) (but not the asymptotic stabilization problem) is solvable according to Theorem 2.9 because $\tilde{A}$ and $\tilde{B}$ are already of the form (2.10) with $A_3 = 0$, i.e., condition *(c)* is fulfilled.

Since the measured mouse data is available in discrete time, we use time-discrete dynamics. We particularly emphasize that this restricts the admissible control functions u to piecewise constant functions, which are identified with the respective sequence of their finitely many values, i.e., $u = (u_n)_{n\in\{1,\dots,N-1\}}$.
However, instead of using the exact transformation formula (2.14) to define the corresponding time-discrete control system, we approximate the time derivative on the left-hand side of (4.36a) using **forward differences**, i.e., we use

$$\dot{x}(t) := \lim_{\tau\to t} \frac{x(\tau) - x(t)}{\tau - t} \approx \frac{x(t+h) - x(t)}{h} \tag{4.37}$$

with some small sampling time $h > 0$, which is often referred to as **Euler method** [9].
Moreover, we discretize the considered time interval $I := [0, t_f]$ with this sampling time, i.e., we evaluate x only at the times $\{(n-1)h \mid n \in \{1,\dots,N\}\}$, where $(N-1)h = t_f$ applies, and denote the respective values by $x_n := x((n-1)h)$, $n \in \{1,\dots,N\}$. Together, this yields the time-discrete approximation of (4.36)

$$x_{n+1} = x_n + h(\tilde{A}x_n + \tilde{B}u_n), \quad n \in \{1,\dots,N-1\}. \tag{4.38a}$$

To obtain a unique solution trajectory, x_1 needs to satisfy the initial value condition

$$x_1 = \bar{x}_1 \tag{4.38b}$$

for some given $\bar{x}_1 \in \mathbb{R}^3$. Using (4.36b), the resulting *time-discrete linear control system with sampling time* $h > 0$ is thus given by

$$\begin{aligned} x_{n+1} &= Ax_n + Bu_n, \quad n \in \{1,\dots,N-1\}, \\ x_1 &= \bar{x}_1, \end{aligned} \tag{4.39a}$$

with

$$A := (\mathrm{ID}_3 + h\tilde{A}) = \begin{bmatrix} 1 & h & 0 \\ -hk & 1-hd & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad B := h\tilde{B} = \begin{bmatrix} 0 \\ h \\ 0 \end{bmatrix}. \tag{4.39b}$$

Note that this dynamic, which is similar to the one used by Todorov [67], is nothing but a discrete approximation of (2OL), where the states are additionally augmented by the constant target center T.
In the next section, we make some assumptions on the used controls $(u_n)_{n\in\{1,\dots,N-1\}}$, which are (in contrast to those of 2OL-Eq) not yet clearly defined.

4.2.3 Optimality Criteria

The basic approach of our **2OL-LQR** model is to use a combination of the second-order lag (2OL) and an optimization-based model such as MinJerk (see Corollary 3.2). This means that we represent the biomechanical apparatus by the (linearized) spring-mass-damper system depicted in Figure 3.3 and assume that the controls are optimally chosen with respect to some objective function to be defined.
Fortunately, LQR provides the suitable framework for this approach: The previously derived system dynamics (4.39) are exactly of the required form (4.1b), and using a quadratic objective function as in (4.1a) allows us to assign appropriate objectives to both the state cost matrices Q_n, $n \in \{1,\dots,N\}$, and the control cost matrices R_n, $n \in \{1,\dots,N-1\}$.
The following assumptions about reasonable optimality criteria for pointing tasks are mainly based on Todorov's extension of the Linear-Quadratic Gaussian Controller (LQG), which allows to consider a variety of aspects such as visual-proprioceptive sensory input with additive state-dependent Gaussian noise, motor control dynamics with even temporally correlated additive and multiplicative noise, and time delays [67].[7] In this thesis, however, we want to investigate the limitations of the deterministic LQR framework.

From the considered pointing tasks, which might be summarized by instructions like "Steer the virtual mouse cursor to the displayed rectangular target box by moving the physical mouse device and click on the target box as fast as possible", we believe that the following conclusions can be drawn about the users' understanding of the task requirements:

1. Users aim to maximize the accuracy of their movement, i.e., they aim to minimize the distance between mouse cursor and target.
2. Users aim to minimize the duration of their movement.
3. Users aim to minimize the effort required to fulfill the task.

We now discuss these three assumptions one after another.

[7]The interested reader might also consider Kleinman's model presented in [36] and explained and motivated in detail in [35, 4] and Qian's model [57], which both are based on very similar assumptions.

The accuracy assumption is quite indisputable as it can be directly derived from the task instructions. In particular, it would be straightforward to make use of a cost function which penalizes the distance between the mouse pointer and the target center in the following way: Any mouse position outside the target box is penalized with the same constant weight and any mouse position inside the target box is not penalized at all. Unfortunately, such a cost function would necessarily be non-quadratic in the remaining distance to the target center and even discontinuous at the boundary of the target box. Therefore, it cannot be used in our approach, which is based on the LQR scheme because of its numerous advantages, especially the linear feedback form of the optimal controls (4.21).
In order to obtain such a quadratic penalization of the remaining distance to the target center, we just need to set

$$Q_n := \omega_n \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \tag{4.40}$$

with weights $\omega_n \in \mathbb{R}$ for some desired time steps $n \in \{1, \dots, N\}$ (for further discussions on the concrete choice of these time steps see Section 6.1), which leads to the cost terms

$$x_n^\top Q_n x_n = \begin{bmatrix} \omega_n(p_n - T) & 0 & -\omega_n(p_n - T) \end{bmatrix} \cdot \begin{bmatrix} p_n \\ v_n \\ T \end{bmatrix} = \omega_n (p_n - T)^2 \tag{4.41}$$

in the objective function (4.1a).
It remains to define the weights ω_n. One way would be to choose them proportional to the width W of the rectangular target box, as suggested by Torodov [67]. Setting, e.g.,

$$\omega_n := \frac{4}{W^2}, \tag{4.42}$$

for each $n \in \{1, \dots, N\}$ implies

$$x_n^\top Q_n x_n = \left(\frac{2|T - p_n|}{W} \right)^2 \tag{4.43}$$

for each $n \in \{1, \dots, N\}$, which is closely linked to the ID from (3.1) since $D_n := |T - p_n|$ is exactly the *remaining* distance[8] to the target center at the time step n. In particular, (4.43) becomes one exactly when $D_n = \frac{W}{2}$ holds, i.e., when the boundary of the rectangular target box is reached. Although the logarithm from the actual definition of the ID is missing in this formulation[9], a link to Fitts' law (3.3) or rather Kvålseth's law (3.6) can be established.

[8] Here and in the further course of this thesis we refer to $|\cdot|$ as the Euclidean norm $\|\cdot\|_2$ (in the one-dimensional case all p-norms coincide anyway).

[9] Note that even though this is due to technical requirements of the LQR scheme, the logarithm should not be used for positions inside the target box anyway since (4.43) is less than one there, i.e., the logarithm would yield negative values.

The state cost term (4.43) can thus be interpreted as a penalization of the task's remaining difficulty at the time step n, depending on the current position of the mouse.
However, the inclusion of the target width W in the cost weights would result in an increase of the positional error costs (4.43) as the ID of the task increases. While the optimal trajectory with respect to an objective function including only state cost terms would not be affected by constant multiplicative weights anyway, the solution to our optimal control problem (4.1), where particularly $R_n \neq 0$ needs to hold for all $n \in \{1, \ldots, N\}$, would be artificially biased by such weights, which cannot be justified.[10] For this reason, we decided to set ω_n independent of the target configuration and, for the sake of simplicity, thus defined

$$\omega_n := 1 \tag{4.44}$$

for all $n \in \{1, \ldots, N\}$. The resulting positional cost terms are supported by prior work, for example by Diedrichsen [16] and Shadmehr [63, 64].
Finally, we want to emphasize that such a continuous cost function penalizing the distance to target was not only required for technical reasons, but there is also some evidence for it: First of all, the information about the terminal position of the trajectory is kept vague, i.e., it is not clear whether the distance to the center or to the nearest edge of the target box should be minimized. Intuitively, one would suspect the latter since the task instruction only distinguishes between "inside the box" and "outside the box". However, it is reasonable to assume that users do not aim to stop their motion at the first possible opportunity, i.e., at the nearest target boundary, to not risk missing the target due to the effects of small positional deviations (e.g., due to erroneous muscle control), which can occur even under the assumption of perfect observation. In particular, this means that users stop their motion *within* the target box (although not necessarily at its center).[11] Continuous transitions of our state cost functions at the boundaries of the target box are thus quite reasonable as there is no "binary" objective to the users.

The second assumed objective of users during pointing tasks is the minimization of the total duration of their movement. While in principle, it is possible to add the duration to the objective function (as, e.g., done in continuous time in Hoff's *minimum jerk/minimum time model* [31] or in discrete time in Shadmehr's *cost of time model* [64]), it is important to clarify that, above all, the ratio of the individual summands of the objective function to each other matters (note that multiples of the entire objective function do not change its minimizer). This implies that reasonable weights of the other objectives characterizing their

[10]The additional optimization of the ratio of state costs to remaining costs as described in Section 5.1 would allow to compensate such a bias. Then, however, it would be this ratio that increases without justification as the ID increases.

[11]It was shown that at least for sufficiently large targets, users save some time by stopping at a short fraction before the target center (but not outside the target box, of course), i.e., they slightly *undershoot* the target. However, this effect is comparatively small relative to the distance between initial and target positions [50, 22]. An alternative ID formulation taking this observation into account is presented in [26, 27].

"importance" relative to the duration of the movement need to be found. In Hoff's model, e.g., this is done by using a specific trade-off function that provides a relationship between the movement duration and the jerk weight used in the objective function (given the initial distance) in order to determine the jerk weight that matches a reasonably expected duration and fix this weight for the considered task.
However, we decided for a different, more implicit way to take the total time of the movement into account: Penalizing the distance to target throughout the entire movement creates an incentive to reach (and stay at the target) sooner rather than later because $x_n^\top Q_n x_n = 0$ holds for the corresponding states $x_n = [T, v_n, T]^\top$ with arbitrary velocity $v_n \in \mathbb{R}$ (and only $v_n = 0$ yields $x_i = x_n$ for all $i \in \{n+1, \dots, N\}$ according to (4.35) and (4.39), i.e., with $v_n \neq 0$, costs will be incurred again in the following steps).
We strongly emphasize that with such a cost structure also one of the major drawbacks of Todorov's approach [67] is bypassed: In his model, the overall movement duration must always be known in advance. For *via-point tasks* (see Section 4.3.2 for details), this assumption might be less demanding since the via-point click times could be optimized in an additional outer loop. However, Todorov needs to assume the same movement duration for any trial of the same task, even of different participants (otherwise each participant would need his/her own model). On the contrary, in our model the penalization of the *remaining* distance to the target during the movement allows to interpret the still required final time step N as the *maximum permissible time step* (which thus could be chosen arbitrarily large in theory), while it is possible to "finish" the task at a much earlier time step (note that the click, which determines the end of the movement, is usually not modeled explicitly).
Although we have already roughly anticipated the structure of the state cost terms, in Section 6.1 we take a closer look at the effects of the different points in time at which the positional error between the mouse pointer and the target is penalized.

The third assumed objective is the minimization of effort during the execution of the task. The reason for this is quite self-explanatory: Among all options that lead to the same desired result, users will always opt for the one with the least exertion.
Since our time-discrete model is based on piecewise constant controls, it is evident that large changes between the single control values require more effort. Thus, the control cost matrices R_n should penalize the changes in control $u_n - u_{n-1}$ rather than the controls u_n itself, which leads to an objective function of the form

$$J_N(x,u) := \sum_{n=1}^{N} x_n^\top Q_n x_n + \sum_{n=1}^{N-1} (u_n - u_{n-1})^\top R_n (u_n - u_{n-1}), \tag{4.45}$$

where an initial **reference control** $u_0 \in \mathbb{R}$ is required now.
There is another justification for adding these terms to the objective function:
Since our model considers the biomechanical apparatus as linearized spring-mass-damper system, users are assumed to control the acceleration of their movements by choosing

$(u_n)_{n\in\{1,\dots,N-1\}}$ in (4.39) appropriately. However, as Todorov [67] emphasizes, if any "energy-like" terms are minimized in order to keep some biomechanical activity[12] to a minimum, it should be terms corresponding to jerk, i.e., the time derivative of acceleration, rather than to acceleration itself.[13] In our time-discrete model with sampling time $h > 0$ such terms are given by the backward differences in control divided by h, i.e., $j_n := \frac{u_n - u_{n-1}}{h}$. These terms can be included in the objective function (4.45) by setting

$$R_n := \frac{r_n}{h^2}, \quad r_n > 0, \tag{4.46}$$

where r_n must take positive values because of the required positive definiteness of the control cost matrices R_n (which are simply scalars in our case). Using (4.46), the third key assumption, minimization of effort, can thus be interpreted as jerk minimization.[14]
Note that it is the underlying LQR scheme which, in contrast the MinJerk model introduced in Section 3.3.1, allows us to combine these different objectives with arbitrary weights and also make assumptions about the biomechanical system.

4.2.4 2OL-LQR and its Analytical Solution

In summary, our **2OL-LQR model** can be described as a time-discrete linear-quadratic optimal control problem with finite horizon $N \in \mathbb{N}$, where users are assumed to optimize accuracy, duration, and effort of their movement, and which is constrained by the approximated second-order lag dynamics (4.39) used to describe the biomechanical apparatus.
It is therefore given in its most general form by

$$\begin{gathered}\textit{Minimize} \quad J_N(x,u) := \sum_{n=1}^{N} x_n^\top Q_n x_n + \sum_{n=1}^{N-1} (u_n - u_{n-1})^\top R_n (u_n - u_{n-1}) \\ \textit{with respect to } u := (u_n)_{n\in\{1,\dots,N-1\}} \subset \mathbb{R} \textit{ for all } \bar{x}_1 \in \mathbb{R}^3,\ \bar{u}_0 \in \mathbb{R},\end{gathered} \tag{4.47a}$$

where $x := (x_n)_{n\in\{1,\dots,N\}} \subset \mathbb{R}^3$ with $x_n := [p_n, v_n, T]^\top$ satisfies

$$\begin{aligned} x_{n+1} &= Ax_n + Bu_n, \quad n \in \{1,\dots,N-1\}, \\ x_1 &= \bar{x}_1, \end{aligned} \tag{4.47b}$$

[12]The question of what is actually to be influenced at the lowest level of biomechanical motor control (neuronal firing, muscle tension, etc.) is still unclear. A good overview of the current state of research can be found in [51].

[13]For a discussion on the different effects of minimizing arbitrary time derivatives of velocity, see [59].

[14]Actually, jerk is defined as the time derivative of the end-effector's acceleration. However, with our interpretation of the biomechanical apparatus, which is depicted in Figure 3.3, it is reasonable to associate jerk with the change in applied force, i.e., the change in the control that is exerted on the system by humans. Minimizing jerk thus corresponds to minimizing muscle excitation.

with sampling time $h > 0$ and system dynamics matrices

$$A := \begin{bmatrix} 1 & h & 0 \\ -hk & 1-hd & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad B := \begin{bmatrix} 0 \\ h \\ 0 \end{bmatrix}, \tag{4.47c}$$

and where $u_0 = \bar{u}_0$ applies.

The objective function J is made up of two parts:
On the one hand, there are quadratic state cost terms $x_n^\top Q_n x_n$ at each step $n \in \{1, \dots, N\}$, where Q_n is either defined by

$$Q_n := \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \tag{4.48}$$

or Q_n is set to 0. In the former case, $x_n^\top Q_n x_n = D_n^2$ applies with $D_n := |T - p_n|$ as the *remaining* distance to target center, i.e., the positional error at the time step n is quadratically penalized. In the latter case, the positional error at time step n is considered irrelevant.
On the other hand, changes in control are also quadratically penalized by some positive R_n of the form

$$R_n := \frac{r_n}{h^2}, \quad r_n > 0, \tag{4.49}$$

which yields $(u_n - u_{n-1})^\top R_n (u_n - u_{n-1}) = r_n (\frac{u_n - u_{n-1}}{h})^2$, i.e., the squares of the "jerk" terms $j_n := \frac{u_n - u_{n-1}}{h}$ are penalized with the jerk weights r_n.
As previously mentioned, we will extensively discuss the concrete use of these two cost terms at different time steps n in Section 6.1.

Our modified objective function J_N, in which the change of controls instead of the controls themselves is penalized, requires a modified solution formula. To this end, we need to introduce **information vectors** $\mathcal{I}_n$, $n \in \{1, \dots, N\}$, which incorporate all information that are both available at time step n and required for the computation of the remaining optimal controls $u_{n+1}^*, \dots, u_{N-1}^*$. As we will see, at each time step $n \in \{1, \dots, N\}$ it is sufficient to only include the most recent control u_{n-1} in the respective information vector, i.e.,

$$\mathcal{I}_n := \begin{bmatrix} x_n \\ u_{n-1} \end{bmatrix} \in \mathbb{R}^4, \quad n \in \{1, \dots, N\}. \tag{4.50}$$

Moreover, we define

$$\mathrm{I}_x := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \in \mathbb{R}^{3\times 4}, \quad \mathrm{I}_u := \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{1\times 4}, \tag{4.51}$$

which implies

$$\mathrm{I}_x \mathcal{I}_n = x_n \in \mathbb{R}^3, \quad \mathrm{I}_u \mathcal{I}_n = u_{n-1} \in \mathbb{R}, \quad n \in \{1, \dots, N\}, \tag{4.52}$$

i.e., I_x respective I_u are the matrices that extract the state x_n respective the control u_{n-1} from the information vector $\mathcal{I}_n$ for any $n \in \{1, \dots, N\}$.

We can rewrite both our system dynamics equations (4.47b) and our objective function $J_N(x, u)$ in terms of information vectors, which yields the following optimal control problem equivalent to (4.47):

$$\begin{aligned} \textit{Minimize} \quad & \mathcal{J}_N(\mathcal{I}, u) := \sum_{n=1}^{N} \mathcal{I}_n^\top \mathcal{Q}_n \mathcal{I}_n + \sum_{n=1}^{N-1} (u_n - u_{n-1})^\top R_n (u_n - u_{n-1}) \\ & \textit{with respect to } u := (u_n)_{n \in \{1,\dots,N-1\}} \subset \mathbb{R} \textit{ for all } \bar{x}_1 \in \mathbb{R}^3,\ \bar{u}_0 \in \mathbb{R}, \end{aligned} \tag{4.53a}$$

where $\mathcal{I} := (\mathcal{I}_n)_{n \in \{1,\dots,N\}} \subset \mathbb{R}^4$ with $\mathcal{I}_n := [x_n, u_{n-1}]^\top$ satisfies

$$\begin{aligned} \mathcal{I}_{n+1} &= \mathcal{A}\mathcal{I}_n + \mathcal{B}u_n, \quad n \in \{1, \dots, N-1\}, \\ \mathcal{I}_1 &= \bar{\mathcal{I}}_1 := \begin{bmatrix} \bar{x}_1 \\ \bar{u}_0 \end{bmatrix}, \end{aligned} \tag{4.53b}$$

with sampling time $h > 0$ and system matrices

$$\begin{aligned} \mathcal{A} &:= \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & h & 0 & 0 \\ -hk & 1-hd & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathcal{B} := \begin{bmatrix} B \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ h \\ 0 \\ 1 \end{bmatrix}, \\ \mathcal{Q}_n &:= \begin{bmatrix} Q_n & 0 \\ 0 & 0 \end{bmatrix}, \quad n \in \{1, \dots, N\}, \end{aligned} \tag{4.53c}$$

and where $u_0 = \bar{u}_0$ applies.

With these equivalent formulations, we can prove the following theorem:

Theorem 4.5 (2OL-LQR). *The unique solution $u^* = (u_n^*)_{n \in \{1,\dots,N-1\}}$ to the 2OL-LQR optimization problem* (4.47) *is given by*

$$\begin{aligned} u_n^* &:= -\mathcal{K}_n \mathcal{I}_n^*, \quad n \in \{1, \dots, N-1\}, \\ \mathcal{K}_n &= (R_n + \mathcal{B}^\top \mathcal{S}_{n+1} \mathcal{B})^{-1} (\mathcal{B}^\top \mathcal{S}_{n+1} \mathcal{A} - R_n \mathrm{I}_u), \quad n \in \{1, \dots, N-1\}, \end{aligned} \tag{4.54}$$

with $\mathcal{A}, \mathcal{B}$ from (4.53c), *where the symmetric matrices $\mathcal{S}_n \in \mathbb{R}^{4 \times 4}$ can be determined by solving the* ***Modified Discrete Riccati Equations***

$$\mathcal{S}_n = \mathcal{Q}_n + \mathrm{I}_u^\top R_n \mathrm{I}_u + \mathcal{A}^\top \mathcal{S}_{n+1} \mathcal{A} - (\mathcal{A}^\top \mathcal{S}_{n+1} \mathcal{B} - \mathrm{I}_u^\top R_n)(R_n + \mathcal{B}^\top \mathcal{S}_{n+1} \mathcal{B})^{-1} (\mathcal{B}^\top \mathcal{S}_{n+1} \mathcal{A} - R_n \mathrm{I}_u) \tag{4.55a}$$

for $n \in \{1, \dots, N-1\}$ backwards in time with initial value

$$\mathcal{S}_N = \mathcal{Q}_N, \tag{4.55b}$$

where the matrices $\mathcal{Q}_n$, $n \in \{1, \dots, N\}$, are from (4.53c) *and $\mathcal{I}_n^*$ are the information vectors that incorporate the optimal control sequence $u^* = (u_n^*)_{n\in\{1,\dots,N-1\}}$.*

Proof. Basically, the proof works analogously to that of Theorem 4.2 with the difference that (4.53) is used instead of (4.1). Therefore, we decided to tag modified equations with the same reference number supplemented by *.
First, the use of information vectors instead of states in the formulation of the optimal control problem requires some adjustments. In particular, we need to consider

$$\mathcal{J}_N^\lambda(\mathcal{I}, u) := \frac{1}{2}\mathcal{J}_N(\mathcal{I}, u) + \lambda_1^\top \tilde{\mathcal{F}}(\mathcal{I}_1) + \sum_{n=1}^{N-1} \lambda_{n+1}^\top (\mathcal{F}(\mathcal{I}_n, u_n) - \mathcal{I}_{n+1}) =$$

$$= \frac{1}{2}\sum_{n=1}^{N} \mathcal{I}_n^\top \mathcal{Q}_n \mathcal{I}_n + \frac{1}{2}\sum_{n=1}^{N-1} (u_n - \mathrm{I}_u\mathcal{I}_n)^\top R_n (u_n - \mathrm{I}_u\mathcal{I}_n) + \lambda_1^\top \tilde{\mathcal{F}}(\mathcal{I}_1) + \sum_{n=1}^{N-1} \lambda_n^\top (\mathcal{F}(\mathcal{I}_n, u_n) - \mathcal{I}_{n+1})$$

with $\mathcal{F}(\hat{\mathcal{I}}, \hat{u}) := \mathcal{A}\hat{\mathcal{I}} + \mathcal{B}\hat{u}$ and $\tilde{\mathcal{F}}(\hat{\mathcal{I}}) := \bar{\mathcal{I}}_1 - \hat{\mathcal{I}}$, i.e., the system dynamics (4.53b) can be expressed through the equality constraints

$$\begin{aligned} \mathcal{F}(\mathcal{I}_n, u_n) - \mathcal{I}_{n+1} = 0, \quad n \in \{1, \dots, N-1\}, \\ \tilde{\mathcal{F}}(\mathcal{I}_1) = 0. \end{aligned}$$

Moreover, $u_{n-1} = \mathrm{I}_u\mathcal{I}_n$ for each $n \in \{1, \dots, N\}$ was used. Note that the sequence $\lambda = (\lambda_n)_{n\in\{1,\dots,N\}} \subset \mathbb{R}^4$ now consists of Lagrange multipliers of the same dimension as the information vectors $\mathcal{I}_n$, i.e., $l + m = 4$ instead of $l = 3$.
In the following equations, we analogously consider $\mathcal{J}_N^\lambda, \mathcal{J}_N, \mathcal{F}, \tilde{\mathcal{F}}$ instead of $J_N^\lambda, J_N, f, \tilde{f}$ and thus need to replace the states x_n by the corresponding information vectors $\mathcal{I}_n$ and the system matrices A, B, Q_n by their extensions $\mathcal{A}, \mathcal{B}, \mathcal{Q}_n$, which leads to

$$\frac{\partial \mathcal{J}_N^\lambda}{\partial \mathcal{I}_1}(\mathcal{I}, u) = (\mathcal{Q}_1 + \mathrm{I}_u^\top R_1 \mathrm{I}_u)\mathcal{I}_1 - \mathrm{I}_u^\top R_1 u_1 + \mathcal{A}^\top \lambda_2 - \lambda_1, \tag{4.5*}$$

$$\frac{\partial \mathcal{J}_N^\lambda}{\partial \mathcal{I}_n}(\mathcal{I}, u) = (\mathcal{Q}_n + \mathrm{I}_u^\top R_n \mathrm{I}_u)\mathcal{I}_n - \mathrm{I}_u^\top R_n u_n + \mathcal{A}^\top \lambda_{n+1} - \lambda_n, \quad n \in \{2, \dots, N-1\}, \tag{4.6*}$$

$$\frac{\partial \mathcal{J}_N^\lambda}{\partial \mathcal{I}_N}(\mathcal{I}, u) = \mathcal{Q}_N \mathcal{I}_N - \lambda_N, \tag{4.7*}$$

$$\frac{\partial \mathcal{J}_N^\lambda}{\partial u_n}(\mathcal{I}, u) = R_n(u_n - \mathrm{I}_u\mathcal{I}_n) + \mathcal{B}^\top \lambda_{n+1}, \quad n \in \{1, \dots, N-1\}. \tag{4.8*}$$

