Müller, Jörg; Oulasvirta, Antti; Murray-Smith, Roderick

Control theoretic models of pointing

Published in:
ACM TRANSACTIONS ON COMPUTER-HUMAN INTERACTION

DOI:
10.1145/3121431

Published: 01/08/2017

Document Version
Publisher's PDF, also known as Version of record

Please cite the original version:
Müller, J., Oulasvirta, A., & Murray-Smith, R. (2017). Control theoretic models of pointing. *ACM TRANSACTIONS ON COMPUTER-HUMAN INTERACTION*, *24*(4), [27]. https://doi.org/10.1145/3121431

Control Theoretic Models of Pointing

JÖRG MÜLLER, Aarhus University
ANTTI OULASVIRTA, Aalto University
RODERICK MURRAY-SMITH, University of Glasgow

This article presents an empirical comparison of four models from manual control theory on their ability to model targeting behaviour by human users using a mouse: McRuer's Crossover, Costello's Surge, second-order lag (2OL), and the Bang-bang model. Such dynamic models are generative, estimating not only movement time, but also pointer position, velocity, and acceleration on a moment-to-moment basis. We describe an experimental framework for acquiring pointing actions and automatically fitting the parameters of mathematical models to the empirical data. We present the use of time-series, phase space, and Hooke plot visualisations of the experimental data, to gain insight into human pointing dynamics. We find that the identified control models can generate a range of dynamic behaviours that captures aspects of human pointing behaviour to varying degrees. Conditions with a low index of difficulty (ID) showed poorer fit because their unconstrained nature leads naturally to more behavioural variability. We report on characteristics of human surge behaviour (the initial, ballistic sub-movement) in pointing, as well as differences in a number of controller performance measures, including *overshoot, settling time, peak time,* and *rise time*. We describe trade-offs among the models. We conclude that control theory offers a promising complement to Fitts' law based approaches in HCI, with models providing representations and predictions of human pointing dynamics, which can improve our understanding of pointing and inform design.

CCS Concepts: • **Human-centered computing → HCI theory, concepts and models**;

Additional Key Words and Phrases: Control theory, modelling, pointing, targeting, dynamics, aimed movements, fitts' law

ACM Reference format:
Jörg Müller, Antti Oulasvirta, and Roderick Murray-Smith. 2017. Control Theoretic Models of Pointing. *ACM Trans. Comput.-Hum. Interact.* 24, 4, Article 27 (August 2017), 36 pages.
https://doi.org/10.1145/3121431

1 INTRODUCTION

At the outset of the academic study of interaction within human–machine systems, the ideas of control theory were at the core of the subject, e.g., [12, 13, 57], providing models of human

The work of AO has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement no. 637991). RM-S gratefully acknowledges funding support from the EC Horizon 2020 project MoreGrasp, Nr. H2020-ICT-2014-1 643955.
Authors' addresses: J. Müller, Institute for Computer Science, University of Bayreuth, Universitätsstraße 30, 95447 Bayreuth, Germany; email: Joerg.Mueller@uni-bayreuth.de; A. Oulasvirta, School of Electrical Engineering, Aalto University, Konemiehentie 2, PO Box 11000 00076 Aalto University, Finland; email: antti.oulasvirta@aalto.fi; R. Murray-Smith, School of Computing Science, 18 Lilybank Gardens, University of Glasgow, Glasgow G12 8RZ, Scotland; email: Roderick.Murray-Smith@glasgow.ac.uk.

https://doi.org/10.1145/3121431

performance, tools for analysis of joint human–machine system behaviour and frameworks for coupling humans and machines. However, the concepts of control theory and dynamic systems have played only a small role in contemporary human-computer interaction (HCI) research and practice.

We believe it is time to reconsider the role of control theory in HCI, as a better understanding of how humans continuously interact with a computer provides a foundation for designing systems, which can better share the workload between the two, gives us a more scientific framework for describing such interactions, and could support the design of novel interaction techniques. In general, modelling is needed in HCI to bridge the gap between single, isolated observations and theoretical constructs. Formal, executable models offer a systematic, rigorous, and transferable explanation of interaction. They advance scientific explanations by forcing authors to explain assumptions in a clear and scrutinable form. Modelling facilitates generalisation and application of theories, as they can be tested across a variety of conditions.

As a formalism focussed on dynamic systems, control theory moves beyond predicting the typical aggregate descriptions of interaction used in HCI, such as movement time. Importantly, control theory, with its associated dynamic models, allows the prediction of moment-by-moment changes in states like position, velocity, and acceleration. Control theory is of particular interest in a constructive subject like the design of user interfaces. It offers a rich formalism for understanding *interaction* via its constituents: inputs, outputs, feedback, and system states. It can describe the interaction-level dynamics with implications for task-level behaviour [28]. Control theoretic approaches also focus on the closed-loop view, which highlights the fundamental limitations of taking a stimulus–response approach to the causal relationships in situations where the state is fed back to the input, as is the case in human–computer interaction. It is explicitly intended to provide the tools to allow a designer to create the desired system dynamics. Control theory could thereby become a formalism to inform input methods and interaction techniques, from mouse pointing, to gesture-based interfaces, where the dynamics of gesture unfolding in time are core to the approach, or to novel sensor-based interaction methods, which gather evidence over time, and move us beyond the classical, well-tested desktop GUI, or to systems where the computer system dynamics change over time.

While our motivation for reintegrating control theory in HCI is a longer term one, and the models we will explore are well suited for general control tasks, such as tracking and for disturbance rejection, in this article we chose to test the applicability of classical manual control models to an intensively studied core task in HCI: pointing with a mouse.

Surprisingly, the insights from the field of manual control theory have found very little application in HCI. No executable manual control models have been easily available, and many manual control models stem from aircraft pilot modelling, with fewer examples of direct applications to pointing. We decided that replicating and extending this classic pointing task is an ideal starting point for control-theoretic investigations. It affords a link between existing work in both the manual control literature and HCI research and it is directly relevant to many graphical user interfaces where intent is communicated by using a pointer to select spatially bounded regions from a display.

But why work on yet another model of pointing? If we consider pointing as a case, the prevalent Fitts' law [22] based family of modelling offers no explanation about the causal relationship that links conditions and outcomes in pointing. Crucially, it makes no testable predictions about the *process* of pointing. By contrast, control theory offers a model of interaction that allows designers to include explicit assumptions about the human controller, the input device, and issues like feedback, delays, and noise. In particular, it can produce full position, velocity and acceleration profiles to allow inspection of concepts like *overshoot, settling time, peak time*, and *rise time*. Control theory

is a natural approach to pointing dynamics, because while pointing is traditionally understood as a discrete selection task in HCI, the constituent motor action is continuous.

In the future, control-theoretic modelling could complement Fitts' law based analysis by helping us understand the contributions of physical properties of the input device and the properties of feedback. A successful control-theoretic model would support detailed offline analyses, such as what happens when open-loop or closed-loop control strategies are applied, or what is the behavioural impact of a change to the performance objective? In such offline analysis and design, control theory may allow us to simulate the mouse friction, latency, transfer functions, interface dynamics, and mechanical parts. In online use, adaptive interfaces can make real-time model-based predictions about future behaviour extrapolating from the current state and adapting the system behaviour accordingly. However, much research needs to be completed to realise these potentials.

1.1 Article Objectives and Structure

This article addresses three major objectives:

(1) Examine and compare models from manual control theory for their ability to capture dynamic aspects of pointing behaviour in an HCI task, using an automated model fitting process in Simulink and Matlab, based on empirical user data.
(2) Better understand, visualise, and quantify the dynamic behaviour in the oft-used reciprocal pointing task, with a particular focus on the velocity and acceleration profiles during task performance.
(3) Assess models for model fit, complexity, and assumptions, and then use the models to analyse details of lower-level phenomena within a single pointing action, which are neglected in task performance models like Fitts' law.

Our focus in this article is on one-dimensional pointing movements with a mouse as input device, an oft-studied theme for modelling research in HCI. As in previous work in HCI, we assume spatially bounded targets, and time-minimisation as the goal, and we present methods and results from a comparison of four manual control models, which have not been compared before in an HCI context.

We chose models from manual control theory (in particular pilot modelling) based on the following criteria: (1) popularity, (2) small set of parameters that can easily be identified from collected data, (3) ability to model interaction with a non-linear (computer) system, and (4) no requirement to know the movement time in advance. The *second-order lag (2OL)* is a simple linear model widely used in control theory. *McRuer's Crossover model* [34] is a model intended to model pilot behaviour in continuous tracking tasks. The *Bang-bang model* is a non-linear, maximum-effort model. *Costello's Surge model* [11] is a more complex model intended to model pilot behaviour, which combines the Bang-bang model and McRuer's *Crossover* model. These models differ not only in modelling assumptions and fit with data, but also in terms of plausibility as explanations of human movement.

To compare these models, we conducted an experiment using high-fidelity pointer tracking. It captures pointing dynamics at high accuracy, whereas most existing datasets only capture movement endpoints. Besides high temporal and spatial granularity, this data can better expose variability in pointing movements.

In the rest of the article, we first review the literature of aimed movement modelling to contextualise control theoretic models. We then provide an introduction to key control theory concepts and their relationship with HCI. We describe the acquisition of the Pointing Dynamics Dataset and Models, and explain our approach to model fitting.