In particular, the reason for the specific choice of the parameters λ_n, which are now given by

$$\begin{aligned} \lambda_n &:= (\mathcal{Q}_n + \mathrm{I}_u^\top R_n \mathrm{I}_u)\mathcal{I}_n - \mathrm{I}_u^\top R_n u_n + \mathcal{A}^\top \lambda_{n+1}, \quad n \in \{1, \dots, N-1\}, \\ \lambda_N &:= \mathcal{Q}_N \mathcal{I}_N, \end{aligned} \tag{4.9*}$$

remains valid: Given any fixed control sequence u, the corresponding information vectors $\mathcal{I}_n$ are uniquely determined by the constraints (4.53b), i.e., it is reasonable to demand

$$\frac{\partial \mathcal{J}_N^\lambda}{\partial \mathcal{I}_n}(\mathcal{I}, u) = 0, \quad n \in \{1, \ldots, N\},$$

which exactly holds for λ_n from (4.9*). Note that the multipliers λ_n now necessarily depend on both $\mathcal{I}_n$ and u_n, since the controls u_{n-1}, which have already been chosen in the previous time step $n-1$ and thus are fixed at the time step n, need to be included in the partial derivatives with respect to $\mathcal{I}_n$, which was achieved by using $u_{n-1} = \mathrm{I}_u \mathcal{I}_n$ in $\mathcal{J}_N^\lambda(\mathcal{I}, u)$. Because (4.8*) holds instead of (4.8), consequentially, (4.12) needs to be replaced by

$$u_n^* = \mathrm{I}_u \mathcal{I}_n^* - R_n^{-1} \mathcal{B}^\top \lambda_{n+1}, \quad n \in \{1, \ldots, N-1\}. \tag{4.12*}$$

Note that

$$\frac{\partial^2 \mathcal{J}_N}{\partial u_n^2}(\mathcal{I}, u) = 2R_n$$

implies that the sufficient condition (4.13) for u^* being a minimum also applies to $\mathcal{J}_N^\lambda$. Using the dynamics (4.53b), the optimal information vectors are thus given by

$$\begin{aligned} \mathcal{I}_{n+1}^* &= \mathcal{A}\mathcal{I}_n^* + \mathcal{B}(\mathrm{I}_u \mathcal{I}_n^* - R_n^{-1} \mathcal{B}^\top \lambda_{n+1}), \quad n \in \{1, \ldots, N-1\}, \\ \mathcal{I}_1 &= \bar{\mathcal{I}}_1 := \begin{bmatrix} \bar{x}_1 \\ \bar{u}_0 \end{bmatrix}, \end{aligned} \tag{4.14*}$$

with

$$\begin{aligned} \lambda_n &:= (\mathcal{Q}_n + \mathrm{I}_u^\top R_n \mathrm{I}_u)\mathcal{I}_n - \mathrm{I}_u^\top R_n u_n + \mathcal{A}^\top \lambda_{n+1}, \quad n \in \{1, \ldots, N-1\}, \\ \lambda_N &:= \mathcal{Q}_N \mathcal{I}_N. \end{aligned} \tag{4.15*}$$

Assuming that

$$\lambda_n = \mathcal{S}_n \mathcal{I}_n^*, \quad n \in \{1, \ldots, N\} \tag{4.16*}$$

holds for some matrices $\mathcal{S}_n \in \mathbb{R}^{4\times4}$, $n \in \{1, \ldots, N\}$,

$$\lambda_{n+1} \overset{(4.16^*)}{=} \mathcal{S}_{n+1} \mathcal{I}_{n+1}^* \overset{(4.53b)}{=} \mathcal{S}_{n+1}(\mathcal{A}\mathcal{I}_n^* + \mathcal{B}u_n^*), \quad n \in \{1, \ldots, N-1\} \tag{4.17*}$$

together with (4.12*) implies

$$\begin{aligned} & u_n^* = \mathrm{I}_u \mathcal{I}_n^* - R_n^{-1} \mathcal{B}^\top \mathcal{S}_{n+1}(\mathcal{A}\mathcal{I}_n^* + \mathcal{B}u_n^*) \\ \Longleftrightarrow\ & u_n^* = (R_n + \mathcal{B}^\top \mathcal{S}_{n+1} \mathcal{B})^{-1}(R_n \mathrm{I}_u - \mathcal{B}^\top \mathcal{S}_{n+1} \mathcal{A})\mathcal{I}_n^*. \end{aligned} \tag{4.18*}$$

This yields the first part of the solution.
Hence, for all $n \in \{1, \dots, N-1\}$

$$\begin{aligned}
\mathcal{S}_n \mathcal{I}_n^* &\overset{(4.16^*)}{=} \lambda_n \overset{(4.15^*)}{=} (\mathcal{Q}_n + \mathrm{I}_u^\top R_n \mathrm{I}_u)\mathcal{I}_n^* - \mathrm{I}_u^\top R_n u_n^* + \mathcal{A}^\top \lambda_{n+1} \overset{(4.16^*)}{=} \\
&\overset{(4.16^*)}{=} (\mathcal{Q}_n + \mathrm{I}_u^\top R_n \mathrm{I}_u)\mathcal{I}_n^* - \mathrm{I}_u^\top R_n u_n^* + \mathcal{A}^\top \mathcal{S}_{n+1}\mathcal{I}_{n+1}^* \overset{(4.53b)}{=} \\
&\overset{(4.53b)}{=} (\mathcal{Q}_n + \mathrm{I}_u^\top R_n \mathrm{I}_u)\mathcal{I}_n^* - \mathrm{I}_u^\top R_n u_n^* + \mathcal{A}^\top \mathcal{S}_{n+1}(\mathcal{A}\mathcal{I}_n^* + \mathcal{B}u_n^*) = \\
&= (\mathcal{Q}_n + \mathrm{I}_u^\top R_n \mathrm{I}_u + \mathcal{A}^\top \mathcal{S}_{n+1}\mathcal{A})\mathcal{I}_n^* + (\mathcal{A}^\top S_{n+1}\mathcal{B} - \mathrm{I}_u^\top R_n)u_n^* \overset{(4.18^*)}{=} \\
&\overset{(4.18^*)}{=} (\mathcal{Q}_n + \mathrm{I}_u^\top R_n \mathrm{I}_u + \mathcal{A}^\top \mathcal{S}_{n+1}\mathcal{A})\mathcal{I}_n^* + \\
&+ (\mathcal{A}^\top S_{n+1}\mathcal{B} - \mathrm{I}_u^\top R_n)(R_n + \mathcal{B}^\top \mathcal{S}_{n+1}\mathcal{B})^{-1}(R_n \mathrm{I}_u - \mathcal{B}^\top \mathcal{S}_{n+1}\mathcal{A})\mathcal{I}_n^* \\
&\iff \Big(\mathcal{S}_n - \mathcal{Q}_n - \mathrm{I}_u^\top R_n \mathrm{I}_u - \mathcal{A}^\top \mathcal{S}_{n+1}\mathcal{A} - \\
&- (\mathcal{A}^\top \mathcal{S}_{n+1}\mathcal{B} - \mathrm{I}_u^\top R_n)(R_n + \mathcal{B}^\top \mathcal{S}_{n+1}\mathcal{B})^{-1}(R_n \mathrm{I}_u - \mathcal{B}^\top \mathcal{S}_{n+1}\mathcal{A})\Big)\mathcal{I}_n^* = 0
\end{aligned}$$

needs to hold for the optimal information vectors $\mathcal{I}_n^*$ resulting from the optimal control u^* and the corresponding optimal trajectory x^*, which is (according to Bellman's principle of optimality) satisfied if and only if

$$\mathcal{S}_n = \mathcal{Q}_n + \mathrm{I}_u^\top R_n \mathrm{I}_u + \mathcal{A}^\top \mathcal{S}_{n+1}\mathcal{A} - (\mathcal{A}^\top \mathcal{S}_{n+1}\mathcal{B} - \mathrm{I}_u^\top R_n)(R_n + \mathcal{B}^\top \mathcal{S}_{n+1}\mathcal{B})^{-1}(\mathcal{B}^\top \mathcal{S}_{n+1}\mathcal{A} - R_n \mathrm{I}_u) \tag{4.19*}$$

holds for all $n \in \{1, \dots, N-1\}$. Analogously, for $n = N$

$$\mathcal{S}_N \mathcal{I}_N^* \overset{(4.16^*)}{=} \lambda_N \overset{(4.15^*)}{=} \mathcal{Q}_N \mathcal{I}_N^*$$

holds if

$$\mathcal{S}_N = \mathcal{Q}_N. \tag{4.20*}$$

The symmetry of the matrices $\mathcal{S}_n$ again follows recursively from (4.19*) and (4.20*) as the matrices $\mathcal{Q}_n$ maintain the assumed symmetry of Q_n. □

As we have shown, the only innovation, namely the penalization of the changes in control instead of the controls themselves, does not prevent the linear feedback form of the optimal control sequence u^*. However, the optimal feedback gain matrices $\mathcal{K}_n$ now have to be multiplied by the information vectors $\mathcal{I}_n$ and not by the states x_n, which was to be expected since the additional information contained in the information vectors, the last selected control u_{n-1}, is now a decisive factor for the choice of u in terms of the resulting costs. Therefore, compared to the Discrete Riccati Equations (4.3), the Modified Discrete Riccati Equations (4.55) include additional terms consisting of R_n and I_u, which contain information about how u_{n-1}^* affects the subsequent optimal control u_n^* via the cost terms in the objective function.

With the linear feedback form (4.54) in terms of information vectors $\mathcal{I}_n$ it is thus possible to analogously compute the optimal feedback gain matrices $\mathcal{K}_n$ once a priori by solving (4.55) and then, starting from any initial state $\bar{x}_1$ with any reference control $\bar{u}_0$, alternately compute the current information vector $\mathcal{I}_n^*$ using (4.50), the respective optimal feedback u_n^* using (4.54), and the subsequent state x_{n+1} using (4.47b).[15]

It is particularly important for us to emphasize the impact of the choice of $\bar{u}_0$:
Since the difference between each two consecutive controls is penalized, the initial control $\bar{u}_0$ does not only influence the objective function values through an additional cost term (as the initial state $\bar{x}_1$ does). Much more, it directly affects the first optimal control u_1^* due to $u_1^* = -\mathcal{K}_1 \left[\bar{x}_1, \bar{u}_0\right]^\top$, and therefore it indirectly affects all following optimal controls u_n^*, $n \in \{2, \dots, N-1\}$, because u_1^* contained in $\mathcal{I}_2^*$ influences u_2^* by the same linear feedback form, and so on. In our application, however, a penalization of the difference between the first applied control u_1 and the reference control $\bar{u}_0$ should be included in the objective function, because otherwise it would be possible to apply any amount of force in the first time step without incurring costs. This, in turn, might lead to a solution, where most of the muscle activation required for the movement takes place in this first time step, which exactly contradicts our assumption of effort minimization, on the basis of which we have primarily decided to penalize changes in control at all.
Hence, the reference control $\bar{u}_0$ should be chosen as a "direct force" that could actually be applied at the beginning of the movement. A detailed discussion on which values might be considered useful is to be found in Section 6.1.

Finally, we would like to derive an explicit formula for the optimal value function, analogously to Theorem 4.4. For this purpose, we define the optimal value function for the partially cutoff linear-quadratic control problem with horizon $K \in \{1, \dots, N\}$ of (4.53) in terms of information vectors (analogously to Definition 4.3) as

$$\mathcal{V}_N^K(\bar{\mathcal{I}}_{N-K+1}) := \min_{u^{K-1} \subset \mathbb{R}} \mathcal{J}_N^K(\mathcal{I}^{K-1}, u^{K-1}), \quad \bar{\mathcal{I}}_{N-K+1} \in \mathbb{R}^4, \tag{4.56}$$

where

$$\mathcal{J}_N^K(\mathcal{I}^{K-1}, u^{K-1}) := \sum_{n=N-K+1}^{N} \mathcal{I}_n^\top \mathcal{Q}_n \mathcal{I}_n + \sum_{n=N-K+1}^{N-1} (u_n - u_{n-1})^\top R_n (u_n - u_{n-1}), \tag{4.57}$$

and where $\mathcal{I}^{K-1} := (\mathcal{I}_n)_{n \in \{N-K+1, \dots, N\}} \subset \mathbb{R}^4$ and $u^{K-1} := (u_n)_{n \in \{N-K+1, \dots, N-1\}} \subset \mathbb{R}$ satisfy

$$\begin{aligned} \mathcal{I}_{n+1} &= \mathcal{A}\mathcal{I}_n + \mathcal{B}u_n, \quad n \in \{N-K+1, \dots, N-1\}, \\ \mathcal{I}_{N-K+1} &= \bar{\mathcal{I}}_{N-K+1} := \begin{bmatrix} \bar{x}_{N-K+1} \\ \bar{u}_{N-K} \end{bmatrix}, \\ u_{N-K} &= \bar{u}_{N-K} \end{aligned} \tag{4.58}$$

[15]Equivalently, it is possible to alternately compute the information vectors using the dynamics (4.53b) and the respective optimal controls using (4.54).

for some initial values $\bar{x}_{N-K+1} \in \mathbb{R}^3$ and $\bar{u}_{N-K} \in \mathbb{R}$.
Bellman's principle of optimality then yields for each $K \in \{2, \dots, N\}$ and each $\bar{\mathcal{I}}_{N-K+1} \in \mathbb{R}^4$

$$\mathcal{V}_N^K(\bar{\mathcal{I}}_{N-K+1}) = \bar{\mathcal{I}}_{N-K+1}^\top \mathcal{Q}_{N-K+1} \bar{\mathcal{I}}_{N-K+1} + (\tilde{u}^* - \mathrm{I}_u \bar{\mathcal{I}}_{N-K+1})^\top R_{N-K+1} (\tilde{u}^* - \mathrm{I}_u \bar{\mathcal{I}}_{N-K+1}) + \\ + \mathcal{V}^{K-1}(\mathcal{I}_{N-K+2}^{\bar{\mathcal{I}}_{N-K+1},(\tilde{u}^*)}), \tag{4.59a}$$

where $\mathcal{V}_N^{K-1}(\mathcal{I}_{N-K+2}^{\bar{\mathcal{I}}_{N-K+1},(\tilde{u}^*)})$ denotes the minimum value of the partially cutoff optimal control problem with horizon $K-1$ starting in $\mathcal{I}_{N-K+2}^{\bar{\mathcal{I}}_{N-K+1},(\tilde{u}^*)} = \mathcal{A}\bar{\mathcal{I}}_{N-K+1} + \mathcal{B}\tilde{u}^*$ and $\tilde{u}^*$ is the first control of the optimal control sequence $u^{K-1,*} = (u_n^*)_{n \in \{N-K+1, \dots, N-1\}} \subset \mathbb{R}$ of the considered partially cutoff linear-quadratic control problem with horizon K, i.e.,

$$\tilde{u}^* = u_{N-K+1}^*. \tag{4.59b}$$

This implies the following statement:

Theorem 4.6. *For any initial information vector $\bar{\mathcal{I}}_1 \in \mathbb{R}^4$, the optimal value of the objective function $\mathcal{J}_N$ of the optimal control problem* (4.53) *is given by*

$$\mathcal{V}_N^N(\bar{\mathcal{I}}_1) = \mathcal{J}_N(\mathcal{I}^*, u^*) = \bar{\mathcal{I}}_1^\top \mathcal{S}_1 \bar{\mathcal{I}}_1, \tag{4.60}$$

with optimal control sequence $u^* = (u_n^*)_{n \in \{1, \dots, N-1\}} \subset \mathbb{R}$ from (4.54) *and $\mathcal{S}_1$ from the solution to the Modified Discrete Riccati Equations* (4.55).

Proof. The first equality follows directly from Theorem 4.5 and the definition of $\mathcal{V}_N^N$.
Analogously to Theorem 4.4, we prove the second equality (or rather $\mathcal{V}_N^N(\bar{\mathcal{I}}_1) = \bar{\mathcal{I}}_1^\top \mathcal{S}_1 \bar{\mathcal{I}}_1$) by induction with respect to $K \in \{1, \dots, N\}$.
As can be easily seen by definition,

$$\mathcal{V}_N^1(\bar{\mathcal{I}}_N) = \bar{\mathcal{I}}_N^\top \mathcal{Q}_N \bar{\mathcal{I}}_N = \bar{\mathcal{I}}_N^\top \mathcal{S}_N \bar{\mathcal{I}}_N$$

holds for any $\bar{\mathcal{I}}_N \in \mathbb{R}^4$.
We assume that for some arbitrary but fixed $K \in \{1, \dots, N-1\}$

$$\mathcal{V}_N^K(\bar{\mathcal{I}}_{N-K+1}) = \bar{\mathcal{I}}_{N-K+1}^\top \mathcal{S}_{N-K+1} \bar{\mathcal{I}}_{N-K+1} \tag{4.30*}$$

holds for any $\bar{\mathcal{I}}_{N-K+1} \in \mathbb{R}^4$ with $\mathcal{S}_{N-K+1}$ from the solution to the Modified Discrete Riccati Equations.

Then for $\tilde{u}^* \in \mathbb{R}$ first control of the optimal control sequence $u^{K,*} = (u_n^*)_{n \in \{N-K, \dots, N-1\}}$ of $\mathcal{J}_N^{K+1}(\mathcal{I}^K, u^K)$ with $\mathcal{I}_{N-K} = \bar{\mathcal{I}}_{N-K}$, i.e.,

$$\tilde{u}^* = -\mathcal{K}_{N-K} \bar{\mathcal{I}}_{N-K} \tag{4.31*}$$

the following holds for any $\bar{\mathcal{I}}_{N-K} \in \mathbb{R}^4$:

$$\begin{aligned}
\mathcal{V}_N^{K+1}(\bar{\mathcal{I}}_{N-K}) &\overset{(4.59)}{=} \bar{\mathcal{I}}_{N-K}^\top \mathcal{Q}_{N-K}\bar{\mathcal{I}}_{N-K} + (\tilde{u}^* - \mathrm{I}_u\bar{\mathcal{I}}_{N-K})^\top R_{N-K}(\tilde{u}^* - \mathrm{I}_u\bar{\mathcal{I}}_{N-K}) + \\
&\quad + \mathcal{V}_N^K\left(\mathcal{I}_{N-K+1}^{\bar{\mathcal{I}}_{N-K},(\tilde{u}^*)}\right) \overset{(4.30^*)}{=} \\
&\overset{(4.30^*)}{=} \bar{\mathcal{I}}_{N-K}^\top \mathcal{Q}_{N-K}\bar{\mathcal{I}}_{N-K} + (\tilde{u}^* - \mathrm{I}_u\bar{\mathcal{I}}_{N-K}))^\top R_{N-K}(\tilde{u}^* - \mathrm{I}_u\bar{\mathcal{I}}_{N-K}) + \\
&\quad + \left(\mathcal{I}_{N-K+1}^{\bar{\mathcal{I}}_{N-K},(\tilde{u}^*)}\right)^\top \mathcal{S}_{N-K+1}\left(\mathcal{I}_{N-K+1}^{\bar{\mathcal{I}}_{N-K},(\tilde{u}^*)}\right) \overset{(4.53b)}{=} \\
&\overset{(4.53b)}{=} \bar{\mathcal{I}}_{N-K}^\top \mathcal{Q}_{N-K}\bar{\mathcal{I}}_{N-K} + (\tilde{u}^* - \mathrm{I}_u\bar{\mathcal{I}}_{N-K}))^\top R_{N-K}(\tilde{u}^* - \mathrm{I}_u\bar{\mathcal{I}}_{N-K}) + \\
&\quad + (\bar{\mathcal{I}}_{N-K}^\top \mathcal{A}^\top + (\tilde{u}^*)^\top \mathcal{B}^\top)\mathcal{S}_{N-K+1}(\mathcal{A}\bar{\mathcal{I}}_{N-K} + \mathcal{B}\tilde{u}^*) \overset{(4.31^*)}{=} \\
&\overset{(4.31^*)}{=} \bar{\mathcal{I}}_{N-K}^\top\Big(\mathcal{Q}_{N-K} + \mathrm{I}_u^\top R_{N-K}\mathrm{I}_u + \mathcal{A}^\top\mathcal{S}_{N-K+1}\mathcal{A} + \mathcal{K}_{N-K}^\top(R_{N-K} + \mathcal{B}^\top\mathcal{S}_{N-K+1}\mathcal{B})\mathcal{K}_{N-K} - \\
&\quad -(\mathcal{A}^\top\mathcal{S}_{N-K+1}\mathcal{B} - \mathrm{I}_u^\top R_{N-K})\mathcal{K}_{N-K} - \mathcal{K}_{N-K}^\top(\mathcal{B}^\top\mathcal{S}_{N-K+1}\mathcal{A} - R_{N-K}\mathrm{I}_u^\top)\Big)\bar{\mathcal{I}}_{N-K} \overset{(4.54),(*)}{=} \\
&\overset{(4.54),(*)}{=} \bar{\mathcal{I}}_{N-K}^\top\Big(\mathcal{Q}_{N-K} + \mathrm{I}_u^\top R_{N-K}\mathrm{I}_u + \mathcal{A}^\top\mathcal{S}_{N-K+1}\mathcal{A} + \\
&\quad +(\mathcal{A}^\top\mathcal{S}_{N-K+1}\mathcal{B} - \mathrm{I}_u^\top R_{N-K})(R_{N-K} + \mathcal{B}^\top\mathcal{S}_{N-K+1}\mathcal{B})^{-1}(\mathcal{B}^\top\mathcal{S}_{N-K+1}\mathcal{A} - R_{N-K}\mathrm{I}_u) - \\
&\quad -2(\mathcal{A}^\top\mathcal{S}_{N-K+1}\mathcal{B} - \mathrm{I}_u^\top R_{N-K})(R_{N-K} + \mathcal{B}^\top\mathcal{S}_{N-K+1}\mathcal{B})^{-1}(\mathcal{B}^\top\mathcal{S}_{N-K+1}\mathcal{A} - R_{N-K}\mathrm{I}_u)\Big)\bar{\mathcal{I}}_{N-K} = \\
&= \bar{\mathcal{I}}_{N-K}^\top\Big(\mathcal{Q}_{N-K} + \mathrm{I}_u^\top R_{N-K}\mathrm{I}_u + \mathcal{A}^\top\mathcal{S}_{N-K+1}\mathcal{A} - \\
&\quad -(\mathcal{A}^\top\mathcal{S}_{N-K+1}\mathcal{B} - \mathrm{I}_u^\top R_{N-K})(R_{N-K} + \mathcal{B}^\top\mathcal{S}_{N-K+1}\mathcal{B})^{-1}(\mathcal{B}^\top\mathcal{S}_{N-K+1}\mathcal{A} - R_{N-K}\mathrm{I}_u)\Big)\bar{\mathcal{I}}_{N-K} \overset{(4.55)}{=} \\
&\overset{(4.55)}{=} \bar{\mathcal{I}}_{N-K}^\top\mathcal{S}_{N-K}\bar{\mathcal{I}}_{N-K}.
\end{aligned}$$

In $(*)$, the symmetry of $\mathcal{S}_{N-K+1}$ and the resulting symmetry of $(R_{N-K} + \mathcal{B}^\top\mathcal{S}_{N-K+1}\mathcal{B})$, which implies $(R_{N-K} + \mathcal{B}^\top\mathcal{S}_{N-K+1}\mathcal{B})^{-\top} = (R_{N-K} + \mathcal{B}^\top\mathcal{S}_{N-K+1}\mathcal{B})^{-1}$, were used.
Hence, the induction hypothesis (4.30*) is true for all $K \in \{1, \ldots, N\}$.
The last case $K = N$ thus yields

$$V_N^N(\bar{\mathcal{I}}_1) = \bar{\mathcal{I}}_1^\top\mathcal{S}_1\bar{\mathcal{I}}_1 \tag{4.32*}$$

for all $\bar{\mathcal{I}}_1 \in \mathbb{R}^4$, which proves the claim. □

4.3 A Few Generalizations

Although not relevant for the further proceeding, all presented optimal control problems can easily be adapted to 2D and 3D pointing tasks as well as to so-called "via-point tasks".
In the following, we briefly discuss the necessary adjustments to the terms used.

4.3.1 Extension to Higher Dimensions

The time-discrete linear control system with sampling time $h > 0$ (4.39) (and, equivalently, the time-continuous variants (4.36) and (4.34)[16]) can easily be extended to 2D or 3D pointing tasks by augmenting x_n and u_n with the respective components for the additional dimensions, i.e., defining

$$x_n := \begin{bmatrix} p_n^1 \\ \vdots \\ p_n^{\hat{d}} \\ v_n^1 \\ \vdots \\ v_n^{\hat{d}} \\ T^1 \\ \vdots \\ T^{\hat{d}} \end{bmatrix} \in \mathbb{R}^{3\hat{d}}, \quad n \in \{1, \dots, N\}, \quad u_n := \begin{bmatrix} u_n^1 \\ \vdots \\ u_n^{\hat{d}} \end{bmatrix} \in \mathbb{R}^{\hat{d}}, \quad n \in \{0, \dots, N-1\},$$

and extending each entry of the system dynamics matrices to block matrices by multiplying with the respective identity matrix, which leads to

$$A := \begin{bmatrix} \mathrm{ID}_{\hat{d}} & h\mathrm{ID}_{\hat{d}} & 0 \\ -hk\mathrm{ID}_{\hat{d}} & (1-hd)\mathrm{ID}_{\hat{d}} & 0 \\ 0 & 0 & \mathrm{ID}_{\hat{d}} \end{bmatrix} \in \mathbb{R}^{3\hat{d}\times 3\hat{d}}, \quad B := \begin{bmatrix} 0 \\ h\mathrm{ID}_{\hat{d}} \\ 0 \end{bmatrix} \in \mathbb{R}^{3\hat{d}\times \hat{d}},$$

where $\hat{d} \in \{2, 3\}$ denotes the dimension of the pointing task and, as in the entire section, $0 := 0_{\hat{d}} \in \mathbb{R}^{\hat{d}\times\hat{d}}$.