2 MODELS OF AIMED MOVEMENTS

Manual control theory [11, 34] seeks to model the interaction of humans with machines, for example aircraft pilots, or car drivers. In contrast, *motor control theory*, e.g., [50], seeks to understand how the human central nervous system controls the body, for example in reaching tasks. In this article because we are focussing on unaugmented pointing, we essentially have a 'unit gain' for the controlled system (it could be argued that the mouse mass is almost negligible), so one might reasonably question whether this is really *manual control*, which focusses on human control of a dynamic system, or *motor control* which relates to control of a human's own body? We argue that while this article does not investigate control of target interface elements with complex dynamics, our goal is to develop an engineering framework, which is well-suited to such closed-loop analysis.

In summary, here we do not seek to compare models from manual control theory with motor control models in general, but we compare four models from classical manual control theory against each other, focussing on their ability to model human pointing with a mouse. However, in order to position manual control models relative to motor control models of aimed movements, we provide a brief review of the main types of pointing models, focussing on models of spatially constrained time-minimisation tasks; that is, where the task is to hit a spatially determined target as quickly as possible.

Various authors have proposed simple feedback control models compatible with Fitts' law (e.g., [5, 10, 28]). Winston and Nelson [38] also provide a good initial coverage of the early literature. While there are too many models to review thoroughly, it suffices here to look at them in terms of the phenomena they cover in pointing. From this angle, they can be divided in two classes: (1) task performance-level models and (2) dynamics-level models. Task performance-level models essentially map task demands into indicators of task-level performance, such as total movement time and accuracy. While Fitts' law makes no assumption about the constituent process of pointing, later extensions cover, for example, the number of sub-movements. The dynamics-level models include production of movement in a given task. An overview of models is given in Table 1.

2.1 Models of Task Performance

Performance-level models predict total time and/or variance in end-point error in aimed movements. The best known model, *Fitts' law*, captures a statistical relationship that links performance to task demands [22]. More precisely, it describes movement time (MT) as a function of movement distance (D) and the size of the target (W):

$$MT = a + b\,\mathrm{ID} = a + b\log_2\left(\frac{D}{W} + 1\right), \tag{1}$$

in the Shannon formulation [33], where ID is the index of difficulty of the task. In Fitts' law, movement time is related to the inverse of spatial error. Several variants have been proposed (e.g., [27, 60, 62]). However, most have little to say about what happens during movement.

An important exception is the *iterative corrections model* of [14]. They model motion as a series of ballistic, open-loop sub-movements of equal duration. Detailed kinematic recordings, however, showed only one or at most two corrections. Moreover, considerable variation has been found in the duration of the initial sub-movement. An extension of the idea was proposed by [36]. The *stochastic optimised sub-movement model* defines MT as a function of not only D and W, but also the number of sub-movements:

$$MT = a + b\left(\frac{D}{W}\right)^{1/n}, \tag{2}$$

where n is an upper limit on sub-movements ($n = 2.6$ minimised the RMS error empirically). Several extensions have been proposed to compute also end-point variability [35].

Table 1. Comparison of Selected Pointing Models

Model	Outputs	Predictors	Free Parameters	Assumptions
Fitts' law [22]	MT	D, W	a, b	Limited capacity motor system
Meyer's law [36]	MT	D, W	a, b, (n)	Sub-movements stochastically optimised
Plamondon KT [39]	$x(t)$, $v(t)$, $a(t)$	$\mathcal{D} \times 2$, t_0	$\mu \times 2$, $\sigma \times 2$	Agonistic and antagonistic muscle synergies
Minimum-Jerk [23]	$x(t)$, $v(t)$, $a(t)$	D, MT		Jerk minimisation
VITE [4]	$x(t)$, $v(t)$, $a(t)$	D	$G(t)$, α	Neural network
E-LQG [53]	$x(t)$, $v(t)$, $a(t)$	Q_i, R	A, B, C_i, Σ_1, H, Ω^ω	Optimal control
(This article)	$x(t)$, $v(t)$, $a(t)$	D	$(T_I, T_L, T_n, \tau_e, K_p)$ or (m, d, k, τ) or (g, u, τ)	Error minimisation

Notes: MT = movement time, D = target distance, W = target width, n = number of sub-movements, $x(t)$ = position at time t, $v(t)$ = velocity at time t, $a(t)$ = acceleration at time t, KT: $\mathcal{D}$ = amplitude of impulse command, μ and σ are the logtime delay and the log-response time of the agonist and antagonist neuromuscular system involved in a synergy; VITE: $G(t)$ = GO signal, α = rate, E-LQG: A, B, H = dynamics of controlled system, C_i = signal dependent noise, Σ_1 = initial state covariance, Ω^ω = sensory noise covariance, Q_i = state cost, can depend on D and MT, R = control cost.

These two models have some relationship to control theoretic models of pointing. They assume intermittent feedback control, where each action is based on the error at the start of the action. Meyer also assumes that the neuromotor system is noisy, and that this noise increases with the velocity of the sub-movements. This causes the primary sub-movement to either undershoot or overshoot the target. Although noise per se is not modelled, its consequences are. Critiques of the Meyer model have pointed out trouble capturing the case where sub-movements have different speed–accuracy trade-off properties [39, 40]. It cannot explain why targets with small and large ID values do not conform to the model. Moreover, external factors like gain function and delay may be hard to include. A more critical shortcoming is that the number of sub-movements is fixed. For a given D and W, the sequence of sub-movements would always be the same, and it is not possible to explain why the target is missed at times. Gawthrop et al. [25] demonstrate that a simple predictive controller can be consistent with Fitts' law, while non-predictive controllers cannot. The same paper also presents the link between *intermittent control*, predictive control and human motor control, and further develops this in [24].

2.2 Dynamic Models

Dynamic models include fully time-varying phenomena during pointing. These models can be implemented algorithmically.

Plamondon's Kinematic Theory (KT) assumes that the motor system produces movement by controlling muscles acting in groups, or synergies [39, 40]. KT assumes two systems: agonist and antagonist. These are linear time-invariant systems that produce a velocity output from an impulse command. The velocity of the end-effector of the synergy is given by

$$v(t) = v_1(t) - v_2(t) = \mathcal{D}_1 H_1(t - t_0) - \mathcal{D}_2 H_2(t - t_0), \tag{3}$$

where $\mathcal{D}$ is the amplitude of the impulse command, subscripts 1 and 2 represent the agonist and the antagonist systems respectively, and $H(t - t_0)$ is the impulse response of each system. The response is assumed to converge toward a log-normal curve in most conditions, and becomes a weighted difference of two log-normals, or what is called as the Delta-lognormal Law:

$$v(t) = \mathcal{D}_1\Lambda\left(t; t_0, \mu_1, \sigma_1^2\right) - \mathcal{D}_2\Lambda\left(t; t_0, \mu_2, \sigma_2^2\right), \tag{4}$$

where μ characterises the log time delay (a scale parameter) and σ the log response time (a shape parameter) of the neuromuscular system. The model can be used to predict speed–accuracy trade-off and velocity profiles. It also covers observed phenomena, such as including triple peak velocity profiles and the increase of velocity with increasing distance.

The *Minimum-Jerk Model* [23] analyses hand movements that are smooth and graceful, such as when drawing or gesturing. The motor system is assumed to maximise the smoothness of movements (or, equivalently, minimise jerk). The cost function CF is proportional to the mean square of the jerk, or the time derivative of the acceleration:

$$CF = \frac{1}{2}\int_{t_1}^{t_2}\left[\left(\frac{d^3x}{dt^3}\right)^2 + \left(\frac{d^3y}{dt^3}\right)^2\right]dt. \tag{5}$$

For the case of point-to-point movements, the optimal time series can be computed analytically as a fifth-order polynomial ([23], Appendix A). An application to gesture typing was recently presented by [45]. It samples via-point features from empirical movement samples to generate movements with realistic figural shape and dynamics. However, it does not predict absolute movement time. Furthermore, because it assumes optimal motion, corrective movements are not modelled.

Vector Integration To Endpoint (VITE [4]) is a model inspired by neural networks that takes as input a time-dependent GO signal $G(t)$, a rate α, a target T and initial state and produces a time series of movement towards the target. The model is governed by the equations:

$$\frac{dV}{dt} = \alpha(-V + T - P) \qquad \frac{dP}{dt} = G[V]^+. \tag{6}$$

It is thus a second-order model related to the 2OL explored in this article. A major difference is the non-linear GO function $G[V]^+$ that is applied when integrating the state V (velocity) to the state P (position). The function can be chosen such that the model has a symmetric velocity profile. This is a major difference to linear second order systems, in which the acceleration is necessarily proportional to the error. Any linear system will thus have an asymmetric velocity profile.

Todorov presents a linear optimal control model of motor control based on an extension of the Linear Quadratic Gaussian regulator (E-LQG) [53]. In particular, the additive noise model in the conventional LQG is replaced with a multiplicative, and thus signal dependent, noise term. The model represents the human perceptual system with a Kalman filter and human control as an optimal controller that is controlling a linear biomechanical system. In contrast to the fixed parameter models that we investigate in this article, the control law is determined automatically, while noise terms, controlled system dynamics and cost functions are given as free parameters. The model has been evaluated qualitatively in via-point movements, with parameters adjusted iteratively. It is not obvious how to adjust the model parameters to replicate the behaviour of a specific user in a specific task. Further, the optimal control technique depends on the overall model being linear. Therefore, users interacting with non-linear computer interfaces (e.g., including transfer

functions[1]) can inherently not be modelled. Finally, the model cannot predict the movement time, but rather needs the movement time as an input to predict the time series.

Todorov's model is related to an earlier optimal control model of pilot behaviour, the Kleinman–Baron–Levison model [30]. Also, [63] presents an approach to pointing target prediction that internally uses an inverse optimal control algorithm. This determines a cost function for which the human pointing trajectory would be optimal, and this approach could potentially be integrated with the above mentioned optimal control models.

Shadmehr [51] and Rigoux and Guigon [47] explore the relationship between movement time and optimal trade-offs between reward and effort, an area which seems very relevant to human–computer interaction 'in the wild' where human motivation can be highly variable. This is related to attempts to consider interfaces which allow users to engage in casual or focussed interaction, depending on their context and the difficulty of the task, e.g., [41].

2.3 Summary

Contrary to task-level models, which do not predict position, velocity, nor acceleration, dynamic models address these shortcomings. They can predict pointer time series, velocity, and acceleration profiles, and – like task-level models – are described by a small set of parameters that can be identified from collected user data. Unlike minimum-jerk models or optimal control models, manual control models can predict the time it takes for the pointer to settle on the target, as a function of distance and target size. This is possible, because the behavioural goal is directly implemented in a control law, which makes decisions about the trajectory on a moment-to-moment basis.

3 CONTROL THEORETIC VIEW OF CONTINUOUS INTERACTION: BASICS

This section outlines basic terminology and concepts for understanding pointing as a special case of continuous interaction from a control theoretic perspective (for introductory books, see [28, 29, 43, 52, 56]). We start from the elements of the interaction loop, review the control-theoretic correspondents, and put them together in a general framework. The goal here is not to propose a concrete, executable model but to outline the shared aspects of all control theoretic models of pointing. We call it a 'framework' because it is not an executable model like the ones we investigate in later sections for pointing, but it serves as a reference when we compare the executable models. The framework is shown in Figure 1.

3.1 Basic Terminology

Control theory uses the concepts of signals, systems, goals, states, error (distance of state from goal), feedback, and feedforward. The *block diagram* is one standard way of presenting their relationship in a particular model, describing how *signals* get processed by *systems* in the presence of *feedback*. *Signals* are variables that change their value over time. In Figure 1, signals are represented by arrows. *Systems*, represented by boxes, convert input signals to output signals. Systems can have *states*. In particular in pointing, we propose that the state should not only be the position, but should also include higher derivatives such as the instantaneous velocity. Acceleration could be another possible state, being correlated to muscle activation. In this article, we limit ourselves to second-order (position and velocity) models. *Feedback* occurs when the output of a series of systems is fed back into the same system. This can result in mathematically complex behaviour, including nulling of errors, instability, and oscillations. The term *feedforward* is used in a range of ways in HCI, but in control theory it has a specific meaning. In Figure 1, it denotes the open-loop

[1]Note that this usage is of *transfer function* in mouse OS sense of a non-linear function mapping, rather than a dynamic system transfer function.

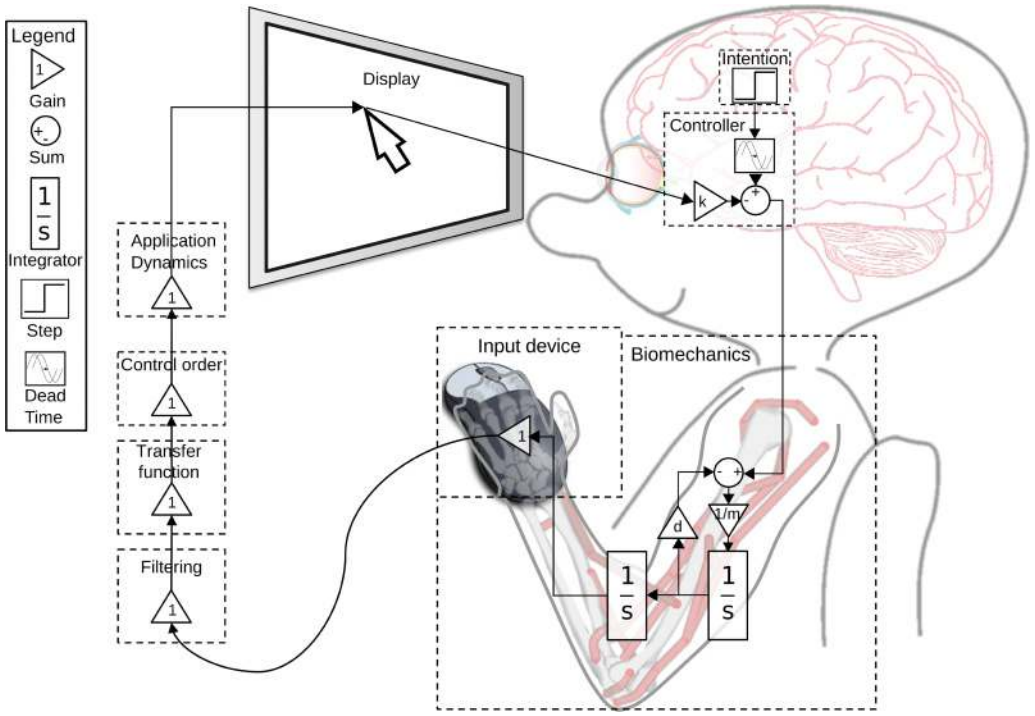

Fig. 1. General control theoretic framework for continuous human-computer interaction. The framework is instantiated as an example with the block diagram of the 2OL model investigated in this article. Interaction is modelled as a continuous control loop between human and computer. The human forms an intention and observes the state of the environment (e.g., pointer position). A controller then calculates the difference between the intention and the state of the environment (error), and calculates a control signal for the biomechanical system from that. The biomechanical system creates forces and motion, which are sensed by input devices. This signal is filtered, and a transfer function (e.g., pointer acceleration function) is applied. The resulting signal can be directly processed for position control, or integrated once for rate control or twice for acceleration control. Finally, application dynamics can range from trivial for a Desktop to complex for a computer game. The state of the application is then fed back to the user via a display. The different components in the control loop can be modelled in many different ways. The other models studied in this article (McRuer, Bang-bang, Surge) instantiate the different components in different ways. Image of the eye is adapted from artwork by Fischer (2013). Image of silhouette inspired by figure in Card et al. (1983).

behaviour of the human when ignoring the feedback, i.e., the path from intention to movement independent of data from perception.

3.2 General Framework for Continuous Interaction

We present a general control theoretic framework for continuous interaction in Figure 1.[2] Pointing is later treated as a special case of this. The signals include *computer input*, e.g., the movement of the mouse on a surface, and *computer output*, the pointer position on the display, as well as signals internal to the human and the computer. As we detail below, the systems of interest are the human and the computer. The human can further be subdivided at least into intention, controller, and biomechanics. These have different roles in controlling pointer position.

[2]See [15, 20] for early HCI discussion of continuous interaction.

On the other hand, the computer can be subdivided at least into input device (e.g., mouse), filtering, transfer function, control order (e.g., position or rate control), application dynamics, and display. Because the output of the computer is fed back into the perception of the human, a feedback-loop emerges. In this article, we will assume the presence of *negative* feedback; that is, the human is calculating the difference of the pointer position from his intention, and is trying to minimise the resulting error.

In the following, we define some terminology for describing interaction from a control theoretic perspective. Based on Figure 1, we define the aspect of the interface that the user is interested to control as the *state* of the interface. This can, for example, be the pointer position and velocity or zoom level and velocity. We define the difference between the user's intention and the state as the *error* signal. In Figure 1, the error would be calculated in the controller.

We can now provide a definition of continuous interaction: *If we can identify a continuous signal loop between user and interface, as depicted in Figure 1, and the user is continuously trying to minimise the error between his intention and the system state, we can say that the user is continuously interacting* with the system.

This definition is independent from the specific interface. For example, the system state could be displayed in a visual, auditory, or haptic modality. Input from the user could be through mouse, joystick, eyes, or mid-air gestures. In many cases there will be a de-clutching mechanism (e.g., letting go the mouse, in the case of pointing) and the interaction will be *intermittently continuous*. This definition also includes continuous interaction with non-computer interfaces, such as driving a car or riding a bicycle.

Many HCI tasks have a direct correspondent in control theoretic terms. For example, the task of pointing to a target that appears at a point in time corresponds to step tracking. Following a moving target might involve ramp tracking. In this formulation, the target over time is represented as an input signal to a control theoretic model.

One relevant aspect is the issue of *disturbances*. In contrast to typical control engineering tasks, in pointing the majority of the disturbances or variability will be typically internal to the human controller, rather than in the system being controlled (the mouse or interaction objects). Games or e.g., flight or driving simulations provide a specific example where disturbances are of the more traditional type, external to the controller, as would be device use in mobile contexts. Further sources of noise or uncertainty come from sensor noise in the pointing device and visual uncertainty due to issues with the display, or the user's vision.

Because of its origins in engineering, the system being the controlled (the computer in our case) in control theoretic terms is often called the *plant*.