In addition, the 2OL-LQR model (4.47) requires cost matrices of matching dimension, i.e., $Q_n \in \mathbb{R}^{3\hat{d}\times 3\hat{d}}$, $n \in \{1, \dots, N\}$, and $R_n \in \mathbb{R}^{\hat{d}}$, $n \in \{1, \dots, N-1\}$. Here, each dimensional component can be penalized on its own, but a penalization of combinations across dimensions would also be conceivable. The easiest way, however, is to naturally extend the cost structure of the 1D case to higher dimensions. In the $\hat{d}$-dimensional case the state cost matrices (4.48) are then given by

$$Q_n := \begin{bmatrix} \mathrm{ID}_{\hat{d}} & 0 & -\mathrm{ID}_{\hat{d}} \\ 0 & 0 & 0 \\ -\mathrm{ID}_{\hat{d}} & 0 & \mathrm{ID}_{\hat{d}} \end{bmatrix} \in \mathbb{R}^{3\hat{d}\times 3\hat{d}}, \tag{4.48*}$$

and the control cost matrices (4.49) are given by

$$R_n := \frac{r_n}{h^2}\mathrm{ID}_{\hat{d}}, \quad r_n > 0. \tag{4.49*}$$

[16]In this case, of course, all target components that occur below, i.e., the last $\hat{d}$ entries in x_n, the last $\hat{d}$ rows and columns in A, and the last $\hat{d}$ rows in B, must be left out.

The corresponding information vectors are naturally given by

$$\mathcal{I}_n := \begin{bmatrix} x_n \\ u_{n-1} \end{bmatrix} \in \mathbb{R}^{4\hat{d}}, \quad n \in \{1, \ldots, N\}, \tag{4.50*}$$

and the extraction matrices are given by

$$\mathrm{I}_x := \begin{bmatrix} \mathrm{ID}_{\hat{d}} & 0 & 0 & 0 \\ 0 & \mathrm{ID}_{\hat{d}} & 0 & 0 \\ 0 & 0 & \mathrm{ID}_{\hat{d}} & 0 \end{bmatrix} \in \mathbb{R}^{3\hat{d} \times 4\hat{d}}, \quad \mathrm{I}_u := \begin{bmatrix} 0 & 0 & 0 & \mathrm{ID}_{\hat{d}} \end{bmatrix} \in \mathbb{R}^{\hat{d} \times 4\hat{d}}. \tag{4.51*}$$

Finally, the system matrices are extended analogously for use with the information vectors $\mathcal{I}_n$ instead of the states x_n, i.e.,

$$\mathcal{A} := \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \mathrm{ID}_{\hat{d}} & h\mathrm{ID}_{\hat{d}} & 0 & 0 \\ -hk\mathrm{ID}_{\hat{d}} & (1-hd)\mathrm{ID}_{\hat{d}} & 0 & 0 \\ 0 & 0 & \mathrm{ID}_{\hat{d}} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{4\hat{d} \times 4\hat{d}}, \quad \mathcal{B} := \begin{bmatrix} B \\ \mathrm{ID}_{\hat{d}} \end{bmatrix} = \begin{bmatrix} 0 \\ h\mathrm{ID}_{\hat{d}} \\ 0 \\ \mathrm{ID}_{\hat{d}} \end{bmatrix} \in \mathbb{R}^{4\hat{d} \times \hat{d}},$$

$$\mathcal{Q}_n := \begin{bmatrix} Q_n & 0 \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{4\hat{d} \times 4\hat{d}}, \quad n \in \{1, \ldots, N\}. \tag{4.53c*}$$

4.3.2 Extension to Via-Point Tasks

While this thesis mainly focuses on pointing tasks as defined in Section 4.2.1, it is also possible to apply our 2OL-LQR model to more generalized pointing tasks, which can be formalized as follows:
In a **via-point task**, users are asked to move the pointing device from its initial position to a specific target on which they need to click in order to indicate the end of their movement, with the additional condition that they need to pass and click on multiple intermediate targets, the so-called **via-points**, on their way there in a specific order.
A typical example, which was, e.g., used by Todorov to investigate the curvature of the individual partial trajectories [67], is the via-point task with several via-points arranged in two dimensions along a zigzag line. But also the 1D reciprocal pointing task on which the later used dataset is based (see Section 5.2 for a detailed description) could be considered as a single via-point tasks with initial and target position as alternating via-points instead of dividing it into multiple trials of two different pointing tasks, as we do.

There are many ways to extend our 2OL-LQR model, which was tailored to pointing tasks, to the via-point setting. However, for each of them it is crucial to know the so-called **via-point times**, i.e., the time steps at which the via-points are assumed to be passed through[17], in order to make a reasonable choice of cost matrices.

[17]In discrete time, we need to assume that this happens exactly at a sampled point in time.

In the following, we denote the target center positions by $(T_i)_{i\in\{1,\dots,P\}} \subset \mathbb{R}^l$, where $P \in \mathbb{N}$ is the number of via-points including the final target. Note that all these target center positions now must be included in the state vectors x_n to be able to penalize distances to them through the state cost matrices Q_n. Moreover, we denote the corresponding via-point times by $(k_i)_{i\in\{1,\dots,P\}} \subset \mathbb{N}$, $k_1 \leq \ldots \leq k_P := N$, and call the time spans between each two consecutive via-point times **via-point sections** (of course, including the time span between the beginning of the movement and the first via-point time). In particular, the case $P = 1$ exactly corresponds to a single pointing task with target center $T := T_1$.
While the final time step N can be interpreted as the maximum admissible time to reach the target, the remaining via-point times should, in theory, fit to the exact click times for the following reason: Since in our model convergence towards a target is achieved by state cost matrices Q_n that penalize the remaining distance to this target, at each time step information about which target to aim at and click on next is crucial to penalize the "correct" positional error. Unfortunately, an a priori definition of the state cost terms by cases depending on whether a particular target T_i will have been reached at time step n by the respective trajectory is not possible, as this would violate the assumption on quadratic costs required for any LQR-based model. The only possibility is therefore to specify the via-point times $(k_i)_{i\in\{1,\dots,P\}}$ in advance and define Q_n depending on which via-point time follows n first. This means that the via-point times are *not* the exact times at which a given trajectory passes the via-points, as would be desirable, but the times at which the model *suggests* to pass the via-points by penalizing appropriate positional errors during the movement. In particular, even the optimal trajectory does not necessarily need to reach the via-points at the via-point times (just as the optimal trajectory for pointing tasks does not have to be at the target center T at time step N in general, since no boundary conditions are implemented).
In our 2OL-LQR model, we could, for example, simply replicate the cost matrices up to the first via-point, $(Q_n)_{n\in\{1,\dots,k_1\}}$ respective $(R_n)_{n\in\{1,\dots,k_1\}}$, for the other via-point sections (where in Q_n the row and the column corresponding to the respective via-point must be used in each case).[18] Of course, it would also be possible to select these cost matrices basically different between individual via-point sections, whereby different objectives (or, at least, different weights of the objectives) are assumed for different via-point sections.
The required via-point times $(k_i)_{i\in\{1,\dots,P\}}$ could be chosen according to experimental observations or they are derived from a specific reference trajectory, e.g., from the minimum-jerk trajectory. However, as Todorov suggests for his extended LQG model [67], it might also be a good idea to additionally minimize the objective function with respect to the via-point times $(k_i)_{i\in\{1,\dots,P\}}$ in an outer loop, i.e., to find the combination of via-point times, such that the minimum among the optimal objective function values og all admissible combinations is assumed in the resulting optimal trajectory.

[18]If these matrices non-trivially depend on n and if the via-point times are not equidistantly distributed, a stretch or compression of the continuous extension of this sequence adjusted to the respective number of time steps of each via-point section might be reasonable.

Chapter 5

Parameter Fitting Process

As we have shown in the previous chapter, for any system matrices A, B, Q_n, R_n that meet the required properties together with any initial state $\bar{x}_1 \in \mathbb{R}^3$ (including the fixed target $T \in \mathbb{R}^3$) and any reference control $\bar{u}_0 \in \mathbb{R}$ there is a unique solution to the optimal control problem given by (4.47). However, trajectories[1] for the same pointing task vary considerably between different participants [50]. The main idea is thus to identify one or more parameters that affect the system matrices in our model for a specific participant or even for a specific trial. This *parameter fitting process* is described in Section 5.1, while the concrete trajectories used for this purpose are discussed in Section 5.2. Finally, our resulting complete algorithm is given in Section 5.3.

5.1 Least Squares Parameter Fitting

In the following, we consider *admissible parameter sets* of the form $\Lambda = (\lambda_1, \ldots, \lambda_s)$, i.e., sets that consist of $s \in \mathbb{N}$ parameters λ_i, each from a respective *admissible set* L_i.

Given an experimentally observed user trajectory y^{USER}, which can be formally described as the successive coupling of the position, velocity, and acceleration sequences $(p_n^{\text{USER}})_{n\in\{1,\ldots,N\}}$, $(v_n^{\text{USER}})_{n\in\{1,\ldots,N\}}$, and $(a_n^{\text{USER}})_{n\in\{1,\ldots,N\}}$, i.e., $y^{\text{USER}} = (y_n^{\text{USER}})_{n\in\{1,\ldots,3N\}} \subset \mathbb{R}^{\hat{d}}$ with

$$y_n^{\text{USER}} := \begin{cases} p_n^{\text{USER}}, & \text{if } 1 \le n \le N, \\ v_{n-N}^{\text{USER}}, & \text{if } N+1 \le n \le 2N, \\ a_{n-2N}^{\text{USER}}, & \text{if } 2N+1 \le n \le 3N, \end{cases} \tag{5.1}$$

and given a model that yields a solution trajectory $y^{\Lambda} = (y_n^{\Lambda})_{n\in\{1,\ldots,3N\}} \subset \mathbb{R}^{\hat{d}}$ of the same form for any admissible parameter set Λ, the **least squares parameter fitting (LSPF)**

[1]While so far we have used *trajectory* as a synonym for the state sequence $x = (x_n)_{n\in\{1,\ldots,N\}}$, regardless of what information these states contain, we now specifically mean the following dynamics properties of a motion: position, velocity and acceleration.

problem reads as follows:

$$\begin{aligned} &\textit{Minimize}\ \ \mathrm{SSE}(\Lambda) := \sum_{n=1}^{3N} (f_n(y_n^\Lambda) - \tilde{f}_n(y_n^{\mathrm{USER}}))^2 \\ &\textit{with respect to}\ \ \Lambda = (\lambda_1, \dots, \lambda_s),\ \lambda_i \in L_i, \end{aligned} \tag{5.2}$$

i.e., find the parameter set Λ^* that minimizes the **sum squared error (SSE)** between certain features of the simulation trajectory y^Λ and certain features of the observed user trajectory y^{USER}, which are given by the *feature functions* $f_n \colon \mathbb{R}^{\hat{d}} \longrightarrow \mathbb{R}$ respective $\tilde{f}_n \colon \mathbb{R}^{\hat{d}} \longrightarrow \mathbb{R}$ for all $n \in \{1, \dots, 3N\}$, among all admissible parameter sets Λ.
Note that in general it is not possible to analytically derive the solution to (5.2) (if one exists at all). For example, this is the case if y^Λ does not linearly depend on Λ but is given by a rather complex algorithm (as in our case described below). Instead, numerical methods that iteratively approach a (local) minimum starting from an *initial parameter set* Λ_0 must be applied then. For further details about least squares problems and data fitting processes consider, e.g., [3] or [6].

In our 2OL-LQR model given by (4.47) with dimension $\hat{d} = 1$, fixed target center $T \in \mathbb{R}$, sampling time $h > 0$, initial state $\bar{x}_1$, reference control $\bar{u}_0$, with state cost matrices Q_n that are for each $n \in \{1, \dots, N\}$ either defined by (4.48) or by setting $Q_n = 0$ (details on the latter two assumptions are given in Section 6.1), and with control cost matrices defined by (4.49), the possible parameters are $r_1, \dots, r_N, k, d$. Choosing some of them (denoted by $\lambda_1, \dots, \lambda_s$) to be optimized and providing fixed admissible values for the remaining parameters, 2OL-LQR yields, as shown above, a unique solution trajectory[2] for each admissible parameter set Λ, which we denote by y^Λ in the following.[3] While the concrete selection of the parameters to be optimized is dealt with in Section 6.1, this section discusses the choice of the objective function of the LSPF problem, i.e., the choice of the respective feature functions f_n and $\tilde{f}_n$.
At our first attempt, we decided to compare the trajectories at each time step in terms of position, velocity, and acceleration, i.e., we used $f_n(y_n^\Lambda) = y_n^\Lambda$ and $\tilde{f}_n(y_n^{\mathrm{USER}}) = y_n^{\mathrm{USER}}$ for each $n \in \{1, \dots, 3N\}$, which led to the sum squared error function

$$\mathrm{SSE}(\Lambda) := \sum_{n=1}^{N} \left[(p_n^\Lambda - p_n^{\mathrm{USER}})^2 + (v_n^\Lambda - v_n^{\mathrm{USER}})^2 + (a_n^\Lambda - a_n^{\mathrm{USER}})^2 \right]. \tag{5.3}$$

Note, that the velocity sequence v^Λ arises from forward differences applied to p^Λ due to the choice of the time-discrete system dynamics matrices from (4.39b) and for the acceleration

[2]Both p^Λ and v^Λ are included in the solution state vector x^Λ, and a^Λ can be derived from v^Λ using forward differences (except from a_N^Λ, which can be neglected).

[3]Note that the number of considered time steps N now corresponds to the click time step of the user trajectory to be compared. However, it would also be possible to include $\tilde{N} > N$ steps in the 2OL-LQR objective function and consider only the trajectory values of the first N time steps in the SSE function.

sequence a^Λ we proceed analogously. Thus, if forward differences applied to p^{USER} respective v^{USER} do not coincide with v^{USER} respective a^{USER} as well[4], it is not possible to reach zero as the minimum value of (5.3) anyway, i.e., the optimal trajectory y^{Λ^*} and the observed trajectory y^{USER} can definitely not match completely. In particular, if the optimal SSE value is small enough, gradient-based optimization algorithms will slow down a great deal as further reduction of positional errors leads to larger velocity and acceleration errors due to these numerical inconsistencies and vice versa, i.e., it is unclear how to further improve the parameter set Λ. Since already one of the three sequences of the solution trajectory clearly determines the other two (given initial position and velocity), it is thus straightforward to decide for exactly one LSPF criterion: Minimize either positional errors *or* velocity errors *or* acceleration errors. For reasons of simplicity, we chose the first option, i.e., we used $f_n(y_n^\Lambda) = y_n^\Lambda$ and $\tilde{f}_n(y_n^{\mathrm{USER}}) = y_n^{\mathrm{USER}}$ for all $n \in \{1, \ldots, N\}$ and $f_n(y_n^\Lambda) = \tilde{f}_n(y_n^{\mathrm{USER}}) = 0$ for all $n \in \{N+1, \ldots, 3N\}$, or, equivalently, implemented

$$\mathrm{SSE}(\Lambda) := \sum_{n=1}^{N} (p_n^\Lambda - p_n^{\mathrm{USER}})^2 \tag{5.4}$$

as objective function of the LSPF problem.

5.2 User Trajectories

In this section, we want to take a closer look on the specific user trajectories y^{USER} that we use for our LSPF problem. All used trajectories are based on the **Pointing Dynamics Dataset** by Müller, Oulasvirta, and Murray-Smith, which is described in detail in [50] and can be accessed at `http://joergmueller.info/controlpointing/`. The most important features are summarized below.

5.2.1 Pointing Dynamics Dataset

The Pointing Dynamics Dataset results from an experiment in which 12 participants (with expert level computer experience) were asked to perform an one-dimensional reciprocal pointing task that can be described as follows: Two rectangular target boxes are displayed on the screen, each with a width of W meters and a distance of D meters between their centers. The participants need to alternately move the mouse pointer from one target box to the other and click on it. Precisely, the participants were asked to "click on the target boxes as fast as possible while maintaining an error rate of below 5%". It was not possible to repeat failed trials, i.e., after a click outside the target box the participants were asked to move from their current position to the other target box. For a better overview, the current target

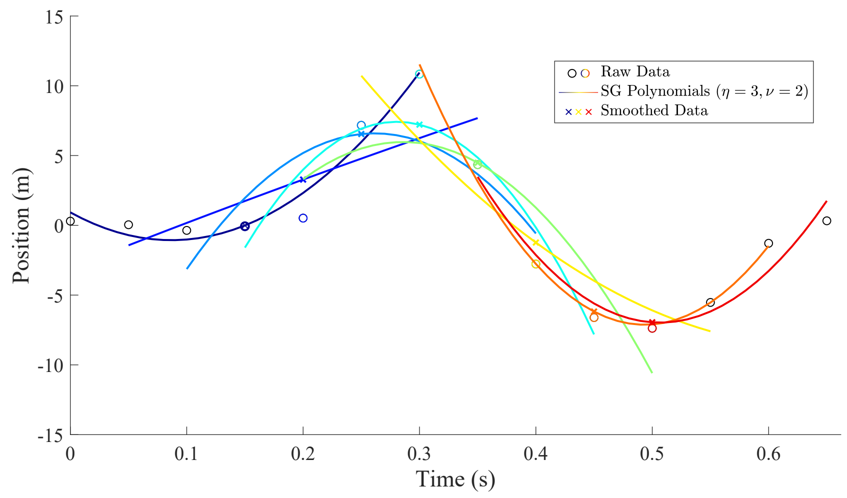
[4]As described in Section 5.2, it is indeed reasonable to use a Savitzky-Golay filter instead of forward differences when deriving the velocity and acceleration sequences of user trajectories.

ID	Distance D in px	Width W in px
2	1275	425
2	765	255
4	1275	85
4	765	51
6	1275	20
6	765	12
8	1275	5
8	765	3

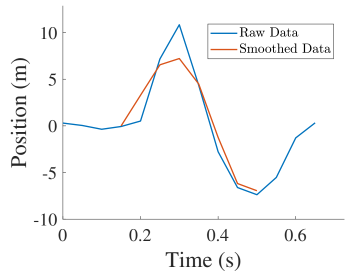
Table 5.1: *Overview of all conditions in the Pointing Dynamics Dataset.*

box was marked red, whereas the other one remained grey, respectively.
These individual alternating pointing tasks were performed one after the other without interruption; in particular, the color switch took place as soon as the participant clicked. After 102 such successive trials, the participants were allowed to rest and relax before the complete reciprocal pointing task was repeated with another **condition**, i.e., with different W and D. It should also be mentioned that there were training phases of 22 trials for each condition before the beginning of the experiment, i.e., the participants were already familiar with the concrete pointing tasks within the experiment.
In addition, this experiment only contained one-dimensional pointing tasks (see Section 4.2.1), i.e., for each condition both target boxes were placed at the same vertical position of the display, which means that only movements to the right or to the left were required. Hence, only the x-directional components of the measured data of the mouse device were used, i.e., movements in the y-direction had no effect.
The experiment included the same eight conditions for each participant, which had in pairs the same ratio of distance D to width W and thus the same ID, namely $2, 4, 6, 8$ using Shannon's formulation (3.5). An overview of all conditions (in pixels) is given in Table 5.1.

The raw position data was captured at a sampling rate of $h = 2$ms and then preprocessed, i.e., the first 20 trials were omitted from each condition block in order to avoid effects due to initially unaccustomed tasks. In addition, a *finite impulse response (FIR) filter* was used to eliminate high frequency disturbances from the input data and, above all, to ensure smooth time-discrete derivatives of the positional data, i.e., velocity and acceleration sequences without artificial oscillations caused by disturbed positional data (see the red lines in Figure 5.2). Müller used the **Savitzky-Golay (SG) filter** for this purpose, which can be summarized as follows (see [61] for details):
Let $p = (p_n)_{n\in\{1,\dots,N\}} \in \mathbb{R}$ denote the unprocessed position sequence.
Given a frame length $\eta \in \{1, \dots, \lfloor \frac{N}{2} \rfloor\}$ and a polynomial degree $\nu \in \{0, \dots, \eta - 1\}$, the SG filter smoothes for any $n \in \{1, \dots, N\}$ the partial sequence that consists of p_n and the nearest η values in both directions by choosing the ν-th degree polynomial that fits best.

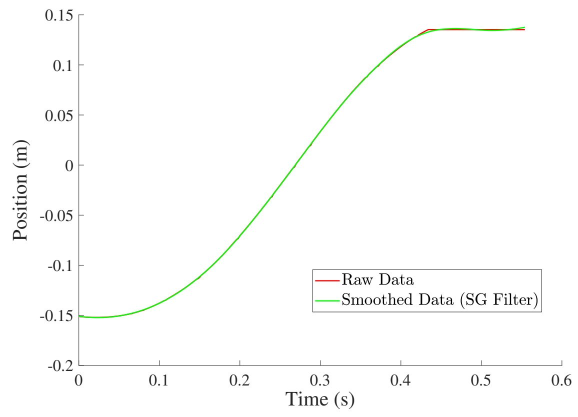


(a) *SG Polynomial Derivation*

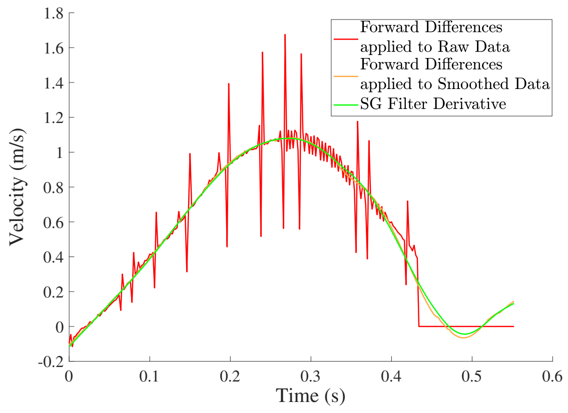


(b) *Position Time Series*

Figure 5.1: *Given a position time series p (**(a)** circles, **(b)** blue line), the SG filter with $\eta = 3$, $\nu = 2$ calculates for each discrete time $t \in \{0.15, 0.2, \ldots, 0.5\}$ the quadratic polynomial π_n (**(a)** any colored line) that best approximates the associated value (**(a)** same colored circle) and the three adjacent values in each direction with respect to (5.6). The resulting smoothed sequence $(\pi_n(0))$ is shown **(a)** by crosses of the respective color and **(b)** as red line.*



(a) *Position Time Series*



(b) *Velocity Time Series*

Figure 5.2: *Comparison of **(a)** the position time series with (green) and without (red) applied SG filter and **(b)** the velocity time series either derived by applying forward differences to the position time series with (orange) and without (red) applied SG filter or by using the derivatives of the individual polynomials from the SG filter (green). In each case, the same data from a representative trial of participant 1 (P1) for an ID 2 task was used.*

Formally, for each $n \in \{1, \ldots, N\}$ the coefficients $(c_i^n)_{i\in\{0,\ldots,\nu\}}$ of the ν-th degree polynomial

$$\pi_n(t) := \sum_{i=0}^{\nu} c_i^n t^i, \quad t \in \mathbb{R}, \tag{5.5}$$

are chosen so that $(\pi_n(j))_{j\in\{-\eta,\ldots,\eta\}}$ approximates $(p_{n+j})_{j\in\{-\eta,\ldots,\eta\}}$ best in terms of least squares, i.e.,

$$\sum_{j=-\eta}^{\eta} (\pi_n(j) - p_{n+j})^2 \tag{5.6}$$

is minimized.[5] In particular, the optimal coefficients (and thus the optimal polynomials) are uniquely determined as $\nu < \eta$ is assumed.
Repeating this procedure for all[6] $n \in \{1, \ldots, N\}$, where only the central value $\pi_n(0)$ is kept at each step, leads to the output sequence $(\pi_n(0))_{n\in\{1,\ldots,N\}}$ of the SG filter, which is a smoothed version of the input sequence $(p_n)_{n\in\{1,\ldots,N\}}$. The filtering process is illustrated in Figure 5.1 using a simple exemplary sequence and setting $\eta = 3$ and $\nu = 2$ for reasons of clarity (which, unfortunately, only leads to a slightly smoothed position time series as can be seen in Figure 5.1b), whereas Müller applied $\eta = 101$ and $\nu = 4$.
However, even though the computation of the time derivative of this smoothed sequence using numerical methods such as forward differences does not lead by far to such large oscillations as for the respective derivative of the unprocessed raw data (see Figure 5.2b), the resulting sequence is not arbitrarily smooth, which at least results in oscillatory behavior if the same method is used again for the computation of higher derivatives. Instead, it is advisable to directly use the polynomials π_n for each time step $n \in \{1, \ldots, N\}$ and to compute their derivatives up to the desired order $k \in \mathbb{N}$ at time $t = 0$, i.e., $\pi_n^{(j)}(0)$ for $j \in \{1, \ldots, k\}$, which, moreover, essentially consist of products of the coefficients c_i^n. The compound sequences $(\pi_n^{(j)}(0))_{n\in\{1,\ldots,N\}}$, $j \in \{1, \ldots, k\}$, then analogously yield smoothed versions of the respective derivative sequences, as depicted in Figure 5.2b with green color.

5.2.2 Further Processed Trajectories

Fur our purposes, we decided to further process the raw data, i.e., the measured positional data before the SG filter was applied. The concrete procedure included the removal of reaction times from the individual trials on the one hand and the computation of average trajectories on the other hand, and is described in detail below.
First of all, we want to motivate both modifications.

[5]Note that even though technical simplifications are the main reason for choosing $(t_j = j)_{j\in\{-\eta,\ldots,\eta\}}$ as the times at which π_n needs to approximate p_{n+j}, at least the underlying assumption of equidistant time steps is indispensable for this approach.

[6]Problems that arise for the first and the last η time steps, i.e., for $n \in \{1, \ldots, \eta\} \cup \{N - \eta + 1, \ldots, N\}$, are discussed at the end of Section 5.2.2.