3.3 Earlier Examples of Control Theory and Dynamic Models in HCI

Control theory provides an engineering framework, which is well-suited for analysis of closed-loop interactive systems. This can include properties such as feedback delays, sensor noise (see e.g., [54]), or interaction effects like 'sticky mouse' dynamics, isometric joystick dynamics [2], magnification effects, inertia, fisheye lenses, speed-dependent zooming, all of which can be readily represented by dynamic models. The use of state space control methods was explored in document zooming context in [16–19, 31] and [44] reviewed the challenge of optimising scrolling transfer functions and used a robot arm to identify the dynamics of commercial products. Examples of the use of dynamic models in interactive systems are now widespread in commercial systems, and there are also examples in the academic literature, including [8, 59]. Quinn and Zhai [45] use control models to understand how input trajectories associated with words entered into gesture keyboards are likely to vary.

The control perspective can also lead to unusual approaches to interaction, such as interfaces that inferred the user's intent based on detection of control behaviour, as developed in [58] and built on by [21]. It also has the potential to help us understand how to describe interfaces which span the spectrum from casual interaction to highly engaged interaction [41], depending on the effort the user is willing to apply to the task.

We now move on to describe instantiations of this generic framework in models of pointing.

4 POINTING MODELS

We instantiate our framework with particular executable implementations for the case of one-dimensional pointing with the mouse. To this end, we compare specific implementations of sub-models for the human and the computer subsystem of the model.

The task we examine is the translation (of a pointer) along the x dimension. The input device is the mouse. We investigate step tracking with a performance objective of minimising time while avoiding errors. In mouse pointing, users sometimes lift up the mouse and reposition it. Because the tracking of the mouse is broken when the mouse is lifted, in this way the input space can be repositioned in relation to the output space. This process is commonly called 'clutching'. In the instantiations of the framework in this article, clutching is not modelled.

4.1 Submodels for the Human

The manual control literature provides a wide range of control models, based on different assumptions about human behaviour, with a range of computational complexity and number of free parameters. These models model the 'human' part in Figure 1 and have been used for example to model aircraft pilot behaviour. We chose the most widely used models in manual control theory, which can be adjusted to experimental data through a limited set of parameters in a system identification process (see Section 2.3). All models are second order and represent the instantaneous pointer position and velocity, e.g., in two integrators.

A *second order lag* (2OL) can be interpreted as a simple spring–mass–damper system.[3] The model is related to the equilibrium point theory of motor control [50]. In this theory, the human motor system is understood as a spring–mass–damper system, such as the 2OL. When a target changes, the human sets the equilibrium point of the system to the new target and the human body swings towards the target. In order to model human and system delays, we extend the pure second order lag with a dead time. This dead time delays the target signal before it is processed by the spring–mass–damper system. The 2OL can be represented by the differential equation

$$\ddot{y}(t) = \frac{1}{m}(u(t-\tau) - d\dot{y}(t) - ky(t)), \tag{7}$$

where d is the damping factor, k the spring constant, m the mass of the system, τ the dead time, y the pointer position, and u the target signal. Use of the Laplace transform allows this differential equation to be written in the form

$$G(s) = \frac{Y(s)}{U(s)} = \frac{e^{-s\tau}}{ms^2 + ds + k}, \tag{8}$$

which is called the transfer function of the system, commonly denoted $G(s)$. $U(s)$ and $Y(s)$ denote the Laplace transform of the input (e.g., target) and output (e.g., mouse position) signal of the system, respectively. The block diagram of this model is depicted as an example in Figure 1. When

[3]The *system order* usually refers to the highest derivative, or the number of coupled first-order differential equations used in the model. The *control order* refers to the number of integrations between the control input to a controlled system (plant) and the output of a plant.

the target changes, this model quickly accelerates towards the target and decelerates when close to the target. Depending on the damping factor, the model can be underdamped, which means that it overshoots the target and oscillates.

McRuer's Crossover model [34] is a classical model for modelling aircraft pilot behaviour, with a transfer function:

$$G(s) = \frac{K_p e^{-s\tau_e}(1 + sT_L)}{(1 + sT_n)(1 + sT_I)}. \tag{9}$$

Here, T_I is the human lag time constant, T_L is the human lead time constant, T_n is the neuromuscular lag time constant, τ_e is the human dead time, and K_p is the gain constant. Compared to the 2OL, this model is more advanced. In particular it allows to model lead (reaction to the derivative of the error). While this is a good model for tracking a changing reference point, and eliminating the effect of disturbances, a major limitation of this model for representing pointing behaviour is that it is not intended to model reaction to large step changes, like those that characterise a mouse pointing task. In step changes of error, the derivative part of the model creates infinite accelerations, which are clearly not consistent with human behaviour. These infinite accelerations can be damped if the target position is filtered with a low-pass filter before it is passed into the system.

The *Bang-bang model* is a non-linear, maximum-effort, minimum-time model. It maximally accelerates a second-order system (e.g., the human arm) towards the target and then maximally decelerates at the optimal point half-way towards the target. The model is related to the impulse-timing view of motor control [50]. It is also related to Crossman and Goodeve's iterative corrections model. However, the Bang-bang model would not predict sub-movements to be of equal durations, and in its pure form always reaches the target in a single sub-movement. This model cannot be represented as a linear differential equation, and its Laplace transform does not exist. Instead, it is mainly described by the gain parameter g. We extend the pure Bang-bang model with three parameters, to fit human behaviour better. First, we define a dead zone around zero error. Otherwise, in the vicinity of the target, the model will continue to generate short, high-acceleration pulses, without settling. Second, we introduce a *dead time* on the input signal to help compensate for human reaction time delays. Third, we add an *undershoot* parameter. This parameter causes the model not to stop at the target after one bang, but at a point a certain fraction short of the target, like the human.

Costello's Surge model [11] combines the benefits of Bang-bang control and linear control with a switching controller. When the sum of error of pointer location and error velocity is larger than a threshold (such as after a step in the target position) the model enters an open-loop ballistic movement towards the target, as the Bang-bang model does. When the error is smaller than the threshold, the model switches to McRuer's Crossover model. This model is related to Woodworth's model of an initial-impulse phase and later current-control phase [61]. The model is a multiple model (Bang-bang and McRuer) controller, which brings a number of associated complexities. From an implementation perspective, the two controllers need to share the system state. This means that the same integrators need to be controlled by both the Bang-bang and McRuer controller. Further, when the model switches controllers, discontinuities in system behaviour can arise and care needs to be taken with simulation integration techniques. The Bang-bang model we use inside the Surge model is the same as described above, including *dead time* and *undershoot*.

4.2 Submodels for the Computer

Control theory provides tools for modelling key aspects of the computer system, many of which are important for HCI but have been hard to conceptualise. In Figure 1, the input device can be easily modelled as a position control device (such as a mouse) or a rate control device (such as a joystick). Effects like sensor noise, sampling rate, input resolution, frictions, and non-linearities (e.g., limits

of joysticks) can be modelled in a natural way. A particularly neglected aspect in HCI literature are pointing transfer functions[4] (notable exceptions include [6, 7] and [2, 48]), which can be modelled explicitly in this approach. Control theory can also model interfaces with more complex dynamics than the desktop, such as Bumptop [1], physics-based computer games, or interaction effects like 'sticky mouse' dynamics, magnification effects, inertia, fisheye lenses, or speed-dependent zooming, all of which can be readily represented by dynamic models. Finally, aspects of the display such as output resolution can also be modelled. Input device, software, and display introduce latencies, which can be modelled explicitly. While a core strength of control theoretic modelling is that all these aspects of the computer can be modelled and simulated explicitly, in this article, we concentrate on comparing submodels of the human. For this reason, in the remainder of this article the computer is modelled as a trivial constant gain.

5 THE POINTING DYNAMICS DATASET

We now describe the Pointing Dynamics Dataset of one-dimensional serial pointing with the mouse. This dataset and models compared in this article are available from http://joergmueller.info/controlpointing/. Our goals in collecting this data were to (1) cover some commonly studied aspects of pointing in HCI and (2) achieve a level of high fidelity that allows analysis of pointing dynamics.

We captured data using a *serial pointing task* [22], where a user clicks between two one-dimensional targets of varying size and distance.

5.1 Method

5.1.1 Participants. 12 unpaid participants (mean age 28 years, std. dev. 2.5, 11 male, all normal or corrected to normal eyesight, all expert computer users) participated in the study. All preferred to use, and used, the mouse with the right hand, while two were left-handed. They had on average 16.8 (std. dev. 6.1) years, of experience of mouse use, and used a mouse on average for 34.9 (std. dev. 20.6) hours per week.

5.1.2 Task and Materials. Two one-dimensional targets are displayed. The x-dimension of the mouse is used to move a white crosshair pointer (1px wide) that only moves horizontally. The task is to click on the targets serially. The current target is shown in red, while the other target is shown in grey. A new trial starts as soon as the participant clicks on the previous target. Missed trials are annotated in the dataset. Distance and width are varied between blocks, but kept constant within each block. One *condition* of the experiment is a combination of distance and width, where all distances and widths are in screen (not mouse) coordinates. We cover four different indices of difficulty (ID) (2,4,6,8) with two different distances. One distance is 212mm, with target widths of 0.83, 3.32, 14.1, and 70.6mm, respectively. The other distance is 353mm, with target widths of 1.38, 5.54, 23.5, and 118mm, respectively. Each combination is repeated for 102 trials, and the order of conditions is counterbalanced using a Latin square.

5.1.3 Procedure. Participants were asked to adjust table, display, and mouse pad to their preferences. All participants were resting their palm on the mouse pad. Participants were introduced to the task and completed a training phase for all conditions before starting the experiment, where they trained for 22 trials in each condition. Participants took a break after each condition, where they were asked to stretch their limbs and relax briefly. They were asked to adjust the mouse position in the first trial of each condition and avoid clutching in the remaining trials. Participants

[4]Note the difference between a pointing transfer function which refers to a changing C:D ratio, and a transfer function $G(s)$ in the control literature which is a linear, time-invariant system.

Fig. 2. The apparatus used for the experiment.

were asked to click on the targets as fast as possible while maintaining an error rate of below 5%. When participants clicked outside the target, a beep tone sounded, and the next target was presented. Participants were not asked to repeat failed trials. Failed trials do not have a large influence on our system identification process, and are not the focus of this article.

5.1.4 Apparatus. Data was captured on a Dell Precision 7810 PC with an AOC G2460PQU monitor (24in., 1,920 × 1,080 resolution, 140Hz, no Vsync). The Logitech G502 mouse was used, with no additional weights on the G240 mousepad. 12,000 DPI resolution was used with a 1,000Hz update rate. The software used libpointing.org for accessing the raw mouse data via rawHID, instead of using the mouse data that is provided preprocessed by the operating system. A resolution-aware constant control-display (C:D) gain, manually tuned to 4.36, was used. OpenGL (glfw3) was used for low-latency graphics generation. The program was running at approximately 2,000Hz, such that the frequency of the overall apparatus was limited by the 140Hz of the monitor. Data was captured using Microsoft Windows 10. The apparatus is shown in Figure 2.

Latency was measured with a Sony DSC-RX10M2 camera at 1,000 FPS. The mouse was hit with a hard object at high speeds, and the number of frames from impact to pointer motion on the display was counted. The latency was 25ms.

5.2 Preprocessing

We work directly from the raw dataset—no outliers or errors were removed. In order to facilitate further analysis, data was preprocessed. We drop the first 20 trials from each condition as participants familiarised themselves with the new condition. We downsample the data to 500Hz using padding of pointer positions. Naive calculation of derivatives (velocity and acceleration) from position data would greatly increase any small noise in the signal. Therefore, we filter the pointer position using a Savitzky–Golay filter [49] with a 4th degree polynomial and a window size of 101 samples (200ms). We chose this approach to creating an FIR filter, so that it could also calculate 1st (velocity) and 2nd (acceleration) derivatives of pointer position, which were added to the dataset. The effective low-pass bandwidth (to a −3dB cut-off) is 8.79Hz. This focusses the modelling effort on the most important aspects of human control behaviour, and filters out high frequency disturbances, and mouse sensor noise.

5.3 Final Dataset

The entire dataset provides mouse movement data for 12 participants in CSV format. The raw dataset contains timestamp, raw mouse movement (x and y), pointer position on screen (x only), condition, trial, target position, program status (pause, etc.) and performance data (success, trial

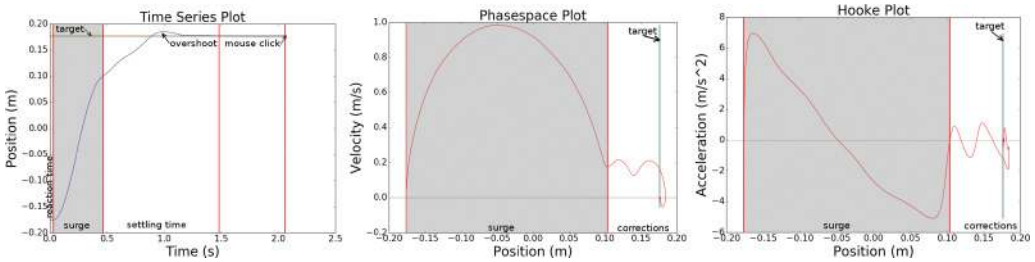

Fig. 3. Core visualisations of pointing dynamics used throughout this article. On the left, a *time series* plot shows how the pointer position changes over time. After an initial reaction time, users move the pointer towards the target in a large, often ballistic, movement (the *surge phase*). There might be some overshooting and gradual corrections towards the target. After the pointer has settled on the target, it rests there for some time until users click the mouse button. In the middle, a *phase space plot* plots pointer velocity over position. The exact shape of the surge phase is clearly depicted in phase space plots. On the right, a *Hooke plot* plots acceleration over position. The shown trial is trial 41 of participant P1 for a target width of 0.83mm, typical for trials with high ID in the dataset.

duration). The processed dataset is sampled at 500 Hz and a smoothed pointer position, velocity, acceleration, target and timestamp are provided. The size of the processed dataset is 217MB. The overall error rate (movement endpoints outside the target) is 2.86%. There are two main sources of error. In some cases, participants accidentally double-clicked on a target. In other cases, they clicked slightly besides the target. In particular, for very small targets (0.83mm = 3px and 1.38mm = 5px), some participants slipped off the target when they clicked the mouse button. In one extreme case, one participant reached an error rate of 11% for 0.83mm targets. The median error of all failed trials over all participants and conditions is 0.82mm. Although participants were instructed to avoid clutching, the dataset includes a few clutch events. During these events, mouse data is missing.

6 MODELLING APPROACH: OBJECTIVES AND IMPLEMENTATION

We evaluate the four models (2OL, McRuer, Bang-bang, and the Surge model) against the collected dataset.

6.1 Visualisations of Pointing Dynamics

A key contribution of control-theoretic models is a deeper understanding of dynamics. To better gain insight into how pointing dynamics differ among the four models, we utilise five plot types from control-theoretic literature:

1. Time series plots: In Figure 3 (left), we can see how the pointer position changes over time. There is an initial reaction time before the pointer starts moving. Then there is a more or less ballistic phase when the pointer moves towards the target, called the *'surge'*. There might then be some corrections, *overshooting*, and oscillations around the target. Finally, after the pointer has *settled* on the target, it rests there for some time until the user finally clicks the mouse button. Time series can be plotted for a single selection, as in Figure 3 (left). In this case they show the *step response* of human-computer system, which is the response to a step in target position. Time series can also be plotted for an entire condition with a number of repeated selections (see Figure 10).

2. Velocity profiles show pointer velocity plotted over time (see Figure 11).

3. Acceleration profiles show pointer acceleration plotted over time (see Figure 14).

4. Phase space plots are an alternative visualisation of pointer velocity, where velocity is plotted against pointer position. They present motion trajectories in the state-space of a second-order system. In Figure 3 (middle), the user accelerates towards the target in one ballistic movement, which lands in the vicinity of the target. Then follows a series of smaller ballistic movements, changing into a more continuous final approach to the target.

5. Hooke plots plot acceleration (y-axis) against pointer position (x-axis) (see Figure 3, right). Guiard [26] gives an introduction to the relationship between pointing behaviours and Hooke's law and the use of Hooke plots. Hooke and phase space plots can provide different insight into dynamics from time-series, and are less susceptible to time-alignment issues.

6.2 Criteria for Model Fitness

To evaluate the applicability of control-theoretic models, it is necessary to assess models against several objectives. Our primary objective is to model the time series behaviour of pointing; that is, the movement of the pointer over time from an initial pointer position towards a target.

To assess model fitness in this respect, we measure the *root mean square error (RMSE)* of predicted pointer position to actual pointer position.[5] The difference between these two positions is squared, and then averaged over the entire duration of the condition. Finally, the root of this value is taken. Calculated like this, we can assess model error, using metres as the unit of measurement, which is natural for pointing tasks. A downside of this metric is sensitivity to timing differences – even if pointing dynamics are predicted with 100% accuracy in terms of position over time, even a slight difference in motion onset time will lead to large RMSE. RMSE can be used to understand error in modelling the dynamics of movements, in other words the difference in time derivatives velocity and acceleration, as well as position. We also compare maximum velocities, accelerations, and decelerations.

In order to compare details of the pointer trajectory, we use *step response* characteristics. The pointing task corresponds to the step response in control theoretic terms, and we can use a variety of metrics to characterise it. *Overshoot* measures by how much the centre of the target is overshot, in percent of the distance from initial position to target centre. *Rise time* measures how long it takes for the pointer to travel from 10% to 90% towards the target. This is a classical metric for steepness of the response. *Settling time* measures how long it takes for the pointer to enter the target such that it does not exit the target before the click. *Peak time* measures how long it takes for the pointer to reach the largest overshoot.

There are many aspects of user behaviour that we are interested in that are difficult to quantify. Therefore, it is also important to judge model behaviour against user behaviour regarding whether the model can reproduce qualitative phenomena. This is done in particular by comparing the time-series, phase space, and Hooke plots qualitatively.

Finally, Ockham's rule suggests that models with fewer parameters are preferable, if they can accurately predict pointer behaviour. However, we prioritise plausibility over the number of parameters.

6.3 Model Implementation

We implement all models in Simulink[6] as block diagrams. Simulink is a Matlab extension that allows simulation of linear and non-linear systems. *The second-order lag* is a linear time-invariant (LTI) system. Matlab allows implementation of LTI systems in state-space form, or as transfer function. These systems can be simulated very efficiently using the lsim command. However, for

[5] See Chapters 3 and 4 of [42] for a detailed discussion on criteria for manual control models.

[6] Simulink, Matlab Release version R2016a, The MathWorks, Inc., Natick, Massachusetts, United States.

consistency with the non-linear systems, we decided to model also the 2OL in Simulink using two integrators, three gains, and a dead time.