Assumptions and Justifications

As discussed in Section 4.2.1, in reciprocal pointing tasks it is difficult to interpret time delays at the beginning of trials as *choice reaction time* [19, 53] because the participants are already familiar with the task and, in particular, with the following target position. However, there is a lot of variance in the duration of the time span between the click on the preceding target, i.e., the beginning of the new trial, and the point of time at which there is *substantial* acceleration first, as can be seen in Figure 5.3.[7] We call this time span **initial phase** of a trial in the following and formalize the above criterion by defining the first time step $n_\delta \in \{1, \dots, N\}$ that is excluded from this phase for a given accuracy barrier $\delta \in [0, 1[$ as follows:

$$n_\delta := \min\{i \in \{1, \dots, N\} \mid \frac{|\pi_i^{(2)}(0)|}{\max_{j\in\{1,\dots,N\}} |\pi_j^{(2)}(0)|} > \delta\}. \tag{5.7}$$

Assuming that the magnitude of acceleration is relatively small at the beginning of a trial and then rises monotonously to its maximum, the initial phase thus ends as soon as the ratio of the current absolute (smoothed) acceleration to the maximum absolute (smoothed) acceleration during this trial exceeds the barrier δ for the first time. In particular, this is reasonable for our use of the system dynamics (4.39) based on (2OL), since here the acceleration is proportional to the applied control.

Due to the large variance of n_δ among different trials and the difficulties of interpretation mentioned above, we decided to remove these initial phases from all trials, i.e., to omit all experimental data that might be related to reaction time phenomena.

In addition, we decided to separately average all left and all right movements that were performed consecutively within a condition block by a particular participant in order to obtain a *mean trajectory* for each condition, each participant, and each direction, which hopefully maintains the "characteristic" behavior of the respective participant for the respective task. The reason for this proceeding was that especially for computationally complex models with many parameters and/or many time steps, it might be a much less time-consuming first approach to concentrate only on the approximation of these few mean trajectories rather than on the approximation of all individual trial trajectories.
However, it is not clear how much both the averaging and the smoothing of the user trajectories affect their qualitative behavior, i.e., more general conclusions possibly can be drawn from the approximation of the raw data. Therefore, most of the results we present in this thesis relate to individual unfiltered trajectories, while mean trajectories are later dealt with in a separate section.

[7]In such a *box plot*, the horizontal red lines mark the medians, the bottom respective top edges of the blue boxes mark the 25th respective 75th percentiles, and the red plus signs represent *outliers* (see `https://de.mathworks.com/help/stats/boxplot.html#bu180jd-Whisker` for details). In our case, median and 25th percentile coincide due to the numerous trials with immediate substantial acceleration, i.e., without "reaction time".

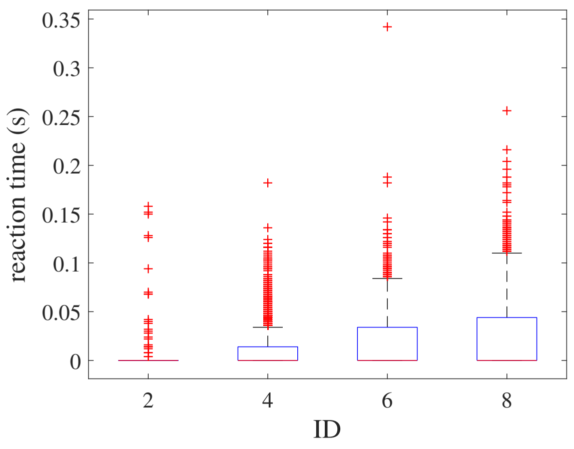


Figure 5.3: *The duration of the initial phase $(n_\delta - 1)h$, i.e., the "reaction time", varies considerably between different trials, especially for tasks with a higher ID (all trial trajectories described below were used).*

Concrete Procedure

The trajectories that contain one or both of the modifications described above were computed from the Pointing Dynamics Dataset as follows:
First, we subdivided the data of each task and each participant according to the individual trials and considered only the half that corresponds to the considered pointing task, i.e., either all right or all left movements. Additionally, we omitted all erroneous trials, i.e., any trial with missed target as well as the subsequent trial (since here the initial position was outside the initial box).[8] We then skipped the initial phases of each trial, that is the first $n_\delta - 1$ time steps, where we have chosen $\delta := 0.005$.[9]
The remaining *unprocessed* positional sequences of each trial, i.e., the sequences without applied SG filter, were then used as individual position time series $(p_n^{\text{USER}})_{n\in\{1,\ldots,N\}}$. For qualitative comparisons (but not for the optimization process itself), we additionally used the respective *smoothed*[10] velocity and acceleration sequences that were received by applying the SG filter with $\eta = 101$ and $\nu = 4$ to $(p_n^{\text{USER}})_{n\in\{1,\ldots,N\}}$, i.e.,

$$(v_n^{\text{USER}})_{n\in\{1,\ldots,N\}} := (\pi_n^{(1)}(0))_{n\in\{1,\ldots,N\}}, \quad (a_n^{\text{USER}})_{n\in\{1,\ldots,N\}} := (\pi_n^{(2)}(0))_{n\in\{1,\ldots,N\}}. \tag{5.8}$$

We denote the combination of these three sequences, which exactly correspond to the respective parts of the "y", "sgv", and "sga" columns in Müller's preprocessed dataset, as **trial trajectory** $(y_n^{\text{USER}})_{n\in\{1,\ldots,N\}}$ in the following.

[8] Although this criterion is very strict, only "completely failed" trials, i.e., trials with unintentional clicks at the beginning or in the middle of the movement, were sorted out.

[9] Note that the smoothed acceleration time series $(\pi_n^{(2)}(0))_{n\in\{1,\ldots,N\}}$ required for the computation of the time step n_δ for each trial were already included in the dataset.

[10] See Section 5.2.1 for a discussion on a reasonable computation of time-discrete derivatives.

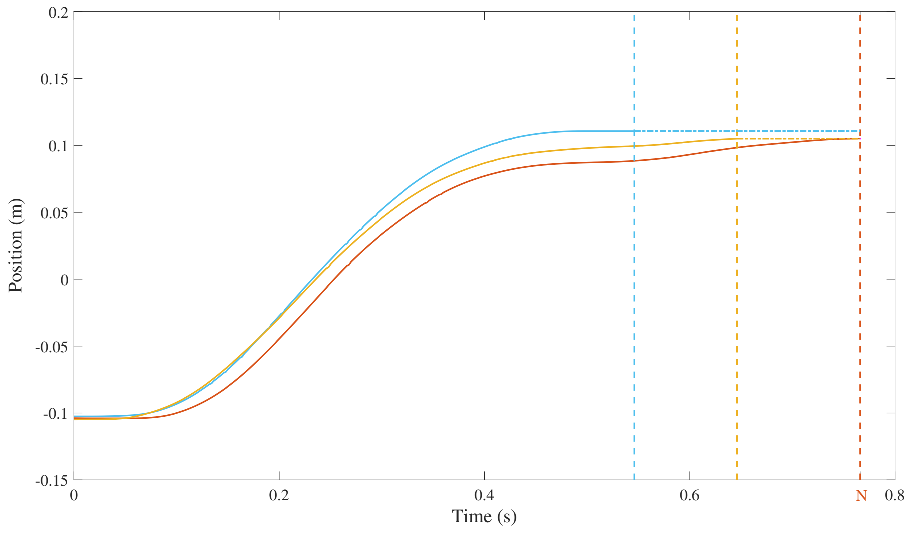


Figure 5.4: *For the computation of mean trajectories we need to bring all considered trajectories to the same length. In this example, both the blue and the yellow position sequence are thus extended by continuing their respective last value constantly up to the duration of the longest sequence (red), i.e., up to N.* `[P1, ID 4 (765,51), movements to the right]`

Furthermore, we extended these positional sequences to the maximum length of all considered trials by continuing all shorter trials constantly with their last position (as shown in Figure 5.4 exemplarily for three trajectories). The resulting sequences of the same length were averaged component by component. Finally, the same SG filter as above was applied to these averaged position sequences, leading to the final (smoothed) mean position sequence along with its time derivatives. For the sake of simplicity, we call such a trajectory **mean trajectory** in the following and use the existing notation, i.e., y^{USER} respectively p^{USER}, v^{USER}, and a^{USER}.

Transient Filtering

Last but not least, we would like to point out that whenever the SG filter is applied as described above difficulties may arise for $n \in \{1, \ldots, \eta\} \cup \{N - \eta + 1, \ldots, N\}$: Here, in at least one direction there exist less than η neighboring values of p_n that can be used for the fitting process with respect to (5.6). There are basically two ways to circumvent this problem: On the one hand, for such time steps n a variant of the SG filter which derives the polynomial that best fits only the available $\tilde{\eta} < 2\eta + 1$ neighboring input values could be applied. This was done for the preprocessed data of the Pointing Dynamics Dataset, for example.[11]

[11]However, this had virtually no effect on our considered trial trajectories, since here the successively executed movements for each condition and each participant were not split up before filtering, i.e., almost all

For the mean trajectories, where we applied the SG filter ourselves, we decided for another approach:
Since we only consider single unidirectional pointing tasks, whereas the Pointing Dynamics Dataset is based on reciprocal pointing tasks that contain successive trials of two alternating pointing tasks, in fact there *is* data available for both the starting and the terminal phase of our pointing task: We could either use end data from the previous trial or initial data from the following trial to fill the missing positions.[12] So what we actually did for the computation of our mean trajectories was to add η unprocessed positions both before and after each considered trial (after extending all trials to the same maximum length), average these sequences, and then apply the SG filter (with the above mentioned variant for the first and the last components). Afterwards, the previously appended components were removed again.
The advantage of this approach is what we call *consistent smoothing*: Each position (and thus each velocity and acceleration) is derived by solving a least squares problem with $2\eta+1$ neighboring values (in short, because the first and the last η values were artificially added and could be thus omitted at the end).

5.3 2OL-LQR Algorithm

We are now able to summarize our complete approach for the simulation of pointing task movements:
Given a trial or a mean trajectory y^{USER} of a specific pointing task, an initial state $\bar{x}_1$, a reference control $\bar{u}_0$, and a choice of parameters out of $\{r_1, \dots, r_N, k, d\}$ which are to be optimized, we want to minimize the SSE function

$$\text{SSE}(\Lambda) := \sum_{n=1}^{N} (p_n^{\Lambda} - p_n^{\text{USER}})^2 \tag{5.9}$$

with respect to all admissible parameter sets Λ, where p^{Λ} is from the solution to the respective 2OL-LQR problem (4.47) (or, equivalently, (4.53)), which is well defined only by the concrete choice of parameters Λ and therefore sometimes referred to as 2OL-LQR$^{\Lambda}$.[13]
In particular, k and d determine the system dynamics matrix A, whereas r_n defines the control cost matrix R_n for each $n \in \{1, \dots, N\}$. Solving the Modified Discrete Riccati Equations (4.55) then yields the optimal feedback gain matrices K_n, which in turn lead to the optimal control sequence u_n^* and the corresponding information vector sequence $\mathcal{I}^* = (\mathcal{I}_n^*)_{n\in\{1,\dots,N\}}$ including the desired position sequence $p^{\Lambda} = (p_n^{\Lambda})_{n\in\{1,\dots,N\}}$.

of our trajectories originate from the "middle" of the respective complete reciprocal pointing task anyway.

[12]We thus decided to only omit the first and the last trial for the computation of our mean trajectories to ensure that such data is always available.

[13]In accordance to our pointing dataset, which was described in detail in Section 5.2, we use a sampling time of $h = 2$ms.

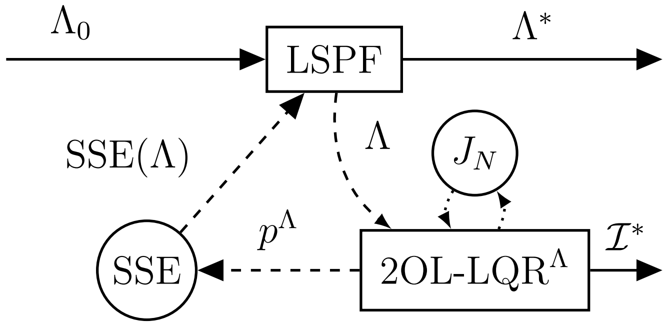


Figure 5.5: *Starting with an initial parameter set $\Lambda = \Lambda_0$, the least squares parameter fitting (LSPF) method obtains the sum squared error (SSE) value for the currently considered parameter set Λ. To do this, it calls 2OL-LQR$^\Lambda$, which sets up the respective optimal control problem (4.53) and obtains the solution information vector sequence $\mathcal{I}^*$. The resulting position time series p^Λ is used to compute* SSE(Λ)*, which is transmitted back to LSPF. As an LSPF algorithm, we use MATLAB's nonlinear least squares algorithm `lsqnonlin`, which uses a gradient-based search method to obtain the next set of parameters Λ until it convergences to an optimal parameter set Λ^* with (local) minimum SSE. Finally, Λ^* is returned along with the respective solution information vector sequence $\mathcal{I}^*$.*

In summary, in each step of the LSPF problem, which aims to find the parameter set Λ that best fits the observed trajectory y^{USER} with respect to (5.9), and which we solve by using MATLAB's nonlinear least squares algorithm `lsqnonlin`, an optimal control problem of the form (4.53) needs to be solved. The complete procedure is illustrated in Figure 5.5.

Finally, it should be noted that this least-squares-based algorithm may possibly only converge to a local minimum of the SSE function. Therefore, we executed the entire fitting process Υ times[14] with different initial parameter sets Λ_0 consisting of parameters $(\lambda_i)_{i\in\{1,\dots,s\}}$ that were randomly chosen from a continuous uniform distribution on the interval between 0 and the parameter-dependent upper bound ς_{λ_i}, respectively.[15]
This complete algorithm, which consists of optimal control problems of our 2OL-LQR model (with $\mathcal{J}_N$ as objective function) within an outer least squares parameter fitting process (with SSE as objective function) again within an outer global iteration loop, is called **2OL-LQR algorithm** and summarized in Algorithm 5.1.

[14]According to our experience, $\Upsilon = 100$ global iterations (in 2OL-LQR$_2$ even $\Upsilon = 50$) sufficed to provide results that would not improve considerably by iterating more.

[15]To speed up convergence, it proved useful to choose $\varsigma_r = 10^{-5}$ and $\varsigma_k = \varsigma_d = 100$.

```
   Input: target $T$, time steps $N$, sampling rate $h$, user trajectory $y^{\text{USER}}$
   Output: optimal parameter set $\Lambda^*$, optimal trajectory $x$, optimal control sequence
           $u^*$, approximation error $\text{SSE}(\Lambda^*)$
 1 compute $\mathcal{B}$, $(\mathcal{Q}_n)_{n\in\{1,\dots,N\}}$, and $\bar{x}_1$;
 2 choose $\Upsilon$ initial parameter sets $(\Lambda_0^{(i)})_{i\in\{1,\dots,\Upsilon\}}$;
 3 for $i = 1$ to $\Upsilon$ do
       // find local minimum $\Lambda^{*,(i)}$ of SSE($\Lambda$) starting from $\Lambda_0^{(i)}$:
 4     $\Lambda_{\text{new}} \leftarrow \Lambda_0^{(i)}$;
 5     while stopping criteria of lsqnonlin not fulfilled do
 6         $\Lambda \leftarrow \Lambda_{\text{new}}$;
 7         compute $\mathcal{A}$, $(R_n)_{n\in\{1,\dots,N\}}$, and $\bar{u}_0$;
 8         $\mathcal{I}_1^* \leftarrow [\bar{x}_1, \bar{u}_0]^T$;
 9         begin                                              // solve 2OL-LQR$^\Lambda$
10             obtain $(\mathcal{K}_n)_{n\in\{1,\dots,N-1\}}$ by solving the Modified Discrete Riccati Equations
                 backwards in time;
11             alternately compute $u_n^*$ and $\mathcal{I}_{n+1}^*$ for each $n \in \{1,\dots,N-1\}$;
12         end
13         extract $p^\Lambda$ from $(\mathcal{I}_n^*)_{n\in\{1,\dots,N\}}$ and compute $\text{SSE}(\Lambda)$;
14         find appropriate new parameter set $\Lambda_{\text{new}}$ ;                  // lsqnonlin
15     end
16     $\Lambda^{*,(i)} \leftarrow \Lambda$ ;                                   // store local minimum
17     $\mathcal{I}^{(i)} \leftarrow (\mathcal{I}_n^*)_{n\in\{1,\dots,N\}}$ ;           // store corresponding information vector
18 end
19 $i^* \leftarrow \text{argmin}_{i\in\{1,\dots,\Upsilon\}} \text{SSE}(\Lambda^{*,(i)})$ ;                     // find global minimum
20 $\Lambda^* \leftarrow \Lambda^{*,(i^*)}$;
21 extract $x$ and $u^*$ from $\mathcal{I}^{(i^*)}$;
```

Algorithm 5.1: 2OL-LQR algorithm.

Chapter 6

Design of the Cost Function and Results

In this chapter, we want to motivate the still outstanding concrete objective function J_N of our 2OL-LQR model given by (4.47) (and thus the actually implemented objective function $\mathcal{J}_N$ for the equivalent optimal control problem in information vector form (4.53)). For this purpose, in Section 6.1 we first show the disadvantages of an obvious approach for the selection of the cost matrices Q_n and R_n, which we refer to as 2OL-LQR$_1$. Subsequently, an improved variant called 2OL-LQR$_2$ is presented and, in Section 6.2, evaluated and compared with two previously considered models, 2OL-Eq and MinJerk.

6.1 Variants of 2OL-LQR

6.1.1 2OL-LQR$_1$

Based on our made assumptions about users' task understanding and the resulting internalized objectives, we defined in Section 4.2.3 appropriate state cost matrices Q_n by (4.48), which quadratically penalize the remaining distance to the target center at time step n, i.e.,

$$x_n^\top Q_n x_n = (T - p_n)^2 \tag{6.1}$$

applies to these matrices Q_n.
In addition, we motivated the use of the control cost matrices R_n from (4.49) with weights $r_n > 0$ that penalize the jerk $j_n := \frac{(u_n - u_{n-1})}{h}$ at time step n, i.e.,

$$(u_n - u_{n-1})^\top R_n (u_n - u_{n-1}) = r_n j_n^2 \tag{6.2}$$

applies to these matrices R_n.

However, we have not yet determined the time steps $n \in \{1, \ldots, N\}$ and $n \in \{1, \ldots, N-1\}$, respectively, at which these costs should arise.

In our first attempt, we decided to penalize the distance between mouse position and target center only at the last considered time step N, i.e., we chose

$$Q_n := \begin{cases} 0, & \text{if } 1 \leq n < N, \\ \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}, & \text{if } n = N, \end{cases} \tag{6.3}$$

which leads to

$$\sum_{n=1}^{N} x_n^T Q_n x_n = x_N^T Q_N x_N = (T - p_N)^2. \tag{6.4}$$

Note that N is now crucial for the task interpretation: Since the mouse position is cost-neutral at any other time step $n \in \{1, \dots, N-1\}$, i.e., the model only creates incentives to be close to the target at time step N, the choice of N has a substantial influence on the duration of the actual movement, i.e., on the time step from which the trajectory remains at the target center.[1] In particular, with this cost structure it is not possible to select N rather large and interpret it as an upper limit for the duration of the actual movement which is then determined by the parameter fitting process. Instead, it is necessary to equate N in advance with the duration of the user trajectory we want to approximate in order to obtain reasonable cost matrices Q_n from (6.3).[2] However, the reason for using this approach as a starting point of our iterative cost structure design process was that these state cost matrices can be seen as a simplification of the proposed cost structure in Todorov's application of the LQG to via-point tasks [67].

While positional error costs are only incurred at time step N, it is straightforward to permanently penalize jerk with the same weight $r_n = r > 0$ for all $n \in \{1, \dots, N-1\}$ since the magnitude of r_n expresses the participant's preference for moving smoothly towards the target, which is assumed to be constant throughout the whole trial (except from reaction time effects, see Chapter 7). In particular, this is in line with our principle of only using a few parameters that are, however, easy to interpret.

The resulting variant of the 2OL-LQR optimal control problem (4.47), which is denoted by **2OL-LQR**$_1$, thus has

$$J_N(x, u) = (T - p_N)^2 + r \sum_{n=1}^{N-1} j_n^2 \tag{6.5}$$

[1]Although the existence of such a time step cannot be proven in general, at time step N the trajectories had usually passed the target at least once (see Figure 6.1).

[2]Even though we proceed in the same way for the further variants of our model, it is only with this cost structure that a reasonable N cannot be chosen arbitrarily large but extremely depends on the considered user trajectory.

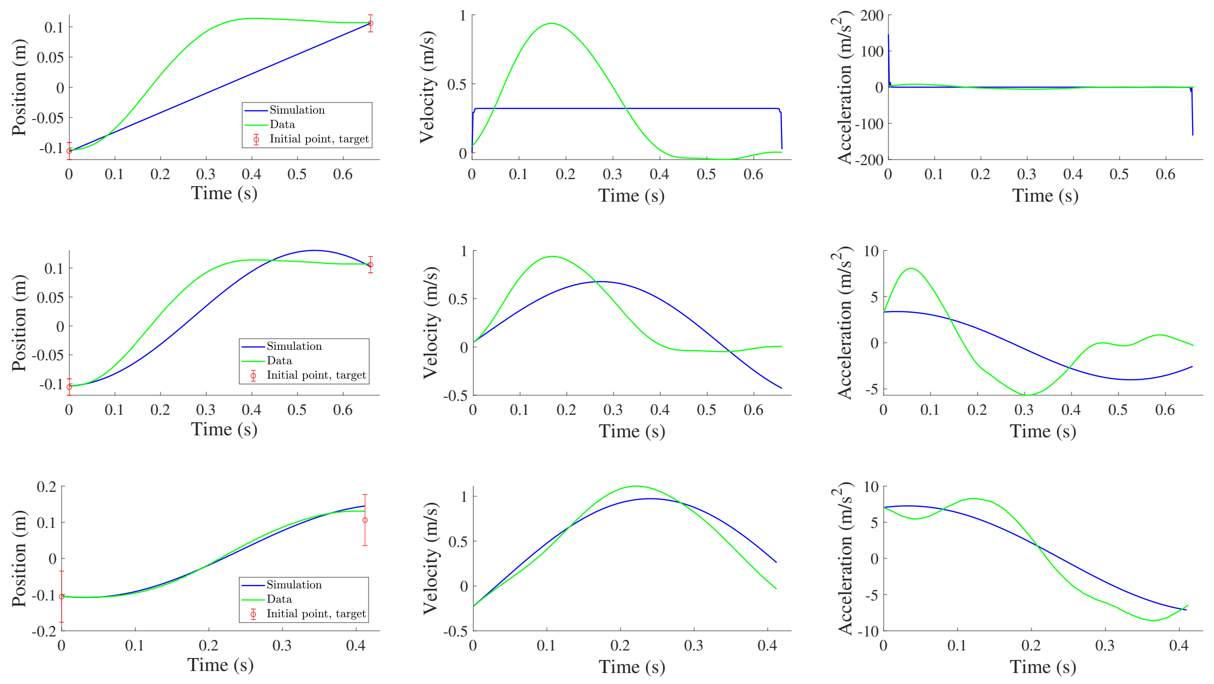
as objective function.
Because we expected the jerk weight r to be user- or even trial-dependent, we included it together with the 2OL-parameters k and d in the least squares parameter fitting process, i.e., $\Lambda = (r, k, d) \in \mathbb{R}^3$ with $r, k, d > 0$.
In total, we used the 2OL-LQR Algorithm described in Section 5.3, where in each step of the least squares parameter fitting process p^{Λ} is extracted from the solution to 2OL-LQR$_1^{\Lambda}$ for a given choice of parameters Λ.

Once again, we would like to explicitly point out the dependency of the resulting optimal simulation trajectory on the choice of the initial "motor template" $\bar{u}_0$ due to its effect on the optimal choice of the first motion-relevant control u_1. In particular, penalizing positional error only at a single time step while penalizing jerk, i.e., the differences between consecutive controls u_n, during the whole movement further enhances this dependency since $u_0 = \bar{u}_0$ strongly affects u_1, which in turn has great effects on u_2, and so on.
With regard to the assumed jerk minimization, there are two reasonable ways to define $\bar{u}_0$ and $\bar{x}_1$ from our point of view: task-dependent and trajectory-dependent.
In the **task-dependent** variant $\bar{x}_1 := [T_0, 0, T]^{\top}$ applies, i.e., only trajectories that start at the initial position $T_0 \in \mathbb{R}$ given by the task instruction[3] with zero velocity are allowed. In this case, we used $\bar{u}_0 := kT_0$ for the following reason: Here, choosing u_1 cost-minimal with regard to *pure* jerk penalization, i.e., $u_1 = u_0 = \bar{u}_0$, would additionally lead to zero acceleration due to the used system dynamics (4.39) and therefore ensure that the mouse cursor remains at the initial position, which appears to be useful for an initial reference control.[4]
However, the considered user trajectories are subtrajectories from the Pointing Dynamics Dataset, which is based on reciprocal pointing tasks, i.e., none of these two "equilibrium-start" properties has to apply at all to the user trajectories we want to approximate.
Hence, one might argue that by focusing on the parameter fitting process rather than on hypothetical assumptions on the pointing task it is straightforward to adapt the initial values $\bar{x}_1$ and $\bar{u}_0$ to certain properties of the considered user trajectory in order to ensure optimal fitting conditions. In the **trajectory-dependent** variant we thus set $\bar{x}_1 := \left[p_1^{\text{USER}}, v_1^{\text{USER}}, T\right]^{\top}$ and $\bar{u}_0 := kp_1^{\text{USER}} + dv_1^{\text{USER}} + a_1^{\text{USER}}$ so that, analogously, for $u_1 = u_0 = \bar{u}_0$ the initial acceleration of the user trajectory a_1^{USER} would now be assumed by the simulation trajectory, since

$$x_2 \overset{(4.39)}{=} Ax_1 + Bu_1 = A\bar{x}_1 + B\bar{u}_0 \tag{6.6}$$

[3]In our case, T_0 thus corresponds to the target center of the immediately preceding movement.

[4]The fact that in our model the application of a nontrivial control $\bar{u}_0 \neq 0$ is required to remain at an initial position $T_0 \neq 0$ results from $\bar{x}_1 := [T_0, 0, T]^{\top}$ not being an equilibrium of the uncontrolled system $\dot{x} = \tilde{A}x$ respective $x_{n+1} = Ax_n$ in our case, i.e., $\tilde{A}\bar{x}_1 \neq 0$ and $A\bar{x}_1 \neq \bar{x}_1$ holds, respectively. Using position coordinates relative to T_0 would circumvent this problem, but then some control $u_n \neq 0$ would still be required to remain at the target $T \neq T_0$. Furthermore, in our experience, such coordinate shifts had no substantial effect on the solution (apart from shifted optimal control sequences, of course).



(a) *Position Time Series* **(b)** *Velocity Time Series* **(c)** *Acceleration Time Series*

Figure 6.1: *2OL-LQR$_1$ model: Using a cost function similar to the one proposed by Todorov results in simulation trajectories (blue) not replicating the user data (green) well.*
[Top Row: `P1, ID 4 (765,51), 21th movement to the right, task-dependent;`
Center Row: `P1, ID 4 (765,51), 21th movement to the right, traj.-dependent;`
Bottom Row: `P1, ID 2 (765,255), 30th movement to the right, traj.-depend.]`

implies

$$\begin{aligned} v_2 &= (-hk)p_1^{\text{USER}} + (1-hd)v_1^{\text{USER}} + h\bar{u}_0 = \\ &= (-hk)p_1^{\text{USER}} + (1-hd)v_1^{\text{USER}} + h(kp_1^{\text{USER}} + dv_1^{\text{USER}} + a_1^{\text{USER}}) = v_1 + ha_1^{\text{USER}} \end{aligned} \tag{6.7}$$

and thus, using the forward Euler method to derive the initial acceleration a_1,

$$a_1 := \frac{v_2 - v_1}{h} = a_1^{\text{USER}} \tag{6.8}$$

applies.

Results

The results of the application of both the task-dependent and the trial-dependent 2OL-LQR$_1$ variant of our 2OL-LQR algorithm to representative trial trajectories are plotted in Figure 6.1, where the initial box[5] and the target box are each represented by an error bar with

[5]Since the user trajectories originate from a reciprocal pointing task, they do not necessarily start from the initial point itself but from some point within the preceding target box, that is, the initial box.

length equal to the width of the box.

The most striking property of the *task-dependent trajectory* shown in the top row of Figure 6.1 is the nearly linear position time series (left plot). While the underlying ID 4 pointing task is obviously already too difficult for the participants to move straight towards the target and click immediately (green line), the simulation trajectory shows exactly this behavior and reaches the target precisely after N time steps (blue line). Similar results were observed for most trial and mean trajectories of each participant in the task-dependent 2OL-LQR$_1$ variant of our 2OL-LQR algorithm and also for several trajectories in the target-dependent variant.
The main reason for this behavior is the absence of obligation to reach the target as early as possible due to the cost structure of 2OL-LQR$_1$: Since $Q_n = 0$ holds for all time steps n up to $N-1$, deviations from the target center are not penalized throughout the whole movement. Hence, there is clearly no need to reach the target earlier than necessary, i.e., before the time step N that corresponds to the click time of the user trajectory we want to approximate. In other words, the cost structure of 2OL-LQR$_1$, which was not only quite intuitive but also inspired by Todorov's approach [67], neglects a fundamental aspect of users' understanding of pointing tasks: the time minimization, or more precisely, the minimization of the duration of the surge phase.
However, this only explains why faster movements are not forced, but not why they obviously do not occur at all. In fact, it is the quadratic jerk penalization term in (6.5) which ensures that (even for small r) many small changes between consecutive controls are cheaper than one very powerful change with which the target is reached much earlier. This allows for highly unconventional movements: Choosing the spring constant k and the damping d very large and the jerk weight r very small leads to an optimal trajectory with respect to (6.5) that exhibits short acceleration impulses both at the beginning and at the end of the movement and thus has a nearly constant positive velocity time series and a nearly linear position time series, which reaches the target center exactly at time step N (see top row of Figure 6.1). Apparently, such an unrealistic choice of parameters[6] minimizes the SSE under the given conditions.

The *trajectory-dependent variant*, however, leads to qualitatively different optimal simulation trajectories if the initial acceleration of the user trajectory (which affects $\bar{u}_0$ in this variant) is sufficiently different from zero: As depicted in the center row of Figure 6.1 for the same trial trajectory as above, the resulting simulation trajectory (blue line) is now much smoother than in the task-dependent variant and does not include any physically implausible initial impulses. However, for the same reason explained above, this variant is also in such cases not able to reproduce the typically long correction phase of users, in which the target is

[6]The spring-mass-damper system is only for reasonable small values of the spring constant k and damping factor d a meaningful description of the biomechanical apparatus. However, we decided not to implement any upper bounds for the parameters since their choice would always be arbitrary.

already approximated quite well, but refinements are necessary to finally steer the mouse pointer into the relatively small target box.

While the above phenomena were observed for different user trajectories of all considered pointing tasks with ID between 4 and 8, user trajectories of ID 2 tasks were nevertheless approximated surprisingly well for both variants, as depicted exemplarily in the bottom row of Figure 6.1. The reason for that is simply the lack of correction phases in user trajectories of very simple pointing tasks. However, we cannot be satisfied with sufficiently good approximations of under-determined tasks, while the behavior typically observed in more difficult tasks is not replicated.
We therefore present an improved variant of our 2OL-LQR model in the following.

6.1.2 2OL-LQR$_2$

One major problem of our previous modeling approach is that we *need* to use the click time step of the specific user trajectory we wish to approximate as the final time step N, because otherwise there is no chance that both trajectories reach the target at the same time. This click time cannot be justified from the task description itself (the only time-related instruction is to complete the task "as soon as possible") and, above all, heavily depends on the concrete trial. While one way to bypass this problem would be an optimization of the parameter N (similar to some of Todorov's proposals [67]), which would involve the solution of an *integer* least squares problem and thus further complicate the parameter fitting process, we instead decided for an expansion of the positional-error-relevant time span, which considerably reduces the impact of N.

In concrete terms, we defined

$$Q_n := \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}, \quad n \in \{1, \dots, N\}, \tag{6.9}$$

which immediately implies

$$\sum_{n=1}^{N} x_n^T Q_n x_n = \sum_{n=1}^{N} (T - p_n)^2, \tag{6.10}$$

i.e., the distance between mouse position and target center is now penalized equally at every time step.[7]

[7]Note that the first state cost term $x_1^\top Q_1 x_1$ cannot be affected since $x_1 = \bar{x}_1$ needs to hold. Setting, e.g., $Q_1 = 0$ or, consistent with the approach for via-point tasks, analogously penalizing the distance between the required initial position T_0 and the actually implemented initial position p_1 by the first state cost matrix Q_1 (which would also require T_0 to be included in the states x_n) would thus only change the minimum value of J_N but not the optimal control sequence u^*.

Using the same jerk cost terms as in 2OL-LQR$_1$, the objective function of this variant, which we call **2OL-LQR**$_2$, is given by

$$J_N(x,u) = \sum_{n=1}^{N}(T-p_n)^2 + r\sum_{n=1}^{N-1} j_n^2. \tag{6.11}$$

In contrast to the 2OL-LQR$_1$ objective function (6.5), the additional penalization of the distance between the respective mouse position and the target center at each time step $n \in \{1,\dots,N-1\}$ implies higher values of J_N, even for the optimal control u^*, but, as can be seen below, leads to smaller optimal SSE values at the same time, i.e., to solution trajectories of our 2OL-LQR algorithm that better adapt user behavior.

Due to the relatively small impact of the initial values $\bar{x}_1$ and $\bar{u}_0$, which we observed in this variant, from now on we concentrate only on the *trajectory-dependent variant* without further reasoning.

6.2 Main Results

In this section, we evaluate our model 2OL-LQR$_2$ by comparing it with the two pure models whose approaches it combines – the minimum-jerk model from [23] and the second-order lag with equilibrium control from [50], which were introduced in detail in Section 3.3.1 and Section 3.2.2, and which we further refer to as MinJerk and 2OL-Eq, respectively. We also investigate how the parameters of our model change with different tasks, i.e., different IDs, and with different participants. In addition, we compare the results obtained from the application to trial trajectories to those obtained from the application to mean trajectories.

For the sake of clarity, we present our results using particularly illustrative trajectories. However, if not explicitly mentioned, we did not find any major qualitative differences to the remaining trials of the considered tasks and participants.

6.2.1 Qualitative Comparisons

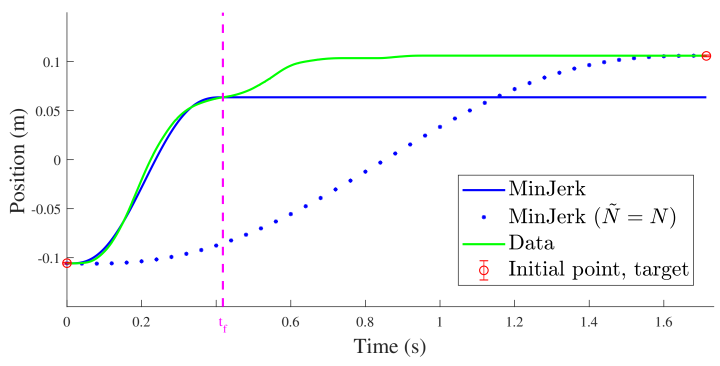
MinJerk

The MinJerk model yields a position time series $(p_n^{\text{MinJerk}})_{n\in\{1,\dots,\tilde{N}\}} = (p((n-1)h))_{n\in\{1,\dots,\tilde{N}\}}$ with $p:[0,t_f] \longrightarrow \mathbb{R}$ minimizing the total squared jerk

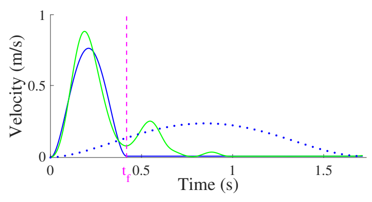
$$J(p) := \frac{1}{2}\int_0^{t_f} (\dddot{p}(t))^2 \, \mathrm{d}t \tag{6.12}$$

with initial condition

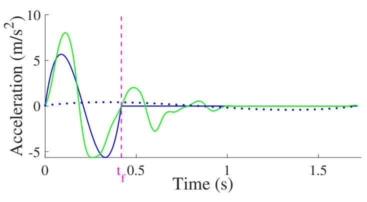
$$\begin{bmatrix} p(0) \\ \dot{p}(0) \\ \ddot{p}(0) \end{bmatrix} = \begin{bmatrix} \bar{p}_0 \\ \bar{v}_0 \\ \bar{a}_0 \end{bmatrix} \in \mathbb{R}^3 \tag{6.13}$$



(a) *Position Time Series*

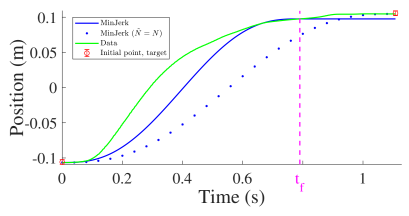


(b) *Velocity Time Series*

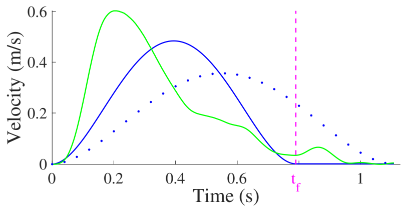


(c) *Acceleration Time Series*

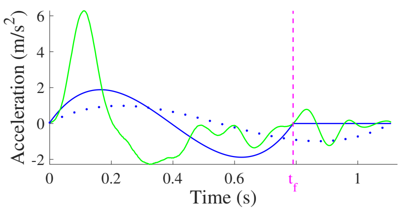
Figure 6.2: *For the MinJerk model, we have to decide whether we want to model the surge well, but not reach the target (blue solid line with constant continuation after the surge's end at time t_f), or reach the target, but not model the entire movement well (blue dotted line). In this thesis, we decided for the former option. The better fit of this variant to the user trajectory (green line) is quantified by the SSE: While MinJerk applied to the entire movement, i.e., $\tilde{N} = N$, exhibits an SSE value of 7.2333, the surge phase variant has an SSE value of 0.993, which is mainly caused by the missed target at time t_f (the respective SSE value of the surge phase only, i.e., up to t_f, is 0.0044).* `[Participant 1 (P1), ID 8 (distance: 765 px, width: 3 px), 21th movement to the right]`



(a) *Position Time Series*



(b) *Velocity Time Series*



(c) *Acceleration Time Series*

Figure 6.3: *Any method for the identification of the surge phase that is based on some characteristics of the user trajectory (such as our approach, where t_f denotes the second zero-crossing of the acceleration time series) leads to worse approximations of atypical motions (green) using the MinJerk model.* `[P7, ID 6 (765,12), 30th movement to the right]`

and terminal condition

$$\begin{bmatrix} p(t_f) \\ \dot{p}(t_f) \\ \ddot{p}(t_f) \end{bmatrix} = \begin{bmatrix} \bar{p}_{t_f} \\ \bar{v}_{t_f} \\ \bar{a}_{t_f} \end{bmatrix} \in \mathbb{R}^3, \quad t_f = t_{\tilde{N}} = (\tilde{N} - 1)h, \tag{6.14}$$

which is given by (3.31). However, originally it was derived from data of an experiment that did not involve any corrective movements [23]. This leaves two possibilities to fit the model to our data:
If MinJerk is used for modeling the entire movement, i.e., we set $\tilde{N} = N$, the fit is very poor (see Figure 6.2; dotted line). Instead of a quick movement towards the target with extensive corrective movements, as in our data, the model predicts a slow, smooth movement, reaching the target only at the time of the mouse click.[8]
Therefore, we use MinJerk for only the first, rapid movement towards the target, i.e., the *surge phase*. We assume that the pointer does not move afterwards, which is implemented by setting $\bar{v}_{t_f} = \bar{a}_{t_f} = 0$ in (6.14). Similar to [50], we determine the surge's end t_f from the data as the second zero-crossing in the acceleration time series (see Figure 6.2c).[9] The reason for this choice of t_f is the observation that in one-dimensional pointing tasks movements are typically represented by a smooth and symmetric acceleration time series consisting of a positive peak followed by a negative one (or vice versa for movements in the opposite direction). If the target is not reached at the end of this "N-shaped" phase, typically the same acceleration pattern is repeated with lower amplitude and smaller period, suggesting our interpretation of corrective movements.
As illustrated in Figure 6.2 (blue solid line), applying MinJerk up to t_f results in a fairly good fit of the surge phase, at least for this particular data.[10] However, if the target is not reached at time t_f, as it is the case for many tasks with ID > 2, the overall fit is poor, which is reflected in a high SSE value.

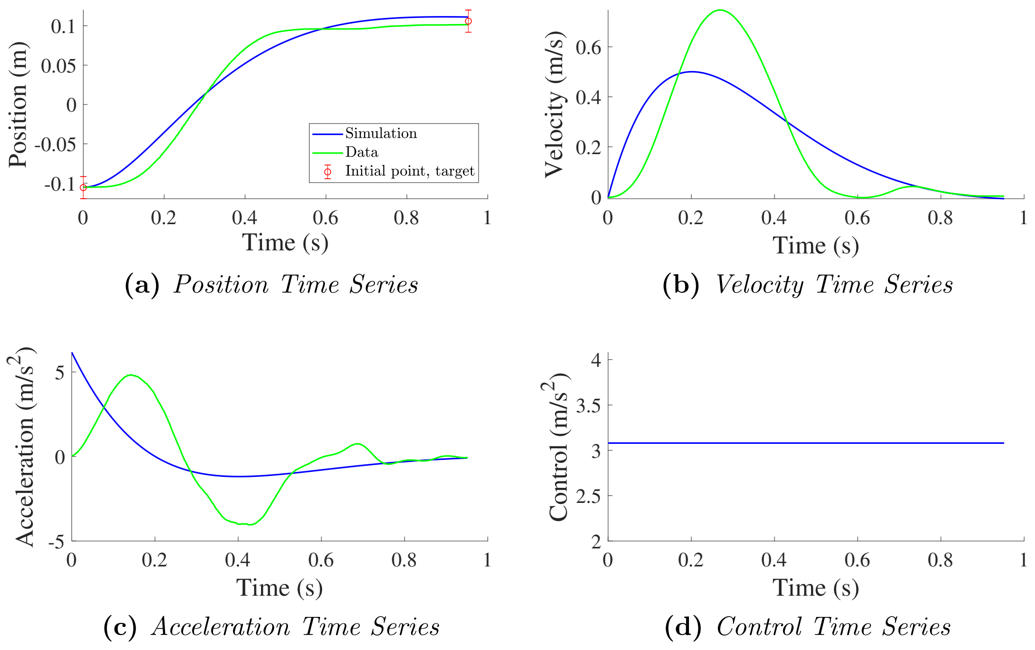
We therefore conclude that although MinJerk appears to be a good model for looking purely at the surge phase, it is not suitable for describing movements like those of the Pointing Dynamics Dataset, which involve an extensive correction phase.[11]

[8]Note that despite the similarity to the behavior of 2OL-LQR$_1$, the smoothness of the trajectory can be guaranteed here since the pure jerk is minimized instead of some combination of jerk and accuracy.

[9]For irregular acceleration time series without second zero-crossing we set $\tilde{N} = N$, i.e., in such cases both variants coincide.

[10]If the movement does not exhibit a clear surge phase, i.e., the first period of the acceleration time series is not clearly identifiable as it does not have the typical course but already contains some sort of corrective impulses towards the end, the fit is expected to deteriorate for any threshold-based method for determining t_f. In fact, this is the case for some trials of some participants in the dataset, as shown in Figure 6.3 using a representative trajectory.

[11]However, Hoff and Arbib's *Dynamic MinJerk model* [32], which was briefly discussed in Section 3.3.1, seems to be a promising approach for modeling such submovements that we did not pursue any further.



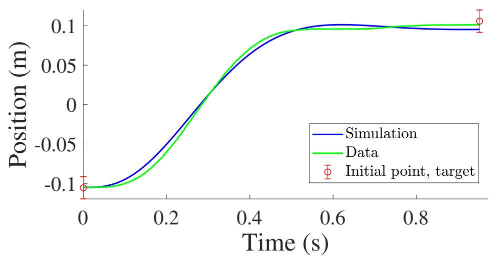
(a) *Position Time Series* **(b)** *Velocity Time Series* **(c)** *Acceleration Time Series* **(d)** *Control Time Series*

Figure 6.4: *2OL-Eq with constant control (blue) yields a much less symmetric surge phase than the user data (green), particularly in velocity and acceleration.* [SSE(k, d) = 0.0892 `with` $k = 29.1$, $d = 8.2$; `P11, ID 4 (765,51), 33th movement to the right`]

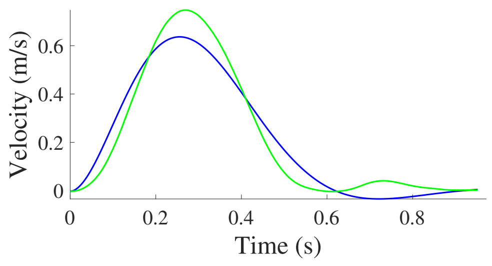
2OL-Eq

For the 2OL-Eq model given by (2OL-Eq) from Section 3.2.2 with $u \equiv T$, we optimize the spring stiffness k and the damping factor d with the same least squares parameter fitting process and the same SSE objective function (5.9) that we use for our 2OL-LQR model. In particular, the admissible parameter sets are of the form $\Lambda = (k, d) \in]0, \infty[\times]0, \infty[$.

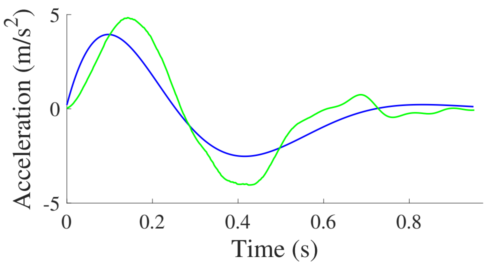
The behavior of 2OL-Eq is illustrated in Figure 6.4. Visually, the model captures the user behavior comparatively well in terms of pointer position, cf. Figure 6.4a. The velocity time series depicted in Figure 6.4b, however, is asymmetric in the 2OL-Eq case, while the user shows a more symmetric and bell-shaped velocity profile during the surge phase. The biggest difference appears in the acceleration time series (see Figure 6.4c). While the user, as described above, performs a symmetric and smooth N-shaped acceleration, the acceleration of 2OL-Eq jumps instantaneously at the start of the movement, and then rapidly declines. As a result, the details of the position time series are not reproduced exactly. In particular, a faster increase of the position at the beginning of the movement as well as a "more balanced", that is, a less pronounced transition from the surge to the correction phase (which in Figure 6.4 approximately starts at $t = 0.7$) can be observed in the position time series.
These observations can be explained by the physical interpretation of the 2OL-Eq as a spring-mass-damper system: Since u is constant in this model, as the system is released, the spring



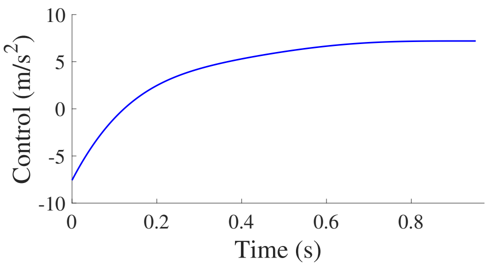
(a) *Position Time Series*



(b) *Velocity Time Series*



(c) *Acceleration Time Series*



(d) *Control Time Series*

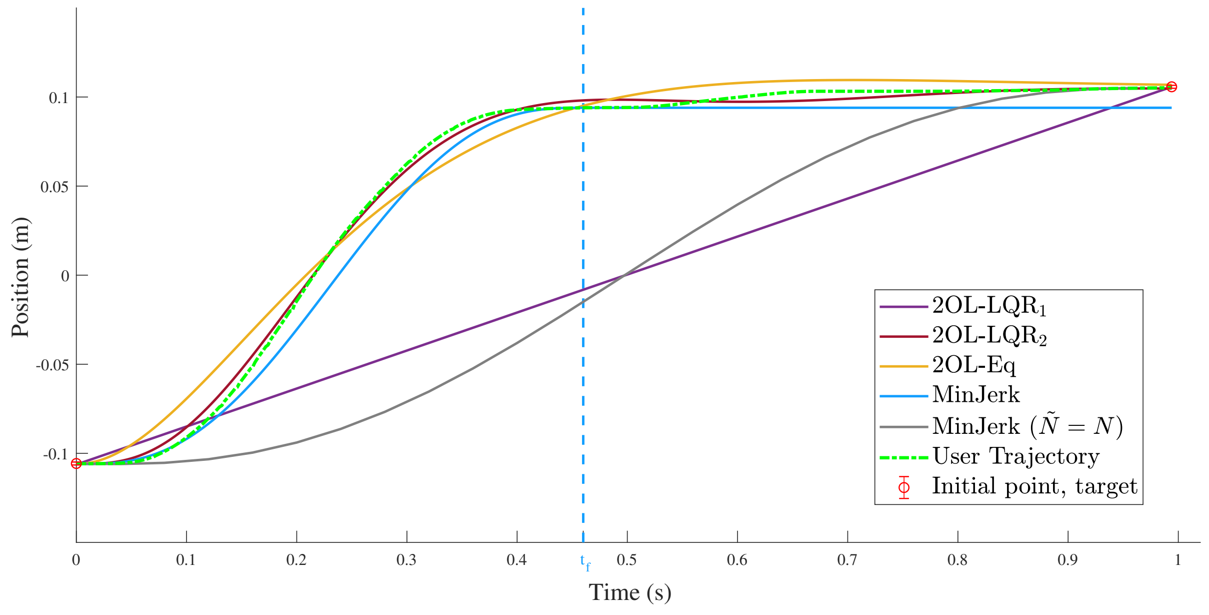
Figure 6.5: *Our second iteration model 2OL-LQR$_2$ models the entire movement well. However, the acceleration in the surge phase is slightly less symmetric than the one of this user.* [$\mathrm{SSE}(r,k,d) = 0.0131$ `with` $r = 5.86 \cdot 10^{-6}$, $k = 73.87$, $d = 7.53$; `P11, ID 4 (765,51), 33th movement to the right`]

instantaneously accelerates the system with a force that is proportional to the extension of the spring, i.e., to the remaining distance to target. Because human muscles cannot build up force instantaneously [62], this behavior is not physically plausible.