McRuer's model is also a LTI system. We implement the differential equation as described in [11], as a block diagram in Simulink. In order to avoid infinite derivatives at step changes of the input signal, we filter the input with a low pass filter with transfer function $G(s) = \frac{1}{0.01s+1}$ (this has a −3dB cut-off at a frequency of 100rad/s = 15.9Hz).

The *Bang-bang model* is a non-linear model. We implement it as a Matlab function within a Simulink model that gets evaluated at every simulation step. The Matlab function creates an acceleration signal that feeds into two integrators. The model receives the target signal after a *dead time*, and calculates the error to the current pointer position. The Matlab function checks whether the error is larger than 5 mm, and if so, generates an acceleration–deceleration Bang-bang pulse. The model does not land exactly at the target, but undershoots by a fraction *undershoot*. Until this Bang-bang is finished, the function does not process the error.[7]

Costello's Surge model combines the McRuer and Bang-bang models and is more difficult to implement. The principal difficulty is that when the model switches between the submodels Bang-bang and McRuer, the state, consisting of pointer position and velocity, must be preserved. Thus, the two submodels need to share the two integrators that represent the state. This excludes the use of transfer functions or state-space representations, which maintain state internally. The model is simply a combination of the McRuer and Bang-bang models with a switch that switches control of the integrators to the McRuer model when the Bang-bang model is not active (within ϵ_x of the target position and below ϵ_v absolute speed), and to the Bang-bang model otherwise. Discontinuities in model behaviour can arise at the instance of switch of submodel.

In order to simulate, the models need solvers for the differential equations they represent. The discontinuous character of the switching Costello and Bang-bang models caused problems with variable step solvers. Therefore, we chose a conservative, fixed-step solver (ode5, Dormand–Prince) with a fixed step width of 1ms for all models.

6.4 System Identification Process

The 2OL model has four free parameters (mass m, spring constant k, damping factor d, and dead time τ). McRuer's model has five free parameters ($T_I, T_L, T_n, \tau_e, K_p$). The Bang-bang model has three free parameters, gain g, *undershoot* and dead time τ. Costello's Surge model has an additional model switching threshold, resulting in nine free parameters. These parameters need to be adjusted to fit the experimental data. This process is called *system identification*.

There exists a comprehensive literature on system identification with a wealth of methods specialised for identifying specific classes of systems [32]. For our study we wanted to use the same system identification technique for all compared models. Because Costello's model is a non-linear switching controller, its parameters are difficult to identify using classical system identification techniques. Therefore, we resorted to a very general technique based on simulation and optimisation.

[7]Two practical pitfalls appear with the Bang-bang model at the instant of step changes. The first is that the simulation environment only executes the code at discrete simulation steps. If these steps do not coincide with the instances where the target changes (step), the simulation interpolates the target at the simulation step. Thus, the model might execute a bang with an interpolated (much smaller) error that lands short of the target. The second is that at step changes, the error velocity is infinite. Because the model uses the error velocity to compute the optimum step size, it overshoots the target. As a practical technique to circumvent both problems, we only execute a bang if the derivative of the target position is close to 0, i.e. the target does not move.

In this technique, an optimiser aims to identify the parameter set that minimises the error between the model and experimental data.[8] The objective function determines which aspects of user behaviour the models are fit to. As objective function we use the sum squared error (SSE) of position, velocity and acceleration, weighted equally. We found that for the linear models, 2OL, and McRuer, regardless of the choice of optimiser (search-based, gradient descent), the process reliably converges to the same parameter set. For these models, we used the Simplex search method as implemented in the Matlab fminsearch function, because of its efficiency. However, for Costello's model, many methods are susceptible to becoming stuck in local optima, resulting in a large variety of parameters for different optimisation methods. The Pattern search method showed to most reliably converge on the best parameter set. Therefore, we resorted to using a Pattern search using a Latin hypercube search method for the Bang-bang and Surge models. We found the initial values to have negligible impact on the result. Therefore, we ran system identification once on a different data set, and set the initial values for the optimisation equal to the average values of the preliminary identification step.

We are initially interested in how well manual control models can represent pointing behaviour of individual participants in individual conditions. We therefore identify model parameters and evaluate model parameters for each set of participant, distance, and width individually. We split the experimental data into a training and a test set. For each condition (combination of distance and width), we use the second half of trials for training, and the first half of trials for evaluation.

6.5 Analysis of the Surge Movement

Undershoot has negligible impact on RMSE, but is an important and interpretable feature of our models, so it is therefore not identified through the optimisation process explained above, but calculated as follows. To identify the surge movement we analyse the acceleration of the pointer. There are different possible definitions of the end of the surge period. We chose to end it when there was a zero crossing of acceleration, after the deceleration phase. From the start of the movement, we find the first point when the deceleration of the pointer is below -1ms^{-2}, and then from that point, we find the next zero crossing of the acceleration and define this as the end of the surge. We validated this heuristic on our data, and it identifies robustly the parts of the movement that constitute the first sub-movement (surge).

7 ANALYSIS OF POINTING DYNAMICS

The participants exhibited different movement strategies, with some participants having a more ballistic, open-loop initial surge and others having a more closed-loop surge with corrections in the deceleration phase.

Figure 4 provides an overview. It shows time series, phase space, and Hooke plots, for all participants for a single condition (W = 1.38 mm, D = 353 mm, ID 8). While there is considerable variability, clear patterns emerge. Our first approach to explore the sources of variability is to plot the data for each participant. Figure 5 shows this for the phase space plot.

We observe two different strategies, exposing a large source of variability.

(1) Some participants have one big surge movement of relatively symmetric form that ends fairly close to the target. This indicates a ballistic surge without much correction once the motion is initiated (open-loop strategy). This is cleaner for the plots on the left.
(2) Other participants accelerate, but then have a flatter coasting trajectory towards the target. This indicates that they are not executing the first surge motion in a purely ballistic

[8]We use Simulink Design Optimisation for computations.

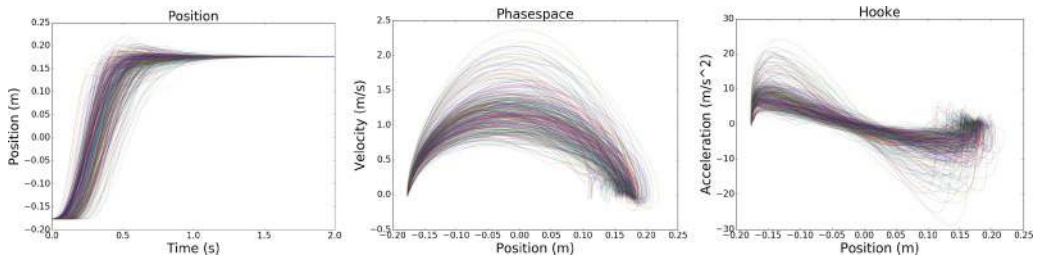

Fig. 4. An overview of dynamics in the task of pointing at 1.38mm targets at 353mm distance (ID 8). Left: Time series; Middle: Phase space plots; Right: Hooke plots. For clarity, failed trials are not shown.

manner but rather correct during the movement to try to land closer to the target (closed-loop strategy).

All participants cover the remaining distance with some smaller (ballistic) movements and finally some more gradual homing in towards the target.

Figure 6 shows the Hooke plots of all participants. All plots have a clear N shape, characteristic for the high ID of the task. We make two observations. First, for some participants the acceleration and deceleration phases are also relatively symmetric, indicating an open-loop strategy. Second, for others, the deceleration is less steep than the acceleration, indicating a closed-loop strategy. Naturally, there is more variability of the endpoint of the surge movement than of the start point, since the target is small.

From Figures 5 and 6, we see that participant P1 had average phase space and Hooke profiles, so we use P1 as an example of an 'average' participant behaviour throughout the article. Figure 7 shows the phase space and Hooke plots of participant P1 for all conditions, sorted by target width. It can be seen that the ID has a strong influence on the shape of the phase space plot. The plots for ID 2 are qualitatively different from higher IDs, showing the oscillating nature of the movement. For ID 2, movement is oscillating between the two targets, and the target is always reached within a single submovement. The deceleration phase of the previous movement continues into the acceleration phase of the next movement, without the acceleration dropping to zero. This results in an oval phase space plot and a straight-line Hooke plot. This phenomenon has been reported by [3, 37]. The higher the ID, the more undershooting occurs, and the higher the number of submovements to reach the target. For higher IDs, the Hooke plot has an N shape, and the individual trials are clearly separated. Overall, there is more undershooting than overshooting of the target. For the same ID, the conditions with different distance and target width have similar phase space and Hooke plots, with differences mainly in scale. Conditions with smaller distance have a lower velocity for the same ID.