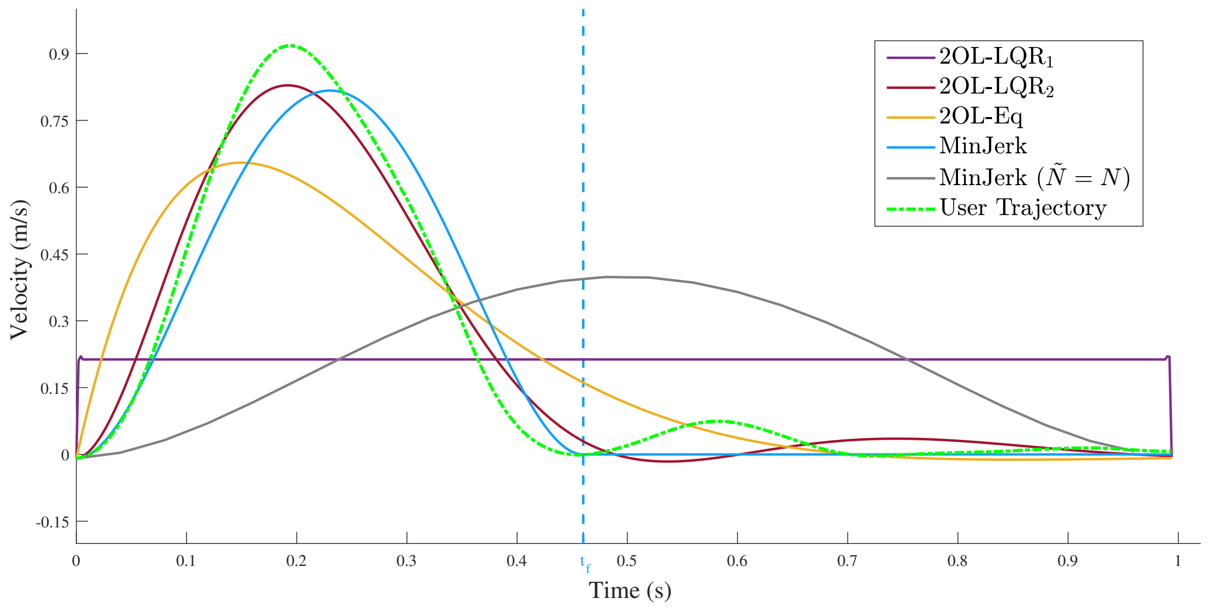
2OL-LQR$_2$

The behavior of our 2OL-LQR$_2$ algorithm is examplarily shown in Figure 6.5 for the same representative trial trajectory we used in Figure 6.4. The solution trajectory approximates the position time series well over the entire movement, cf. Figure 6.5a. In particular, the characteristics both at the beginning and at the end of the surge phase are reproduced quite accurately. More visible differences occur in the velocity time series, where our model slightly underestimates the maximum velocity and the velocity profile is a little less symmetric than the user's (see Figure 6.5b). In particular, this leads to a insufficient approximation of the second spike in velocity which can be seen as an indication of a second submovement.[12]
Similar effects can be observed in the acceleration time series: As can be seen in Figure 6.5c, both the maximum acceleration and the maximum deceleration are slightly underestimated. In addition, the acceleration profile of the simulation trajectory is less symmetric than the

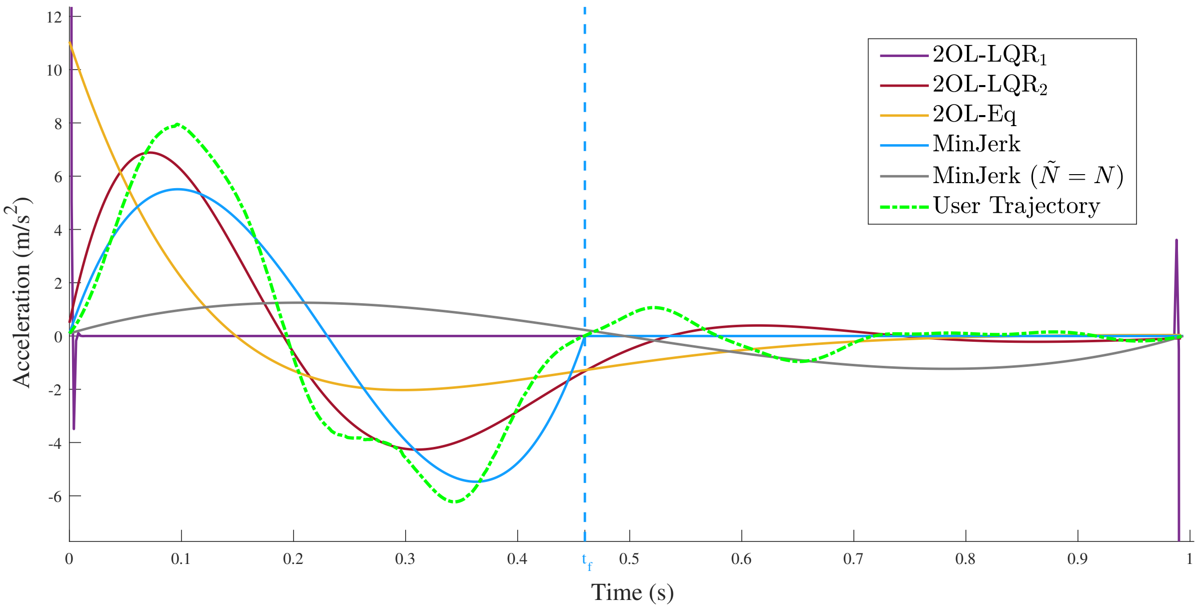
[12]For a detailed description of this well-known phenomenon, see, e.g., [20] or [45].



(a) *Position Time Series*



(b) *Velocity Time Series*



(c) *Acceleration Time Series*

Figure 6.6: *Solution trajectories of all models considered so far applied to the same trial trajectory* `[P1, ID 6 (765,12), 30th movement to the right]`*, which is shown as a green line. 2OL-LQR$_1$ (purple line) and MinJerk with $\tilde{N} = N$ (grey line) approximate the user trajectory by far the worst. 2OL-Eq (yellow line) provides a quite acceptable position time series, but especially with regard to acceleration, large deviations from the actual user movement become obvious. 2OL-LQR$_2$ (red line) and MinJerk applied up to t_f (blue line) come closest to the actual movement, but the latter misses the target due to the omitted correction phase, resulting in a worse overall fit.*
For reasons of clarity, the extreme acceleration values of 2OL-LQR$_1$ at the beginning (values up to 110) and at the end (values down to −113) of the movement are truncated in **(c)**.

Model	SSE	Maximum Error
2OL-LQR$_1$	2.3433	0.1171
2OL-LQR$_2$	0.0051	0.0074
2OL-Eq	0.0529	0.0241
MinJerk	0.0431	0.0190
MinJerk ($\tilde{N} = N$)	2.7491	0.1408

Table 6.1: *SSE and maximum error values of the trajectories depicted in Figure 6.6.*

user's since its surge part is not completely N-shaped, but has a more extended second half which fluently merges into a hinted correction phase with much smaller amplitude.[13] The used optimal control sequence u^* depicted in Figure 6.5d rises relatively fast at the beginning in order to initiate the movement, then it increases slower over time and finally, it is nearly constant during the correction phase.

In summary, the major improvement of our 2OL-LQR$_2$ algorithm over MinJerk is the capture of the entire movement within one plausible model, including an arbitrary long correction phase. Moreover, our model explains the surge movement similarly well as MinJerk does when applied to the surge phase only. The symmetry observed in many user acceleration time series, however, is not quite as well captured by our 2OL-LQR$_2$ model due to its not perfectly N-shaped acceleration time series.
Compared to 2OL-Eq, our algorithm captures position, velocity and acceleration time series noticeably better. The reason for this is that the control time series, in contrast to that of 2OL-Eq, is not constant, but changes optimally over time. This apparently leads to a more N-shaped acceleration time series and a more bell-shaped velocity time series, as predicted by Flash and Hogan [23] and widely confirmed by our user data.
These conclusions can also be seen from Figure 6.6, where the solution trajectories of all considered models applied to the same user trajectory are presented. The respective SSE and maximum error values can be found in Table 6.1.

Last but not least, there are some cases in which asymmetric acceleration time series do occur, cf. Figure 6.7. Due to its two-stage optimization process including both the parameter fitting and the minimization of an objective function corresponding to (a specific combination of) the assumed user objectives, our 2OL-LQR$_2$ algorithm is very flexible and thus even able to approximate such profiles reasonably well, whereas both other models are limited to a specific type of acceleration profiles, e.g., N-shaped acceleration profiles for MinJerk.
Moreover, we want to point out that while MinJerk requires the exact position, velocity and acceleration of the end point, our 2OL-LQR$_2$ model does not need this information, but adjusts its parameters to ensure the best overall fit of the trajectory.

6.2.2 Quantitative Comparisons

In the following, we consider all[14] trial trajectories of all participants and all tasks for quantitative comparisons. The resulting sum squared error values of all three models are shown

[13]Some of the irregularities in the user's acceleration profile might be traced back to insufficiently smooth filtering as the second derivative of our SG filter with degree $\nu = 4$ is only composed of quadratic functions.

[14]In fact, we decided to remove another 30 trajectories from the usable 7732 trajectories due to numerical instabilities of our algorithm, which were caused by extremely high optimal parameters k and d and led to erroneous calculations of the optimal control sequence u^*. For better comparisons, we decided to omit these trials for all considered models. However, we expect that our algorithm would give correct results if we set a reasonable upper bound for these parameters.

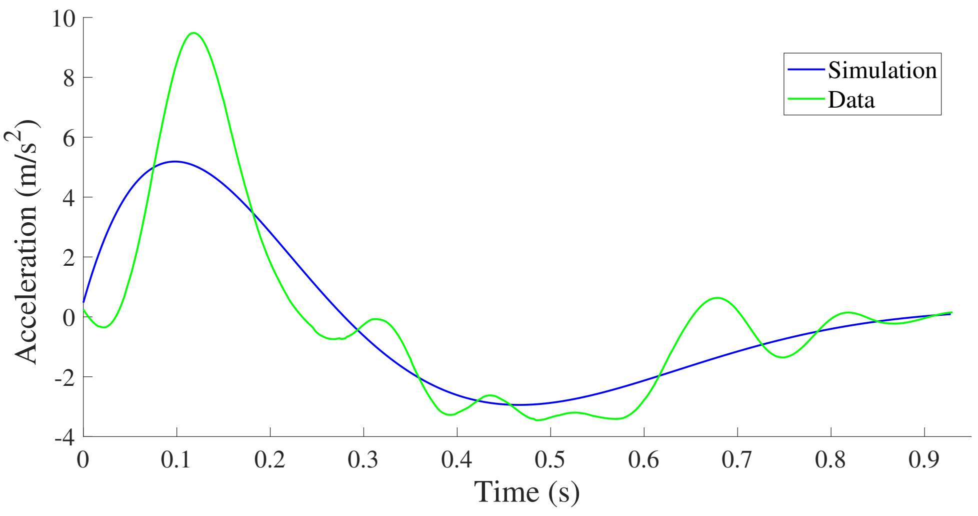


Figure 6.7: *While most movements, such as those in Figure 6.5 and Figure 6.6, have (almost) symmetric acceleration profiles, in other cases such as this one it is more asymmetric. Our 2OL-LQR$_2$ algorithm handles both cases reasonably well.* [SSE$(r, k, d) = 0.0079$ with $r = 8.97 \cdot 10^{-6}$, $k = 50.32$, $d = 8.9$; P7, ID 6 (1275,20), 23th movement to the right]

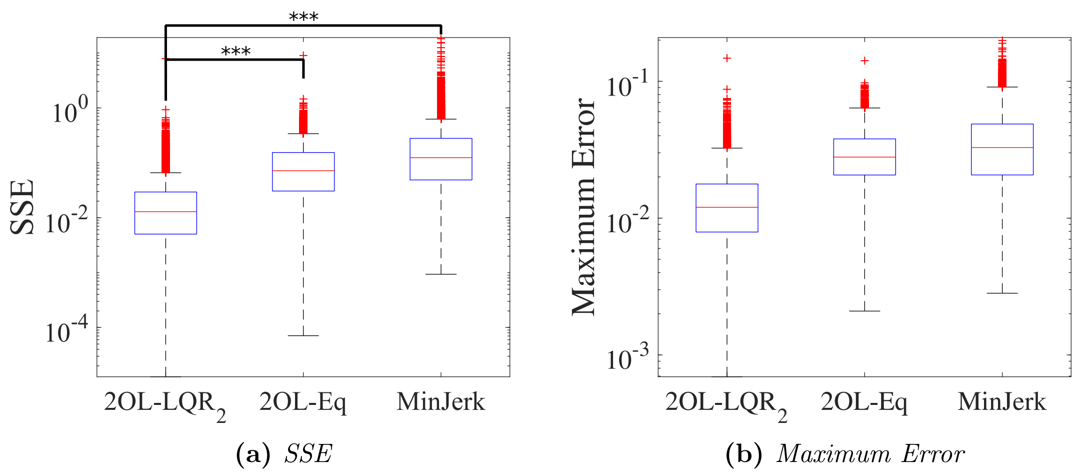


Figure 6.8: *SSE and maximum error values of our 2OL-LQR$_2$ algorithm compared to 2OL-Eq and MinJerk, each applied to the trial trajectories of all participants and all tasks (logarithmic scale). The three asterisks between two models indicate that their values are significantly different with p-value $p < 0.001$.*

in Figure 6.8a, on a logarithmic scale. In addition, the respective arithmetic mean values, standard deviations, and standard errors, i.e., the standard deviations divided by the square root of the number of considered trials (in our case 7702), can be taken from the first three columns of Table 6.2. As it appears, our model 2OL-LQR$_2$ is able to capture human behavior considerably better in terms of SSE than both the 2OL-Eq and MinJerk models. However, we decided to carry out some statistical tests, which are briefly described below, to obtain a more substantiated analysis.

Kolmogorov-Smirnov tests [18] with significance level $\alpha = 0.05$ showed that the distributions of SSE for the three models do not fit the assumption of normality, i.e., the mismatch between the SSE values and those of the normal distribution with appropriate parameters was too large for each model (with all p-values $p < 0.0001$).

Thus, we carried out the non-parametric *Friedman test* [24], which uses the SSE value ranks of all included 7702 trial trajectories. The main factor included in the analysis was which model was used: 2OL-LQR$_2$, 2OL-Eq, and MinJerk. The significance level was again set at $\alpha = 0.05$. The test indicated that the SSE between the three models was significantly different, since the probability that the χ^2-distribution with 2 degrees of freedom is greater than or equal to the observed test statistic $\chi_2^2 = 8661.76$ amounts to less than 0.001, which is well below the significance level α.

Additional two-sided *Wilcoxon Signed Rank tests* [58, 71] with *Bonferroni corrections* [47] showed that the SSE was significantly lower in the 2OL-LQR$_2$ model when compared to the other models, since the z-score of the test statistic, i.e., the standardized sum of the signed ranks (which was $Z = -74.87$ compared to 2OL-Eq and $Z = -72.16$ compared to MinJerk) had in both cases an absolute value greater than the critical value $z_{1-\frac{\tilde{\alpha}}{2}}$ computed from the standard normal distribution using the Bonferroni-adjusted significance level $\tilde{\alpha} = \frac{1}{60}$ ($p < 0.001$ applied for the respective p-values).

In addition, we measured the **maximum error**, i.e., the maximum deviation of each of our simulation trajectories from the corresponding user trajectory:

$$\max_{n=1,\dots,N} |p_n^{\Lambda} - p_n^{\text{USER}}|. \tag{6.15}$$

The findings, which are depicted in Figure 6.8b and summarized in the last three columns of Table 6.2, are comparable to those for SSE according to the same Kolmogorov-Smirnov tests ($p < 0.0001$ for each model) and the subsequent Friedman Test ($\chi_2^2 = 9026.92$, $p < 0.001$). Analogously, Wilcoxon Signed Rank tests resulting in $Z = -75.91$ compared to 2OL-Eq respectively $Z = -72.25$ compared to MinJerk ($p < 0.001$ applies in both cases) allows to conclude that 2OL-LQR$_2$ also approximates user trajectories significantly better than both 2OL-Eq and MinJerk in terms of maximum error.

It is important to emphasize that the summary statistics of both measures, given in Table 6.2 for all three models, do not allow any direct conclusions to be drawn about the *average performance improvement* through the use of our model. To this end, we decided to compute

Model	SSE			Maximum Error		
	Mean	SD	SE	Mean	SD	SE
2OL-LQR$_2$	0.0265	0.0991	0.0011	0.0138	0.0088	0.0001
2OL-Eq	0.1121	0.1576	0.0018	0.0301	0.0132	0.00015
MinJerk	0.2572	0.5762	0.0066	0.0373	0.0224	0.00025

Table 6.2: *Arithmetic mean, standard deviation (SD), and standard error (SE) of the SSE and maximum error values of each model applied to all considered trial trajectories.*

the ratios of performance *before* averaging[15], which led to the following findings:
While the SSE of 2OL-LQR$_2$ was on average only 18.5% of the SSE of 2OL-Eq, the individual SSE ratios differed considerably: At best, the SSE could be reduced by our model to 0.2% of the respective value of 2OL-Eq. Nevertheless, there were several trial trajectories worse approximated by our model than by 2OL-Eq, resulting in an up to 4.2-times higher SSE value. However, this was to be expected due to the large number of trajectories considered, including those already well approximated by 2OL-Eq. Particularly, the improvements by our 2OL-LQR$_2$ model regarding the SSE were naturally the least (or rather, the deteriorations were the greatest) for the relatively simple ID 2 tasks, which were apparently very well approximated by all the models considered. On the other hand, there were only six user trajectories among all tasks with ID 4, 6, and 8, which 2OL-Eq approximated better than our model in terms of SSE (on average of these tasks, our model reduced the SSE to 15.6% of the respective value of 2OL-Eq).
Similar (albeit slightly weakened) results were observed with respect to the maximum error: On average of all trials, the maximum distance between simulation and corresponding user trajectory using our model was only 42% of the maximum distance using the 2OL-Eq model. However, it is important to note that the simulation trajectories of both models were optimized with respect to SSE rather than maximum error, that is, even better results regarding only the maximum error may be achieved for each of the two models.
MinJerk (with extended end point from time t_f as defined above) had much higher mean and standard deviation values than 2OL-LQR$_2$ (see Table 6.2), which is mainly due to the different length of the correction phases that extremely bias the fit of MinJerk trajectories. In addition, the fixed method used to determine t_f might not always yield the best demarcation between the surge phase, which is approximated by MinJerk, and the neglected correction phase.[16] Both are also reasons why we could observe a ratio of the SSE value

[15]For the averaging process, we used the *geometric mean* in each case because, for example, two options should be on average equivalent if each is twice as good as the other in one out of two cases. In fact, using this geometric mean (as opposed to the arithmetic mean shown in Table 6.2), it makes no difference whether the values are averaged before or after computing the respective ratio between two models.

[16]However, we do not expect an additional optimization of t_f through an analogous parameter fitting, which necessarily involves *integer least squares*, to generally produce significantly better results than our model. For instance, the approximation of an asymmetric acceleration profile (which occurred in some cases, see Figure 6.7) by a fully symmetric acceleration profile extended by zero cannot be highly accurate.

Compared Models		SSE Ratio			Maximum Error Ratio		
		Mean	Min	Max	Mean	Min	Max
All IDs	2OL-LQR$_2$/2OL-Eq	0.1852	0.0021	4.1727	0.4196	0.0507	1.7854
	2OL-LQR$_2$/MinJerk	0.0951	0.0001	8.5251	0.3665	0.0166	2.2676
Only ID > 2	2OL-LQR$_2$/2OL-Eq	0.1559	0.0029	1.7101	0.4044	0.0670	1.3937
	2OL-LQR$_2$/MinJerk	0.1510	0.0005	8.5251	0.4444	0.0300	2.2676

Table 6.3: *Geometric mean, minimum, and maximum values of the SSE and maximum error ratios between different models, either considering all trial trajectories or only those from tasks with ID > 2.*

using 2OL-LQR$_2$ to the SSE value using MinJerk of 0.01% for some trial trajectories on the one hand, while on the other hand there were user trajectories which the 2OL-LQR$_2$ model approximated 8.5-times worse than the MinJerk model. However, the average SSE ratio of 2OL-LQR$_2$ to MinJerk amounted to only 9.5%, i.e., the improvement of our model over MinJerk was larger than that over 2OL-Eq.[17]
The results observed regarding the maximum error were little surprising: Here, the ratio of 2OL-LQR$_2$ to MinJerk amounted to an average of 36.6%.
Most of these findings can also be seen in summarized form in Table 6.3.

In summary, our model approximated the trial trajectories of different tasks and from different participants in most cases much better than both the 2OL-Eq and the MinJerk model. Moreover, an average performance improvement could be clearly established: The (mean) squared deviation between model trajectories and corresponding user trajectories[18] was significantly larger for both 2OL-Eq and MinJerk than for our model (on average more than five times larger for 2OL-Eq and more than ten times larger for MinJerk), and also the maximum deviation within a trial was on average about two to three times as large when using one of the other two models.

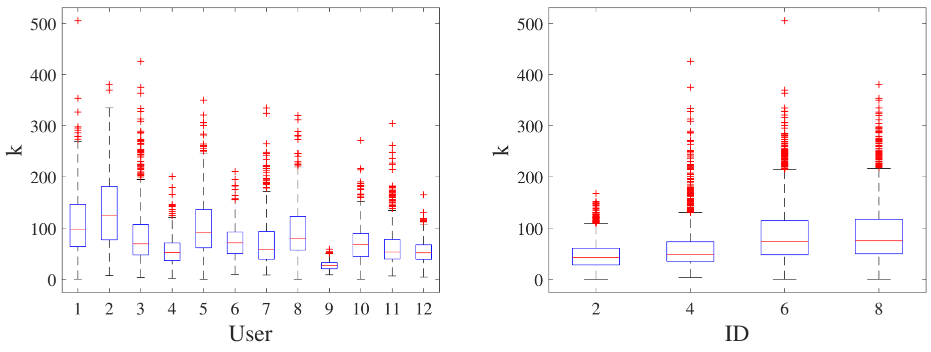
6.2.3 Parameter Distribution

Our model does not only lead to a generally better approximation of user trajectories, but also allows conclusions to be drawn from the optimal parameters about the abilities and intentions of participants during task execution.

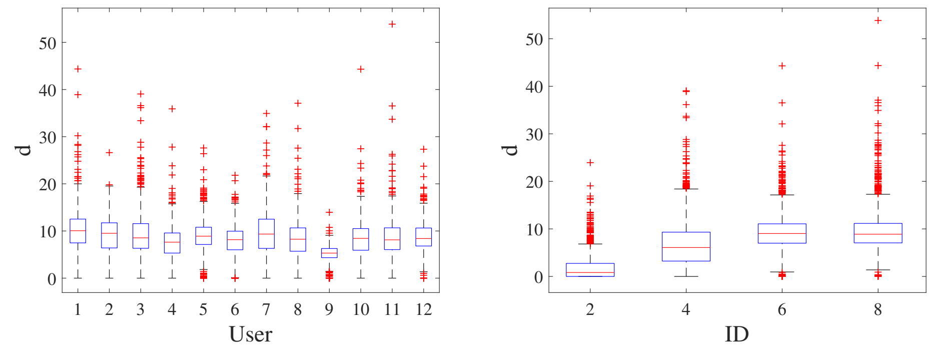
Figures 6.9a-6.9c (left) show the ranges of the three 2OL-LQR$_2$ parameters k, d, and r, optimized for the trial trajectories of all participants and all tasks with ID > 2, grouped

[17]However, this was mainly due to ID 2 tasks, which are comparatively poorly approximated by MinJerk: Considering only tasks with ID > 2, the average SSE ratio of 2OL-LQR$_2$ to MinJerk amounted to 15.1%, while the average ratio of 2OL-LQR$_2$ to 2OL-Eq amounted to 15.6%.

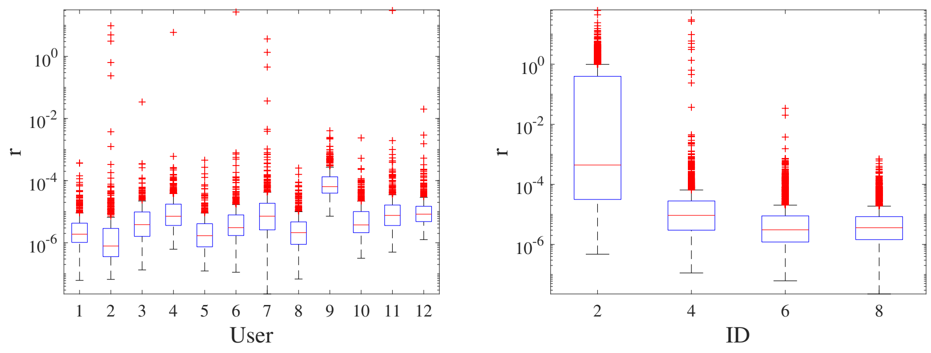
[18]Note that for any fixed user trajectory, all trajectories to be compared have the same length, i.e., an improvement in SSE can also be interpreted as an *average* shorter distance between model and user trajectory.



(a) *Parameter k*



(b) *Parameter d*



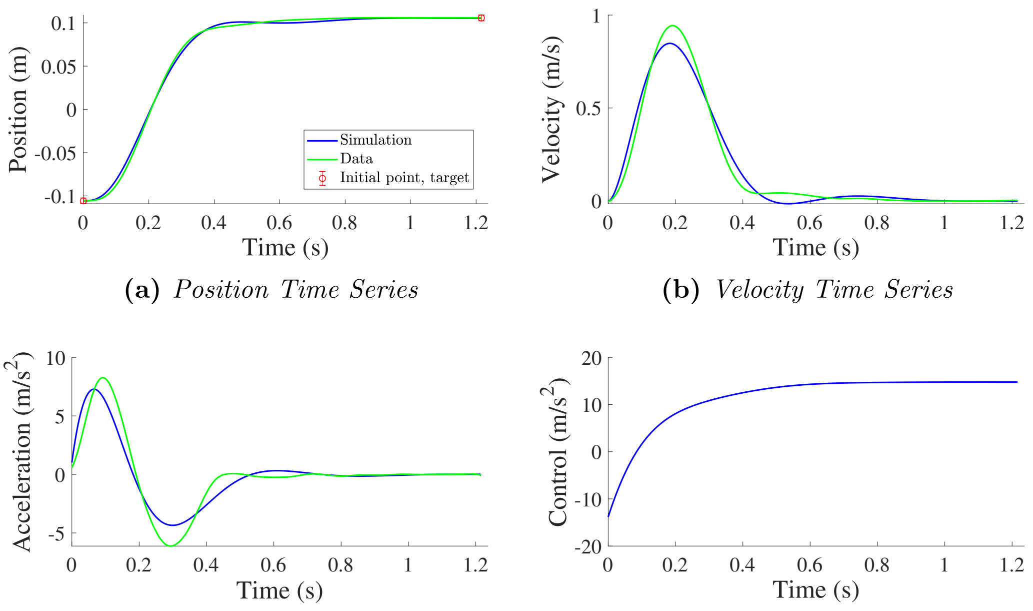
(c) *Parameter r (logarithmic scale)*

Figure 6.9: *Parameters of our 2OL-LQR$_2$ model, optimized for all considered trial trajectories of all participants and all tasks, grouped by participants (left, only ID 4, 6, and 8 tasks) and by ID (right). In the plots for the parameter d, the five biggest outliers, which assumed values between* 58 *and* 181*, were omitted for better clarity.*

by participants.[19] Indeed, different participants are characterized by differing parameter sets. For example, participant 2 is characterized by high spring stiffness k, above-average damping d, and very low jerk weight r. In contrast, participant 9 is characterized by very low spring stiffness k, very low damping d, and very high jerk weight r. The differences between such *user-specific parameter values*, however, strongly depend on the respective parameter: While most participants have a pretty similar spring constant k, the damping factor d and especially the jerk weight r (note the logarithmic scale) differ more clearly.
Remember that according to our definition of the "jerk" terms $(j_n)_{n\in\{1,\dots,N-1\}}$, a higher jerk penalization forces less rapid changes in the control $u = (u_n)_{n\in\{1,\dots,N-1\}}$. Since these control values u_n correspond to the force applied by the user due to the used system dynamics (4.39), the jerk weight r obtained from the parameter fitting process could be interpreted as the *effort* the user is willing to put into the task (see discussion in Section 4.2.3). For example, participant 9 could be regarded as rather lazy, while participant 2 might be most committed to our pointing tasks.