7.1 Surge Duration and Reaction Time

Figure 8 shows an analysis of parameters of pointing dynamics. The top left plot shows the points where the participants clicked (endpoints) relative to the centre of the target, grouped by target width. Naturally, with larger target size, the variance of endpoints increases. More interestingly, in the oscillatory case with ID 2, the median endpoint is not close to the target centre. Instead, participants preferred to click short of the target centre, which decreases the distance of their movement. The top right plot shows the reaction times of all participants and all trials grouped by ID. We define 'reaction time' as the time from the previous mouse click until the pointer has moved more than 5mm from that position. Reaction times for ID 2 are slightly lower

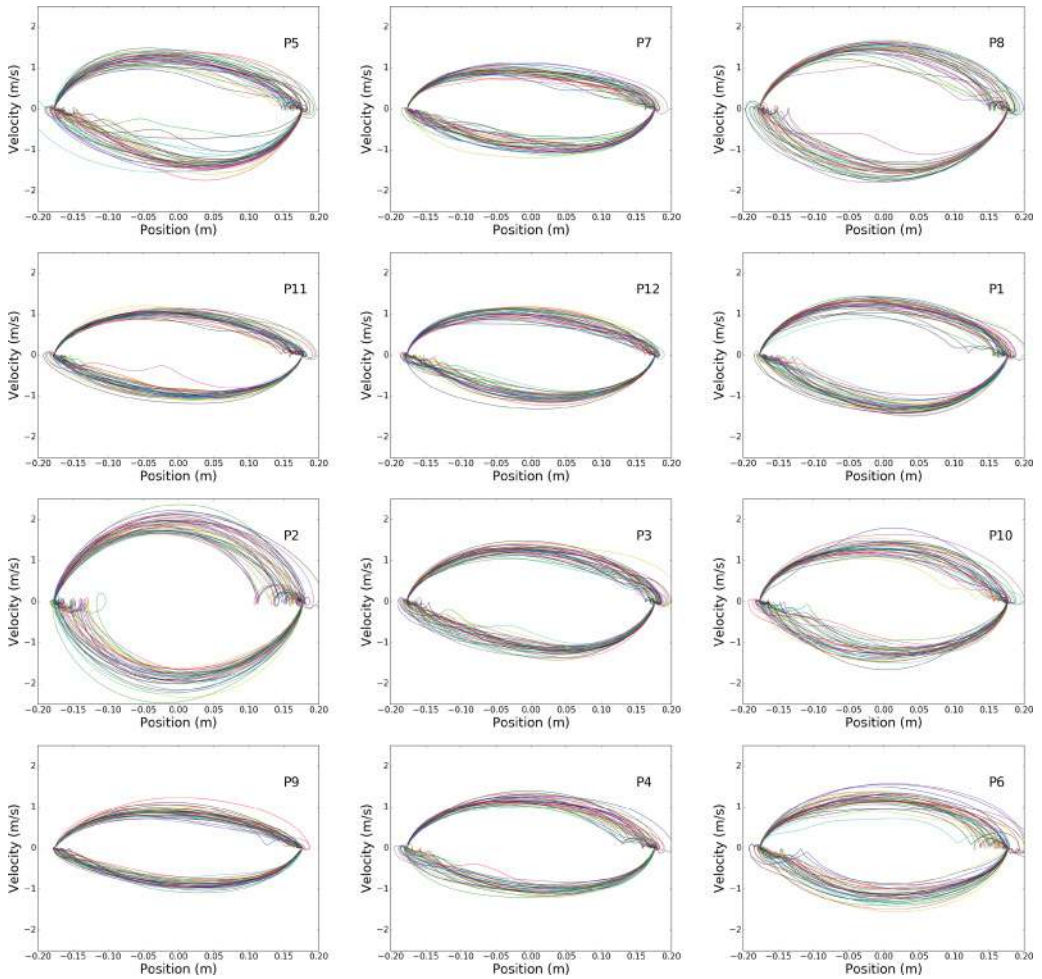

Fig. 5. Phase space plots of all 12 participants for pointing on 1.38mm targets at 353mm distance (ID 8). The plots are sorted by deceleration slope from left to right. Movements on the left are more ballistic, with a relatively symmetric acceleration and deceleration slope. Movements on the right have a less steep deceleration, indicating that the participants are correcting their movement while they are decelerating (c.f. Figure 6). There is considerably more undershooting than overshooting. Finally, the spring–mass properties of the human hand sometimes lead to small 'loopings' when the mouse stops (e.g., P5). For clarity, failed trials are not shown.

(median = 78ms), but for higher IDs reaction times are similar (median between 102 and 108ms). This might be explained by the fact that for ID 2, the acceleration that decelerates the previous movement is directly used to accelerate the next movement. Because acceleration is related to muscle force, the force is already built up at the start of the trial for ID 2. Another possible explanation is that the cognitive overhead of motion planning is lower for ID 2 than for higher IDs.

The bottom left plot shows the proportion of trial time spent in the surge movement grouped by target width. It can be seen that, with increasing target width, the proportion of the trial time that is spent in the surge movement increases. For very small targets (0.83 mm), less than half (median = 34.6%) of the trial time is spent in the surge movement. The majority of the time is spent

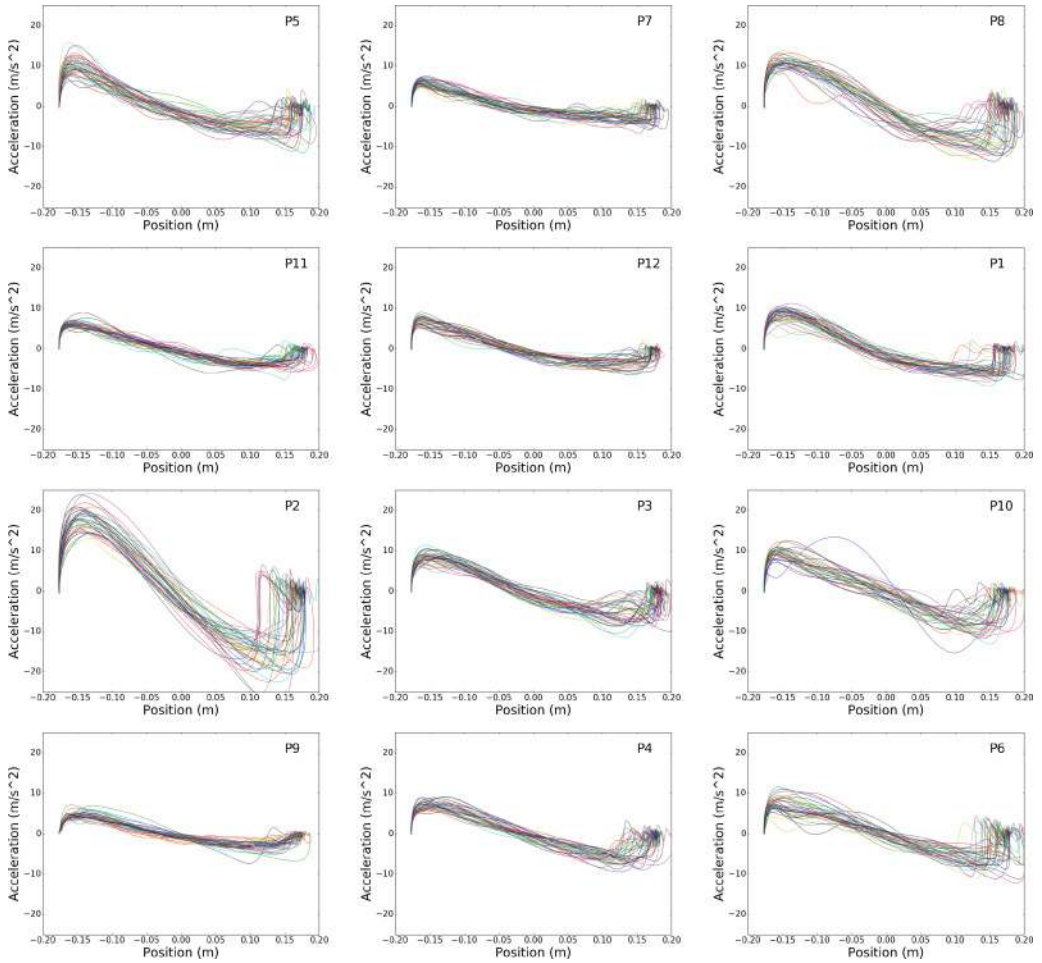

Fig. 6. Hooke plots of all 12 participants for pointing on 1.38mm targets at 353mm distance (ID 8). The order of participants is the same as in Figure 5. For participants on the left, the Hooke plot has a more symmetric N shape, while for those on the right, the deceleration is less steep than the acceleration. Plots on the left show higher accelerations than plots on the right.

in corrections. With very large targets (ID 2), the entire trial time is spent in the surge movement. The bottom right plot shows the proportion of the distance to the target that is covered by the surge movement. With 0.83mm targets, participants undershoot the target considerably, with the surge covering median = 95.5% of the distance. As the target size increases, the surge ends closer to the target centre. At a target width of 23.5mm, the surge covers a median = 99.3% of the distance. With even larger targets, participants stop clicking in the target centre, and the surge undershoots the target centre more. With 118mm targets the surge covers median = 94.2% of the distance. There is a strong trend towards undershooting rather than overshooting the target, with the median surge endpoints undershooting for all target widths.