Figures 6.9a-6.9c (right) illustrate the ranges for k, d, and r, optimized for the trial trajectories of all participants and all tasks, grouped by ID. All three parameters show characteristic variations by ID.
The spring stiffness k increases considerably from ID 4 to ID 6, while on average there is little difference between ID 2 and ID 4 tasks as well as between ID 6 and ID 8 tasks. The truth is, however, that a meaningful interpretation of this observation is hardly possible. In particular, not even the effects of k on the course of the solution trajectory are well understood. The reason for this is that, in contrast to pure motion dynamics models such as 2OL-Eq, our model incorporates (apart from the parameter fitting) another optimization process for any given parameter set $\Lambda = (r, k, d)$, which leads to the respective used control sequence u^*. The spring constant k therefore influences the state sequence $x = (x_n)_{n\in\{1,\dots,N\}}$, which is given by (4.39) in our model, not only directly via the system matrix A but also indirectly via the optimal control sequence u^* (which additionally depends on d and r, further complicating the total analysis). A plausible relationship between the stiffness k and some key properties of the solution trajectory such as peak velocity, which, for instance, usually increases with distance, can therefore not be inferred (in contrast to motion dynamics models including nonlinear stiffness [49], for example).
Regarding the damping d, the observed growth in ID could possibly be logistic, but for the same reason mentioned above it is difficult to interpret this effect within our model. However, the considerably lower damping in ID 2 tasks is consistent with the identified oscillatory behavior of the participants (referring to the entire reciprocal movement sequence), which results from the simplicity of such tasks.
These oscillations might also play a role in the large variance of r that we observed for ID 2 tasks. For larger IDs, however, r is virtually constant. We can thus conclude, that the effort

[19]The inclusion of ID 2 tasks had no qualitative effects, but only led to considerably more outliers due to the underdetermination of the parameters in these simple tasks.



(a) *Position Time Series* **(b)** *Velocity Time Series*

(c) *Acceleration Time Series* **(d)** *Control Time Series*

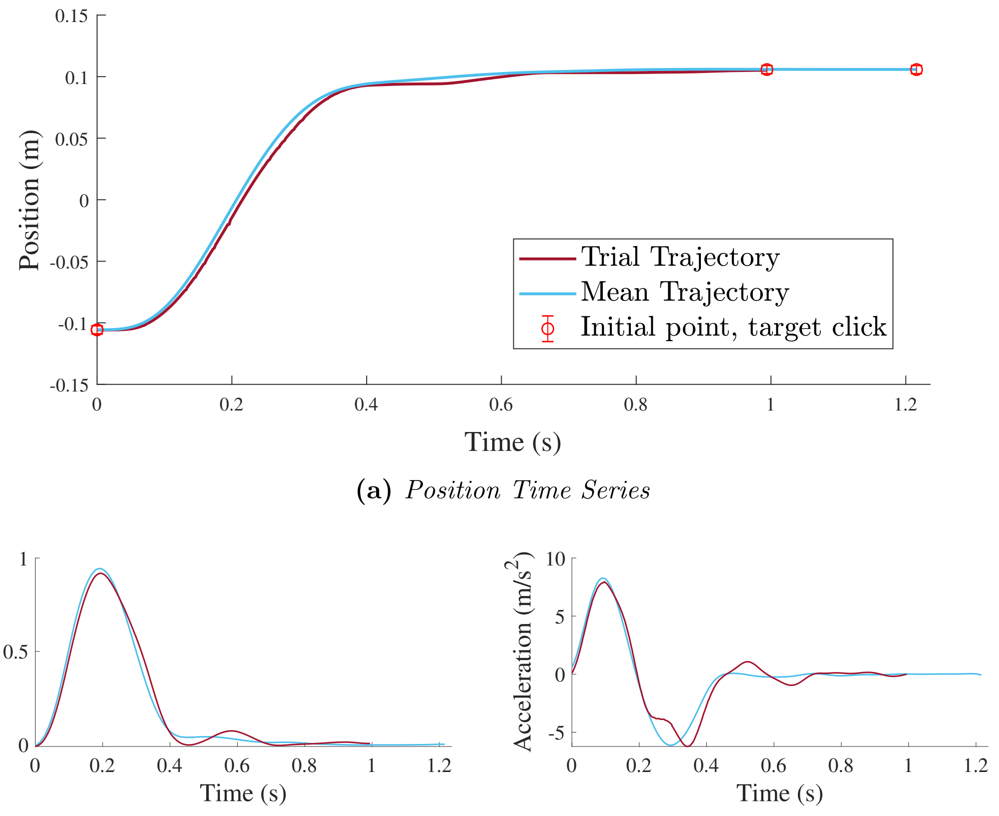
Figure 6.10: *Mean trajectories are similarly well approximated by our 2OL-LQR$_2$ model as individual trial trajectories.* `[`SSE$(r, k, d) = 0.0046$ `with` $r = 8.51 \cdot 10^{-7}$, $k = 140.17$, $d = 11.9$; `P1, ID 6 (765,12), mean trajectory]`

the users are willing to put into the task essentially depends on the users themselves and not on the difficulty of the task (apart from the irregularity observed for very simple tasks).

6.2.4 Mean Trajectory Analysis

In this section, we want to briefly compare the above presented results, which we obtained from the application of the three considered models to trial trajectories, to those obtained from the application to mean trajectories (see Section 5.2.2 for detailed explanation of these trajectories).

First of all, for each model, the approximation of mean trajectories hardly differs qualitatively from that of trial trajectories. This is exemplarily shown in Figure 6.10, where we applied our model to the mean trajectory corresponding to the trial trajectory from Figure 6.6. However, we are mainly interested in the characteristics of the mean trajectories themselves, since we want to find out how informative a good approximation of these trajectories is.
As can be seen in Figure 6.11, the differences between some trial trajectory and its corresponding mean trajectory appear to be small at first glance. Looking more closely, however, we find that most notably the second spike in the velocity time series is less clearly pro-



(a) *Position Time Series*

(b) *Velocity Time Series* **(c)** *Acceleration Time Series*

Figure 6.11: *The trial trajectory (red line) from Figure 6.6 in direct comparison with its corresponding mean trajectory (blue line), whose approximation by our model is shown in Figure 6.10. Since the length of the mean trajectory corresponds to the length of the longest respective trial trajectory, the target click time differs between both trajectories, which is why we have drawn the target twice in the position time series (last two red error bars).*

nounced (and thus the second period in the acceleration time series as well). Moreover, the averaging process led to blurred acceleration profiles without abrupt twists and turns.[20] We therefore conclude that the qualitative properties of typical user movements are not completely lost but at least weakened through averaging.

This is also reflected in the quantitative analysis (see Tables 6.4 and 6.5):
The SSE of 2OL-LQR$_2$ applied to mean trajectories was on average less than two-thirds of the SSE of 2OL-LQR$_2$ applied to trial trajectories, which suggests that mean trajectories are for our model on average somewhat easier to approximate than "real" user trajectories. Similar findings were observed for MinJerk, which was to be expected as the mean trajectories were

[20]Note that the previously depicted velocity and acceleration profiles of trial trajectories were smoothed as well (in contrast to the respective position time series). The only difference in the derivative sequences is therefore the preceding averaging process.

Model	SSE – Trial Trajectories			SSE – Mean Trajectories		
	Mean	SD	SE	Mean	SD	SE
2OL-LQR$_2$	0.0265	0.0991	0.0011	0.0151	0.0130	0.0009
2OL-Eq	0.1121	0.1576	0.0018	0.1044	0.0685	0.0049
MinJerk	0.2572	0.5762	0.0066	0.1514	0.2263	0.0163

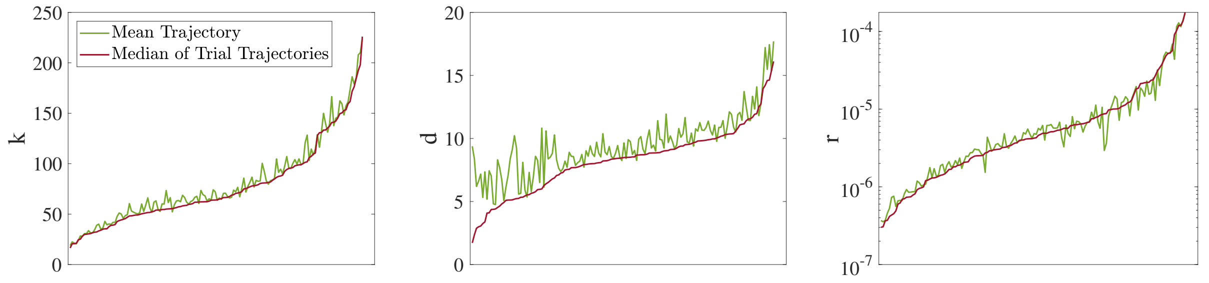
Table 6.4: *Arithmetic mean, standard deviation (SD), and standard error (SE) of the SSE values of each model, either applied to all trial trajectories or to all mean trajectories.*

Compared Models		SSE Ratio – Trial Traj.			SSE Ratio – Mean Traj.		
		Mean	Min	Max	Mean	Min	Max
All IDs	2OL-LQR$_2$/2OL-Eq	0.1852	0.0021	4.1727	0.1301	0.0121	0.7560
	2OL-LQR$_2$/MinJerk	0.0951	0.0001	8.5251	0.1340	0.0044	3.3348
Only ID > 2	2OL-LQR$_2$/2OL-Eq	0.1559	0.0029	1.7101	0.1264	0.0296	0.3959
	2OL-LQR$_2$/MinJerk	0.1510	0.0005	8.5251	0.1210	0.0044	3.3348

Table 6.5: *Geometric mean, minimum, and maximum values of the SSE ratios between different models, each applied to either trial or mean trajectories, considering either all tasks or only those with ID > 2.*

smoothed by some SG filter and are thus advantageous for a model that is completely based on maximum smoothness. For 2OL-Eq, however, hardly any differences could be observed between the average approximation of trial and mean trajectories. Altogether, these findings suggest (slightly) different model comparison results in terms of SSE, depending on whether trial or mean trajectories are considered.
Moreover, in contrast to trial trajectories, for *every* mean trajectory an SSE improvement of our model compared to 2OL-Eq could be observed, resulting in a maximum SSE ratio of less than one. Additionally, the ratio of the SSE using 2OL-LQR$_2$ to the SSE using 2OL-Eq amounted to 13% on average (using the geometric mean), while for trial trajectories this average ratio was 18.5%. Regarding the improvement over MinJerk, we could also observe minor deviations from the results for trial trajectories, but in the opposite direction: For mean trajectories, the average SSE ratio of 2OL-LQR$_2$ to MinJerk was 13.4%, which means a marginally lower improvement of our model compared to MinJerk than can be seen from the trial trajectories (where the ratio was only 9.5%).
Similar effects were observed when considering only tasks with ID > 2: Here, the SSE of our model made up on average 12.6% of the SSE of 2OL-Eq (15.6% for trial trajectories) and 12.1% of the SSE of MinJerk (15.1% for trial trajectories).

In addition, we were interested in the difference between the optimal parameters for individual trials and those for mean trajectories.
To this end, we decided to group all parameter values from trials of the same participant with



(a) *Parameter k* **(b)** *Parameter d* **(c)** *Parameter r (logarithmic scale)*

Figure 6.12: *The optimal parameters of our 2OL-LQR$_2$ model for mean trajectories (green) show clear similarities with the medians of the optimal parameters for the respective trial trajectories (red). The damping d tends to be slightly overestimated in mean trajectories.*

identical task condition and direction of movement. For each parameter, we then calculated the median[21] of each group and compared it to the parameter that fits to the corresponding mean trajectory, i.e., the optimal parameter of our model applied to the trajectory that consists of (arithmetic) mean values of exactly these grouped trial trajectories.
In Figure 6.12, both the median parameter values (red lines) and the parameter values for the corresponding mean trajectories (green lines) are plotted for all participants and all tasks with ID > 2.[22] Here, both parameter values are shown directly on top of each other for each of the resulting 144 groups (one for each combination of participant, task condition and direction), which are sorted by ascending median parameter values.
For both the spring constant k and the jerk weight r, the two parameter values agree surprisingly well in most cases. For the damping d, the damping factors that are optimal for the mean trajectories tend to be a little higher than the respective median damping values. Furthermore, the discrepancy between these pairs of comparable damping parameters tends to increase the smaller their values become, with the smallest median damping not occurring at all among the optimal damping factors for the mean trajectories. Nevertheless, the basic tendency also matches well in terms of damping for both variants.

All in all, the results of our model comparison slightly differ depending on the type of trajectories used, which in detail may lead to interpretations that are not directly transferable. However, the main effect, i.e., the substantially better approximation of user trajectories by our model than by 2OL-Eq and MinJerk, can be observed throughout.

[21]The use of the arithmetic mean instead of the median had little effect on the results discussed below, but in some cases it led to unusually high parameter values (especially for r) due to the strong influence of single outliers.

[22]We decided to omit ID 2 tasks since there are a lot of indications that the parameters are underdetermined in these cases and therefore not very meaningful.

Chapter 7

Modeling Reaction Time

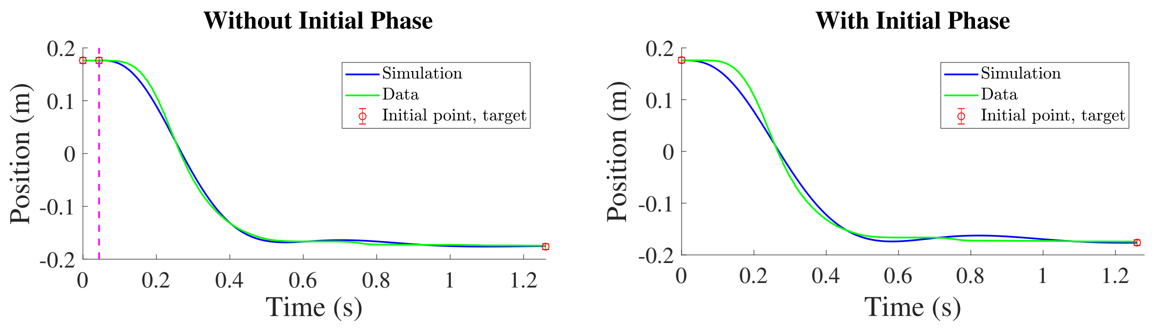
In Chapter 6.2, we have shown that the 2OL-LQR$_2$ variant of our algorithm approximates both trial and mean trajectories sufficiently well. However, while typical pointing movements involve an **initial phase (reaction time phase)** of variable length, in which the users presumably have not yet processed the new target information and thus are at least not moving directly towards the target (even though the upcoming target is basically known in advance), we have eliminated these phases in all trajectories considered so far. As can be seen in Figure 7.1 (right) using the example of a left movement of participant 5, our algorithm with 2OL-LQR$_2$ objective function is not able to reproduce such initial phases, which for this specific trajectory leads to a more than 3.5-times higher SSE value than for the same trajectory without initial phase (shown in Figure 7.1 (left)[1]). This means that in our previous variant of the 2OL-LQR model, even with optimal choice of parameters $\Lambda = (r, k, d)$, it is not possible to reproduce movements that only "start" after an essential initial phase, i.e., that do not rush forward from the beginning. However, the use of a fixed reaction time would not solve this problem, since the length of these initial phases varies greatly among different users and even among different trials of the same user, even if sufficient "training trials" of a previously unknown task have been carried out beforehand.[2] For this reason, we needed to adjust our cost function once again.

In particular, we introduced the parameter $n_\delta \in \{1, \dots, N\}$ to describe the time step at which the user mainly initializes its movement towards the target, i.e., the respective *reaction time step*.[3] Furthermore, we distinguished the jerk penalization weights r_n depending on whether $n < n_\delta$ or $n \geq n_\delta$ applies: While in the latter case r_n was set to the parameter r used so far, we wanted to achieve $r_n \approx cr$ for $n \in \{1, \dots, n_\delta - 1\}$, where c is a very large constant

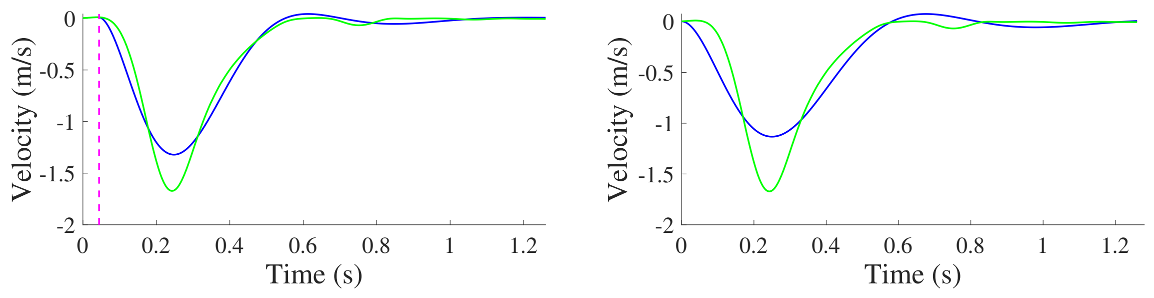
[1]Here, the initial phase of the user trajectory, which was previously removed from every trial trajectory, is additionally shown. For this reason, the simulation trajectory only starts at time $(n_\delta - 1)h > 0$, which is indicated by a second initial point.

[2]This observation could, for example, be attributed to different levels of attention or commitment in different trials [5].

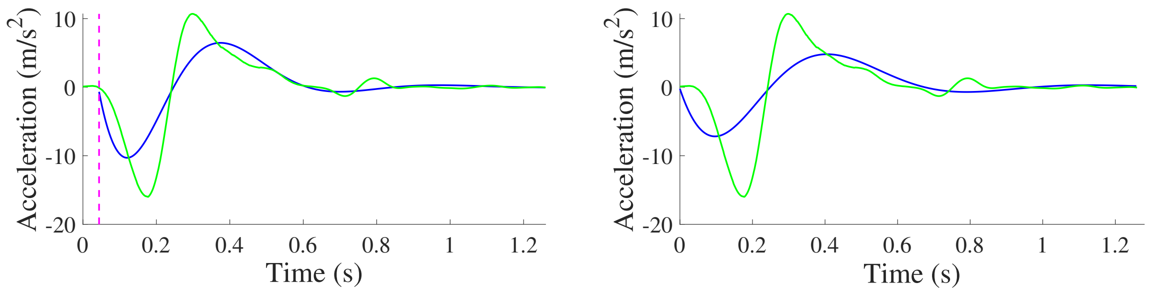
[3]Note that the previous "definition" of this reaction time step from (5.7) is thus nullified.



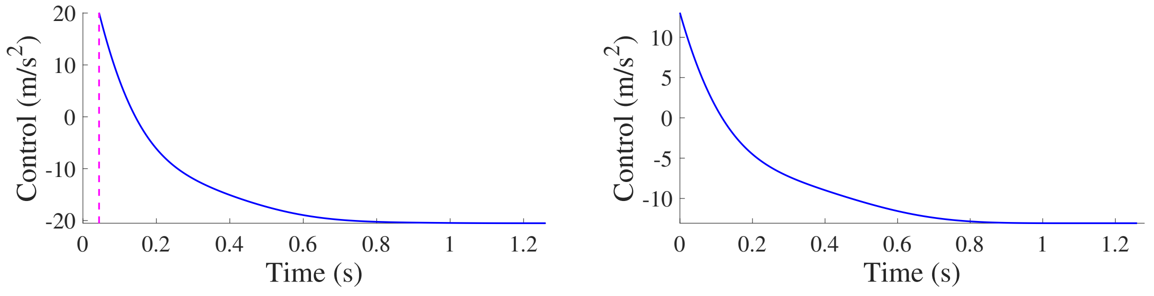
(a) *Position Time Series*



(b) *Velocity Time Series*



(c) *Acceleration Time Series*



(d) *Control Time Series*

Figure 7.1: *While our 2OL-LQR$_2$ model approximates trial trajectories starting at time step n_δ (dashed magenta lines) visibly well (left), its fit to entire trajectories including the initial phase is clearly worse (right).* `[P5, ID 6 (1275,20), 30th movement to the left]`

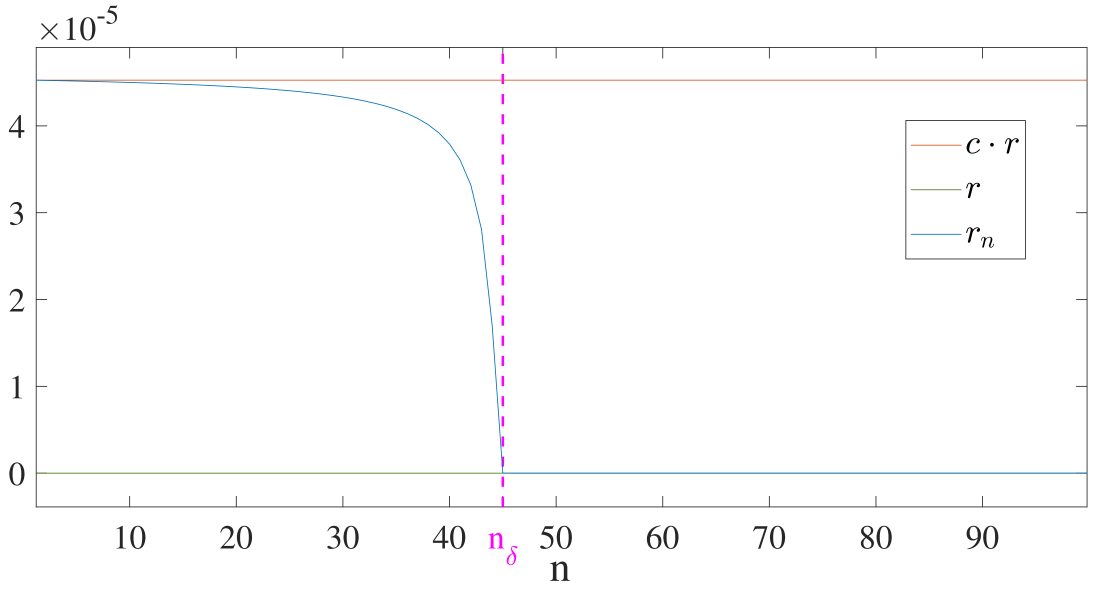


Figure 7.2: *The smoothed jerk weight sequence $(r_n)_{n\in\{1,\ldots,N\}}$ (blue line) first approximates $c\cdot r$ (red line) and then rapidly drops down to r (green line), which is constantly assumed from n_δ. We used $c = 100000$ and the values of r and n_δ originate from our approximation of the trial trajectory depicted in Figure 7.3.*

multiplication factor. In fact, we needed to implement a smoothed version of this piecewise constant sequence of jerk weights $c\cdot r$ and r for some technical reasons.[4] We thus defined the jerk weights as

$$r_n := \begin{cases} f(n)r, & \text{if } n \in \{1,\ldots,n_\delta - 1\}, \\ r, & \text{if } n \in \{n_\delta,\ldots,N\}, \end{cases} \tag{7.1}$$

where

$$f(n) := \exp\left(\frac{1}{n_\delta - 1} - \frac{1}{n_\delta - n}\right)(c-1) + 1, \quad n \in \{1,\ldots,n_\delta - 1\}, \tag{7.2}$$

is close to c for most of the time (see Figure 7.2).
The intrinsic jerk cost parameter r thus still indicates the weighting of smooth trajectories relative to early target achievement, but only after an initial phase of $n_\delta - 1$ time steps in which jerk is extremely penalized. The main reason for doing so was the assumption that in such initial phases users are either not aware of the target position or they are temporarily too exhausted to initiate the movement instantaneously. Whatever interpretation may apply, users have clearly no interest in changing their control extensively. Since in our model,

[4]Since the used least squares algorithm requires continuous parameters, arbitrarily small changes in n_δ must have an effect on the objective function in order to not get stuck for any initial parameter immediately. We therefore decided to include n_δ in the definition of the values that r_n assumes for $n < n_\delta$ as well.

jerk corresponds to the differences between consecutive controls, a very high jerk penalization during the initial phase (relative to the remaining jerk penalization) indeed creates an incentive to largely maintain the initial control throughout.
In order to avoid conflicts of objectives, it is therefore logical to penalize the remaining distance to target only *after* this initial phase, i.e., we set

$$Q_n := \begin{cases} 0, & \text{if } 1 \leq n < n_\delta, \\ \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}, & \text{if } n_\delta \leq n \leq N. \end{cases} \tag{7.3}$$

It remains to be clarified how to determine n_δ:
One possibility would be to extract it directly from the respective user trajectory, e.g., to define it as the time step at which a given small fraction δ of the maximum assumed acceleration is covered first (as we did in eliminating these initial phases above, see (5.7)). However, we decided to optimize n_δ together with the remaining parameters within our least squares parameter fitting process in order to maintain a procedure that does not require prior analysis of user trajectories, and also to provide a little more freedom to our system.[5]

Setting $c = 100000$, the resulting variant of our 2OL-LQR model (4.47), which we denote as **2OL-LQR**$_3$, thus has the following objective function:

$$\begin{aligned} J_N(x,u) &= \sum_{n=1}^{n_\delta - 1} f(n) r j_n^2 + \sum_{n=n_\delta}^{N} (T - p_n)^2 + \sum_{n=n_\delta}^{N-1} r j_n^2, \\ f(n) &= 99999 \exp\left(\frac{1}{n_\delta - 1} - \frac{1}{n_\delta - n}\right) + 1, \quad n \in \{1, \dots, n_\delta - 1\}. \end{aligned} \tag{7.4}$$

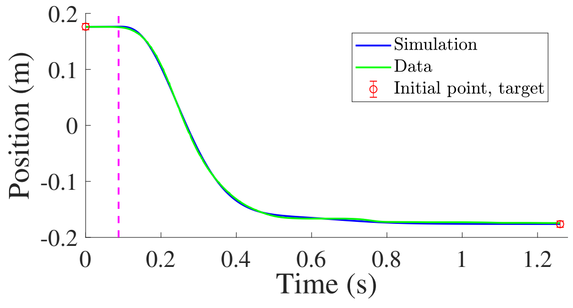
We again use this cost structure within our 2OL-LQR algorithm described in Section 5.3, where n_δ is additionally optimized (as a continuous variable, rounded afterwards), i.e., $\Lambda = (r, k, d, n_\delta) \in \mathbb{R}^4$ with $r, k, d > 0$ and $1 \leq n_\delta \leq N$.[6] The results are discussed in the following.

Results

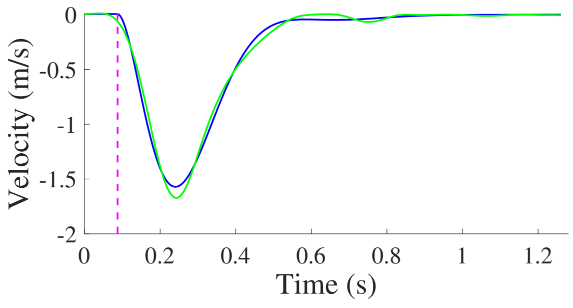
Using the 2OL-LQR$_3$ model, Figure 7.3 shows the approximation of the same randomly selected trial trajectory (including initial phase) we previously tried to reproduce by our 2OL-LQR$_2$ model (see Figure 7.1).

[5]Note that any general criterion for determining n_δ from the user trajectory would be quite arbitrary and thus not more plausible than using some parameter fitting process.

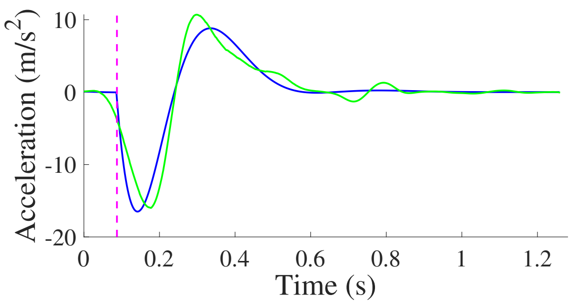
[6]Here, the initial parameter sets Λ_0 were selected as follows: The parameters r, k, d were randomly chosen from the continuous uniform distribution on the interval between 0 and $\varsigma_r = 10^{-10}$ respective $\varsigma_k = \varsigma_d = 100$, and the parameter n_δ was chosen sequentially equidistant between 1 and $\lfloor \frac{N}{3} \rceil$, which has proven to be useful.



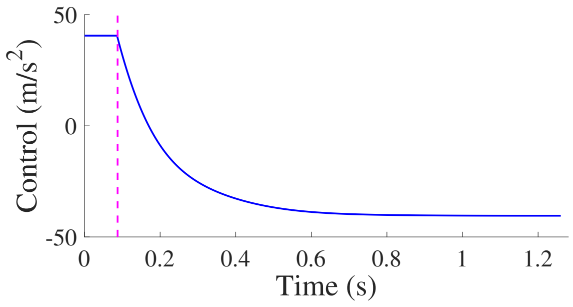
(a) *Position Time Series*



(b) *Velocity Time Series*



(c) *Acceleration Time Series*



(d) *Control Time Series*

Figure 7.3: *The additional parameter n_δ of our 2OL-LQR$_3$ model, represented by dashed magenta lines, allows to model individual movements including reaction time.*
[SSE$(r, k, d, n_\delta) = 0.0033$ `with` $r = 4.5 \cdot 10^{-10}$`,` $k = 229.72$`,` $d = 17.4$`,` $n_\delta = 45 \cong 88$ `ms; P5, ID 6 (1275,20), 30th movement to the left]`

Apparently, the enormous jerk penalization during the initial phase, i.e., up to $n_\delta - 1$, leads to a constant control which in turn prevents changes in acceleration. Since the initial acceleration and the initial velocity are (almost) zero, both velocity and position also remain unchanged up to the time step $n_\delta - 1$. Hence, we can actually interpret n_δ as reaction time in the sense of the time step at which the motion is initiated by the controller.

Regarding the approximation of user trajectories, 2OL-LQR$_3$ has two main advantages:
First, the additional parameter leads to a much better approximation of the complete movement including both initial phase and correction phase, qualitatively as well as quantitatively (e.g., for the user trajectory shown in Figure 7.1 and Figure 7.3, the SSE obtained when using 2OL-LQR$_3$ accounts for just 3.8% of the SSE obtained when using 2OL-LQR$_2$). In particular, 2OL-LQR$_3$ also succeeds in reproducing details of both the position and the (smoothed) velocity time series: Both the peak velocity and the second submovement, which caused the most difficulties for 2OL-LQR$_2$, are much better approximated here, although still not absolutely perfect.
Second, the jerk weight r is more meaningful in 2OL-LQR$_3$ because it is used there only for

the actual movement: Since motion delays are mainly determined by n_δ, the parameter r can thus be interpreted as the effort spent *after* the movement has been initiated.

However, looking closely at the acceleration at the beginning, we found that our model initiates the movement later than the user but with faster increasing acceleration. Similar findings were also observed for all other trajectories investigated.
The reason for this phenomenon is quite simple: The parameter n_δ is not optimized to approximate only the initial phase particularly well, but to maximize the fit of the *entire* position time series. Depending on the characteristics of the individual movement, the optimizer therefore does not necessarily select n_δ as the "real" reaction time but only loosely corresponding to it. In our case, for example, n_δ seems to be chosen slightly too large[7] in order to compensate for the acceleration, which apparently rises too fast at the beginning due to our used system dynamics (4.39).[8]

In summary, the additional parameter n_δ allows not only to model existing reaction times sufficiently, but also to extend the initial phases in order to better approximate the user trajectory in general, even if it loses any reasonable interpretation. Despite the many advantages of the 2OL-LQR$_3$ model, it is therefore advisable to apply 2OL-LQR$_2$ to user trajectories instead, at least for a starting point, to avoid some inexplicable "over-fitting", especially if it is not clear whether such initial phases are present in the user trajectory at all.

[7]Note that in Figure 7.3, where we optimized the reaction time step together with the other parameters, $n_\delta = 45$ applies, whereas in Figure 7.1, where we determined the duration of the initial phase from the trial trajectory using (5.7), $n_\delta = 23$ applies.

[8]This is probably because in the second-order lag (2OL), on which our dynamics is based, the control is necessarily proportional to the acceleration (except for additive terms such as spring stiffness and damping). The use of higher-order dynamics, e.g., based on a third-order lag, would therefore be an interesting approach for the future.

Chapter 8

Conclusion

At first glance, the execution of a mouse movement seems to be a very simple and straightforward action that we practice day after day without even thinking about it. However, it is the incredible and still only partially understood complexity of the human brain that makes it extremely difficult to predict the behavior of computer users. All models that attempt to capture human movements therefore work with simplifications, which, depending on the field of research, either concern the biomechanical apparatus, the neuro-muscular system, or the intentions and objectives of the users.
While in the first part of this thesis we presented (besides some theoretical basics) some of these models of motion dynamics, the main concern was to bring these different approaches together in a general framework. The LQR scheme, which already had been applied to pointing tasks by Todorov [67], albeit in a substantially extended variant, proved to be suitable for this purpose: Here, we were able to maintain the second-order linear dynamics of the 2OL-Eq using equality constraints and simultaneously take the user objectives that we assumed for pointing tasks, namely accuracy maximization, time minimization, and jerk minimization, into account. Moreover, we decided to include three easily interpretable system parameters: the spring stiffness k, the damping factor d, and the jerk weight r.
We were able to derive an analytical solution for our 2OL-LQR model, which yields the optimal trajectory for any task configuration and parameter set. In order to check the suitability of our model, we extended it by a least squares parameter fitting process. In the resulting 2OL-LQR algorithm, we wanted to find the parameter set for which some solution trajectory best approximates a given experimentally observed user trajectory in terms of SSE.

In an iterative design process, we were able to illustrate the problems (missing correction phase, implausible physical behavior) that arise when the positional error is penalized only at the end of the movement, as suggested by Todorov [67]. Furthermore, we succeeded in modifying the cost structure in such a way that both the trial trajectories we extracted from Müller's Pointing Dynamics Dataset [50] and the resulting mean trajectories were visibly well approximated by our model. In contrast, as was shown, two much-cited models from literature failed to do so: While MinJerk, similar to our first variant 2OL-LQR$_1$, could

not reproduce the correction phases of user trajectories, for 2OL-Eq it was the acceleration being necessarily proportional to the remaining distance that prevented a sufficiently good approximation. Moreover, the SSE (and also the maximum error) of our model was on average much lower than that of both 2OL-Eq and MinJerk.
We can thus conclude that our proposed 2OL-LQR$_2$ model is generally more capable of describing user movements and in particular of reproducing concrete user trajectories in the pointing tasks under consideration. Moreover, unlike Todorov's model, no final time has to be specified in order to obtain a reasonable approximation, which increases the applicability of our model, since in most real-world pointing tasks there exists no pre-specified temporal boundary for users, but their personal skills are the time-limiting factor. In addition to that, our algorithm provides a further benefit: As we have shown, it is possible to characterize different users by different stiffness and damping parameters and even draw conclusions about the effort a particular user put into the task, which might provide new insights into the complex interrelationships of human behavior.

Nevertheless, it is important to remember that our model (like most others) is based on some extreme simplifications of reality: In particular, the assumption of linear system dynamics and quadratic costs and the absence of state and control constraints, both of which result from the use of the LQR framework, are the main limitations of our model. For example, they prevent the implementation of constant costs that only occur outside the target box as well as some limitation of the applied force or of the admissible mouse cursor positions of a user. Moreover, in contrast to MinJerk, for example, it cannot be guaranteed that the target is actually reached, since the only requirement for the solution trajectory (besides the system dynamics and the initial conditions) is that it minimizes the total cost function, which also includes other objectives in addition to endpoint accuracy. According to our experience, however, the target has always been reached. In addition, the fact that our objective function is directly derived from the task instructions suggests that our model trajectories do not only outperform those of 2OL-Eq and MinJerk in terms of SSE and maximum error, but they are also qualitatively meaningful as they fit to the original task.

We also compared all three models with respect to their approximation of mean trajectories. Although the results slightly differed from those obtained with trial trajectories in some places, the basic effects were the same. We therefore conclude that mean trajectories seem to be a promising tool to get a first insight into the possibilities and limitations of a model, especially in case of very high costs for the approximation of each single trial trajectory (e.g., due to a very large experimental dataset and/or a time-consuming computation of the simulation trajectories). However, final reliable statements on a model's approximation of human movements should better be made on the basis of unaveraged user trajectories.

Furthermore, we had a closer look at the phenomenon of reaction times. As we have shown, our 2OL-LQR$_2$ model is not able to produce an initial phase during which the mouse cursor essentially remains at the initial position. For this reason, we presented the 2OL-LQR$_3$

variant of our model, which is capable of doing that. Nonetheless, we recommend to use 2OL-LQR$_2$ first, particularly if there are no clear reaction times in the user trajectories, because 2OL-LQR$_3$ is in danger of always obtaining better approximations by arbitrarily adjusting the additional parameter n_δ, whereby all parameters lose their meaningful interpretation.

However, there are still some outstanding issues:
An interpretation of the observed impact of the task ID on the parameters as well as a basic understanding of the effects of these parameters on the optimal trajectory (which is anything but trivial due to the unclear dependencies between the parameters and the optimal control) is still pending. In addition, we suspect that despite the small number and easy interpretability of the parameters, there is still some correlation between them. A principal component analysis (PCA) could provide decisive insights in this respect.
Moreover, it would be of interest to apply our algorithm to user trajectories that do not originate from a reciprocal pointing task but from a task in which the targets are completely unknown to the users in advance (maybe also in 2D or 3D and/or with some via-points included). Regarding the characterization of users by parameter sets, an experiment with considerably more participants could also provide additional insights.
Furthermore, the deterministic approach of our model did not allow for a meaningful consideration of visual or proprioceptive perception. The next logical step would be to extend our model to include stochastic error terms and to distinguish between observed output and actual state of the system, analogously to Todorov's extended LQG model. This would also weaken our model assumption of completely optimal user behavior, which does not exactly correspond to reality.
Thanks to our extremely extensible framework, however, countless other modifications are conceivable as well, e.g., using higher-order system dynamics, replacing the jerk costs by some other effort-related cost terms, or penalizing the movement duration independent of the target accuracy.

In summary, we strongly believe that the LQR framework, which has received far too little attention in the past, represents a promising approach for the research on human-computer interactions. We also hope that with our application to pointing tasks we can contribute a small part to a better understanding of these interactions – which, possibly, in the future will help to construct more intuitive interfaces between human and machine, of which we cannot even imagine today whether "anyone would want them in their home".

Bibliography

[1] Z. Artstein. Discrete and continuous bang-bang and facial spaces or: Look for the extreme points. *SIAM Review*, 22(2):172–185, 1980.

[2] K. J. Åström. *Introduction to Stochastic Control Theory.* Dover Books on Electrical Engineering. Dover Publications, 2012.

[3] Y. Bard. *Nonlinear Parameter Estimation.* Academic Press, New York, NY, 1974.

[4] S. Baron and D. L. Kleinman. *Manned Vehicle Systems Analysis by Means of Modern Control Theory.* Number Bd. 1753 in NASA contractor report. National Aeronautics and Space Administration, 1971.

[5] S. Barthelemy and P. Boulinguez. Manual reaction time asymmetries in human subjects: the role of movement planning and attention. *Neuroscience Letters*, 315(1):41–44, 2001.

[6] Å. Björck. *Numerical Methods for Least Squares Problems.* Society for Industrial and Applied Mathematics, 1996.

[7] A. E. Bryson and Y.-C. Ho. *Applied optimal control: optimization, estimation, and control.* Hemisphere Pub. Corp.; distributed by Halsted Press Washington: New York, rev. printing. edition, 1975.

[8] D. Bullock and S. Grossberg. Neural dynamics of planned arm movements: Emergent invariants and speed-accuracy properties during trajectory formation. In *Neural Networks and Natural Intelligence*, pages 553–622. Massachusetts Institute of Technology, Cambridge, MA, USA, 1988.

[9] J. C. Butcher. *Numerical Methods for Ordinary Differential Equations.* Wiley, 2016.

[10] V. C. K. Cheung, A. d'Avella, M. C. Tresch, and E. Bizzi. Central and sensory contributions to the activation and organization of muscle synergies during natural motor behaviors. *Journal of Neuroscience*, 25(27):6419–6434, 2005.

[11] E. M. Connelly. A control model: Interpretation of fitts' law. In *Twentieth Annual Conference on Manual Control June 12-14, 1984 Ames Research*, pages 621–642, 1984.

[12] R. Costello. The surge model of the well-trained human operator in simple manual control. *IEEE Transactions on Man-Machine Systems*, 9(1):2–9, 1968.

[13] R. Courant and D. Hilbert. *Methods of Mathematical Physics, Vol. I.* Interscience, 1st english edition, 1953.

[14] E. R. F. W. Crossman and P. J. Goodeve. Feedback control of hand-movement and fitts' law. *The Quarterly Journal of Experimental Psychology*, 35(2):251–278, 1983.

[15] Datareportal, Hootsuite, and We Are Social. Digital 2019: Global digital overview. `https://datareportal.com/reports/digital-2019-global-digital-overview` (accessed: 2019-07-10).

[16] J. Diedrichsen, R. Shadmehr, and R. B. Ivry. The coordination of movement: optimal feedback control and beyond. *Trends in Cognitive Sciences*, 14(1):31–39, 2010.

[17] J. J. DiStefano, A. R. Stubberud, and I. J. Williams. *Schaum's outline of theory and problems of feedback and control systems: continuous (analog) and discrete (digital).* Schaum's outline. McGraw-Hill, New York, NY, 2nd edition, 1995.

[18] Y. Dodge. *Kolmogorov–Smirnov Test*, pages 283–287. Springer New York, New York, NY, 2008.

[19] F. C. Donders. On speed of mental processes. *Acta psychologica*, 30:412–31, 02 1969.

[20] D. Elliott, W. Helsen, and R. Chua. A century later: Woodworth's (1899) two-component model of goal-directed aiming. *Psychological bulletin*, 127:342–357, 06 2001.

[21] E. Fernández-Cara and E. Zuazua. Control theory: History, mathematical achievements and perspectives. *Boletín de la Sociedad Española de Matemática Aplicada, 26, 79-140.*, 2003.

[22] P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47(6):381–391, 1954.

[23] T. Flash and N. Hogan. The coordination of arm movements: An experimentally confirmed mathematical model. *Journal of neuroscience*, 5:1688–1703, 1985.

[24] M. Friedman. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, 32(200):675–701, 1937.

[25] K.-C. Gan and E. R. Hoffmann. Geometrical conditions for ballistic and visually controlled movements. *Ergonomics*, 31(5):829–839, 1988. PMID: 3402428.

[26] J. Gori, O. Rioul, and Y. Guiard. To miss is human: Information-theoretic rationale for target misses in fitts' law. In *CHI 2017 Conference on Human Factors in Computing Systems*, pages 260–264, 05 2017.

[27] J. Gori, O. Rioul, and Y. Guiard. Speed-accuracy tradeoff: A formal information-theoretic transmission scheme (fitts). *ACM Transactions on Computer-Human Interaction*, 25:1–33, 09 2018.

[28] L. J. Grady and J. R. Polimeni. *Discrete Calculus: Applied Analysis on Graphs for Computational Science*. Springer London, 2010.

[29] R. Granit. *Receptors and sensory perception*. Yale University Press, 1955.

[30] L. Grüne. *Mathematische Kontrolltheorie (Vorlesungsskript)*. Fakultät für Mathematik und Physik, Universität Bayreuth, 08 2018.

[31] B. Hoff. A model of duration in normal and perturbed reaching movement. *Biological Cybernetics*, 71(6):481–488, 10 1994.

[32] B. Hoff and M. A. Arbib. Models of trajectory formation and temporal interaction of reach and grasp. *Journal of Motor Behavior*, 25(3):175–192, 1993. PMID: 12581988.

[33] S. W. Keele. Movement control in skilled motor performance. *Psychological bulletin*, 70(6p1):387, 1968.

[34] W. Ki Na and B. Gou. Feedback-linearization-based nonlinear control for pem fuel cells. *Energy Conversion, IEEE Transactions on*, 23:179–190, 04 2008.

[35] D. L. Kleinman. Optimal control of linear systems with time-delay and observation noise. *IEEE Transactions on Automatic Control*, 14(5):524–527, 10 1969.

[36] D. L. Kleinman, S. Baron, and W. H. Levison. A control theoretic approach to manned-vehicle systems analysis. *IEEE Transactions on Automatic Control*, 16(6):824–832, 12 1971.

[37] T. O. Kvålseth. An alternative to fitts' law. *Bulletin of the Psychonomic Society*, 16(5):371–373, 11 1980.

[38] H. Kwakernaak and R. Sivan. *Linear Optimal Control Systems*. John Wiley & Sons, Inc., New York, NY, USA, 1972.

[39] C. L. MacKenzie and T. Iberall. *The Grasping Hand*. Advances in Psychology. Elsevier Science, 1994.

[40] I. S. MacKenzie. Fitts' law as a research and design tool in human-computer interaction. *Human-Computer Interaction*, 7(1):91–139, 1992.

[41] J. C. Maxwell. I. on governors. *Proceedings of the Royal Society of London*, 16:270–283, 1868.

[42] D. T. McRuer and H. R. Jex. A review of quasi-linear pilot models. *IEEE Transactions on Human Factors in Electronics*, HFE-8(3):231–249, 09 1967.

[43] D. T. McRuer and E. S. Krendel. *Mathematical Models of Human Pilot Behavior*. Number Nr. 188-196 in Mathematical Models of Human Pilot Behavior. North Atlantic Treaty Organization, Advisory Group for Aerospace Research and Development, 1974.

[44] D. T. McRuer, W. Reisener, E. S. Krendel, and D. Graham. *Human Pilot Dynamics in Compensatory Systems: Theory, Models, and Experiments with Controlled Element and Forcing Function Variations*. British Library Lending Division, 07 1965.

[45] D. E. Meyer, R. A. Abrams, S. Kornblum, C. E. Wright, and J. E. Keith Smith. Optimality in human motor performance: Ideal control of rapid aimed movements. *Psychological review*, 95:340–370, 08 1988.

[46] D. E. Meyer, J. E. Keith Smith, S. Kornblum, R. A. Abrams, and C. E. Wright. Speed-accuracy tradeoffs in aimed movements: Toward a theory of rapid voluntary action. *Attention and Performance XIII*, 01 1990.

[47] R. G. Miller. *Simultaneous Statistical Inference*. Springer-Verlag,, New York, 2nd edition, 1981.

[48] P. Morasso. Spatial control of arm movements. *Experimental brain research*, 42(2):223–227, 1981.

[49] D. Mottet and R. J. Bootsma. The dynamics of goal-directed rhythmical aiming. *Biological Cybernetics*, 80(4):235–245, 04 1999.

[50] J. Müller, A. Oulasvirta, and R. Murray-Smith. Control theoretic models of pointing. *ACM Trans. Comput.-Hum. Interact.*, 24(4):27:1–27:36, 08 2017.

[51] A. D. Nordin, W. Rymer, A. A. Biewener, A. Schwartz, D. Chen, and B. F. Horak. Biomechanics and neural control of movement, 20 years later: What have we learned and what has changed? *Journal of NeuroEngineering and Rehabilitation*, 14, 12 2017.

[52] M. Omrani, M. T. Kaufman, N. G. Hatsopoulos, and P. D. Cheney. Perspectives on classical controversies about the motor cortex. *Journal of Neurophysiology*, 118(3):1828–1848, 2017. PMID: 28615340.

[53] G. O'Shea and T. R. Bashore, Jr. The vital role of the american journal of psychology in the early and continuing history of mental chronometry. *The American Journal of Psychology*, 125(4):435–448, 2012.

[54] G. Padfield and B. Lawrence. The birth of flight control: An engineering analysis of the wright brothers' 1902 glider. *The Aeronautical Journal*, 107:697–718, 12 2003.

[55] R. Plamondon. A kinematic theory of rapid human movements: Part iii. kinetic outcomes. *Biological Cybernetics*, 78(2):133–145, 02 1998.

[56] R. Plamondon and A. Alimi. Speed/accuracy trade-offs in target-directed movements. *The Behavioral and brain sciences*, 20:279–303; discussion 303, 07 1997.

[57] N. Qian, Y. Jiang, Z.-P. Jiang, and P. Mazzoni. Movement duration, fitts's law, and an infinite-horizon optimal feedback control model for biological motor systems. *Neural Computation*, 25:697–724, 2013.

[58] D. Rey and M. Neuhäuser. Wilcoxon-signed-rank test. In M. Lovric, editor, *International Encyclopedia of Statistical Science*, pages 1658–1659, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[59] M. J. E. Richardson and T. Flash. Comparing smooth arm movements with the two-thirds power law and the related segmented-control hypothesis. *Journal of Neuroscience*, 22(18):8201–8211, 2002.

[60] J. S. Rustagi. Chapter ii. classical variational methods. In *Variational Methods in Statistics*, volume 121 of *Mathematics in Science and Engineering*, pages 16–45. Elsevier, 1976.

[61] A. Savitzky and M. J. E. Golay. Smoothing and differentiation of data by simplified least squares procedures. *Analytical chemistry*, 36:1627–1639, 07 1964.

[62] R. A. Schmidt and T. D. Lee. *Motor Control and Learning: A Behavioral Emphasis.* Human Kinetics, 5th edition, 2005.

[63] R. Shadmehr. Control of movements and temporal discounting of reward. *Current Opinion in Neurobiology*, 20(6):726–730, 2010. Motor systems, Neurobiology of behaviour.

[64] R. Shadmehr, J. J. Orban de Xivry, M. Xu-Wilson, and T.-Y. Shih. Temporal discounting of reward and the cost of time in motor control. *Journal of Neuroscience*, 30(31):10507–10516, 2010.

[65] T. Söderström. *Linear Quadratic Gaussian Control*, pages 319–365. Springer London, London, 2002.

[66] E. D. Sontag. *Mathematical Control Theory: Deterministic Finite Dimensional Systems.* Springer-Verlag, Berlin, Heidelberg, 2nd edition, 1998.

[67] E. Todorov. Studies of goal-directed movements. Massachusetts Institute of Technology, 1998.

[68] E. Todorov and M. I. Jordan. Optimal feedback control as a theory of motor coordination. *Nature Neuroscience*, 5:1226–1235, 2002.

[69] A. T. Welford. The measurement of sensory-motor performance : Survey and reappraisal of twelve years' progress. *Ergonomics*, 3(3):189–230, 1960.

[70] A. T. Welford, A. H. Norris, and N. W. Shock. Speed and accuracy of movement and their changes with age. *Acta Psychologica*, 30:3–15, 1969.

[71] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

[72] B. Ziebart, A. Dey, and J. A. Bagnell. Probabilistic pointing target prediction via inverse optimal control. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces - IUI '12*. ACM Press, 2012.

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig angefertigt, nicht anderweitig zu Prüfungszwecken vorgelegt und keine anderen als die angegebenen Hilfsmittel verwendet habe.
Sämtliche wissentlich verwendete Textausschnitte, Zitate oder Inhalte anderer Verfasser wurden ausdrücklich als solche gekennzeichnet.

	, den		
Ort		Datum	Unterschrift