8 MODELLING RESULTS

This section examines modelling results, comparing 2OL, McRuer, Bang-bang and Surge. The identified model parameters are summarized in Table 2. As shown below, there are large differences

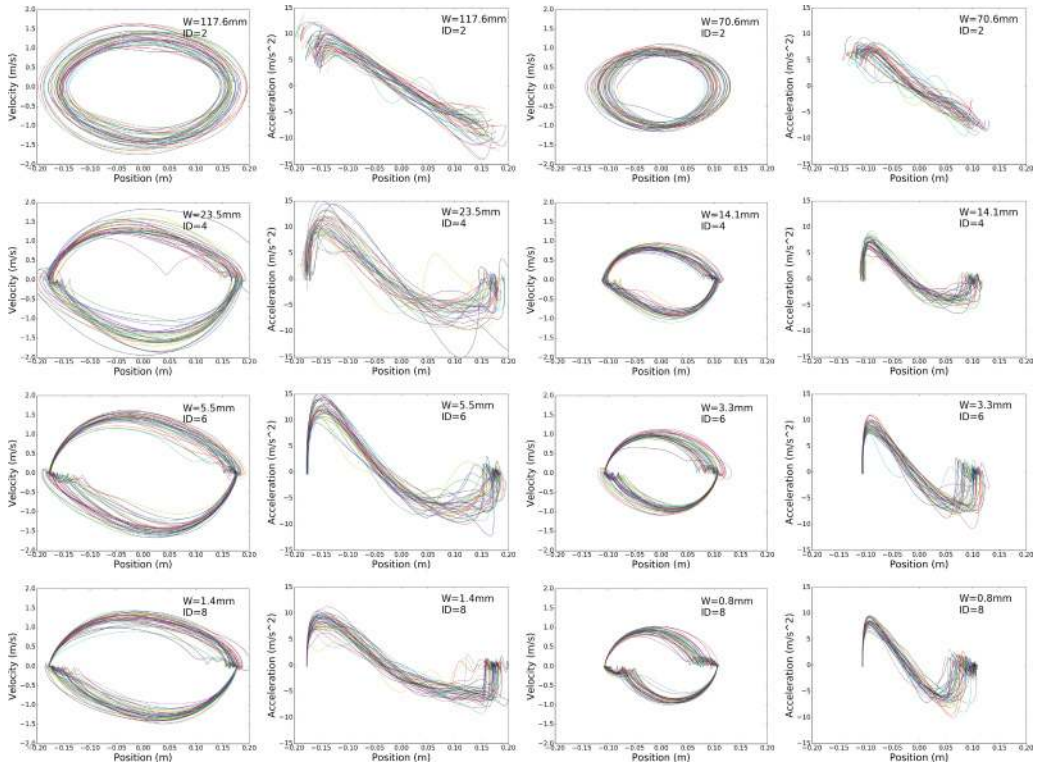

Fig. 7. Phase space and Hooke plots by target width for participant P1. Plots for the same ID look similar, only at different scale. Note how for low IDs, the deceleration continues into the acceleration of the next trial, without the acceleration reaching 0.

in model behaviour for index of difficulty ($ID = \log_2(D/W + 1)$) 2 versus IDs 4, 6, and 8, and in some cases we analysed these separately. To compare model fit, we use RMSE as reported above. We plot RMSE in Figure 9 using 95% confidence intervals for comparison.

8.1 Position

Figure 9 (left column) shows the accuracy of the four models in predicting the position of the pointer over time (top for IDs 4, 6, and 8, and bottom for ID 2). Bang-bang shows best model fit for position and McRuer the worst. Differences are small, however. The median RMSE over all participants and all conditions is 8.8% for 2OL, McRuer 9.1%, Bang-bang 8.4%, and Surge 8.9%. To estimate the variance within the data, we have computed the mean of all trials in the *test set* (mean trial) for each condition. The mean pointer position is calculated relative to time since trial start. The median RMSE of the mean trial is 6.8%.

We also split the RMSE between the dynamic phase (when the participants' pointer is not within the target) and the static phase (when it is within the target). In the dynamic phase, median RMSE of 2OL is 9.9%, McRuer 10.1%, Bang-bang 9.5%, and Surge 10%. In the static phase, median RMSE of 2OL is 4%, McRuer 5.2%, Bang-bang 2.8%, and Surge 3.1%. Figure 10 shows the actual pointer position and predicted pointer position over time for the four models. All models can predict the participant's pointer position well. Because all models have a dead time, they can model the reaction time (the pointer starts moving a small time after the target has switched) of the human,

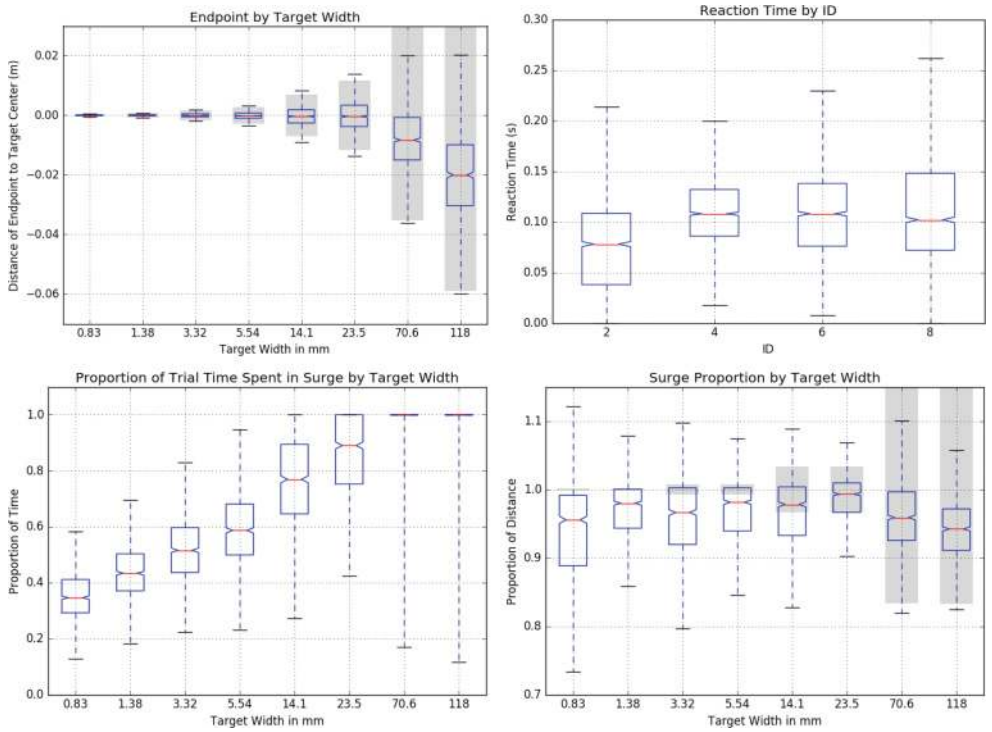

Fig. 8. *Top Left:* Points where the participants clicked relative to the centre of the target, grouped by target width. The targets are shown in grey. *Top Right:* Reaction time by ID. Reaction time measures the time from start of the trial until the pointer has moved more than 5 mm. *Bottom Left:* Proportion of trial time spent in surge by target width. *Bottom Right:* Proportion of distance between targets covered by the initial surge. The targets are shown in grey. See text for interpretation.

although care should be taken when interpreting the identified dead times, as they will also be affected by the inherent imperfection of the model when fitting the experimental data, and will not necessarily represent the human reaction time if jointly optimised with other parameters. Because model parameters are optimised to minimise overall RMSE of position, velocity, and acceleration, model behaviour deviates from human behaviour when close to the target.

8.2 Velocity

All models reproduce human-like velocity profiles to some degree, with systematic differences visible in phase space plots. Figure 9 (middle column) shows the RMSE (in ms^{-1}) of the four models in predicting the velocity of the pointer over time. McRuer shows the best fit for ID 2, but the worst for the other IDs.

Figure 11 shows time series of pointer velocity predicted by the four models. We can see that all models match the participant's velocity profile well. The Bang-bang and Surge models have characteristic triangular velocity spikes. In Figure 12, we see the phase space plots of all models. In the phase space view it becomes more clear that all models have characteristic deviations from human behaviour. Regarding the initial surge, Bang-bang and Surge show a phase space plot closer to a ballistic open-loop strategy, while 2OL and McRuer resemble more a closed-loop correcting strategy, where acceleration is proportional to error.

Table 2. Identified Model Parameters with Mean(Min,Max) for Each ID

Parameter	ID 2	ID 4	ID 6	ID 8
2OL:1/m	25.40(9.10,50.63)	34.73(7.80,78.49)	46.41(12.40,105.19)	42.97(18.50,90.96)
2OL:d	0.18(0.09,0.29)	0.22(0.15,0.37)	0.22(0.13,0.36)	0.23(0.15,0.32)
2OL:k	2.66(1.48,4.14)	1.31(1.13,1.61)	1.12(1.07,1.17)	1.07(1.04,1.10)
2OL:dead time	0.08(0.00,0.17)	0.10(0.08,0.13)	0.09(0.05,0.17)	0.08(0.04,0.14)
MCR:K_p	5.95(2.83,24.86)	6.95(3.25,49.13)	9.09(3.76,26.47)	15.25(4.82,75.40)
MCR:T_I	0.82(0.38,2.53)	1.42(0.53,9.94)	2.20(0.96,7.76)	3.77(1.12,15.29)
MCR:T_L	−0.00(−0.01,0.00)	−0.00(−0.02,0.00)	−0.00(−0.02,0.00)	-0.00(-0.01,0.00)
MCR:T_n	0.23(0.09,0.50)	0.28(0.07,0.58)	0.20(0.06,0.54)	0.18(0.09,0.32)
MCR:τ	0.00(0.00,0.05)	0.00(0.00,0.09)	0.01(0.00,0.07)	0.00(0.00,0.05)
BANG:gain	6.29(2.00,16.00)	2.83(1.00,6.00)	3.17(1.00,6.00)	2.58(1.00,6.00)
BANG:undershoot	0.99(0.98,1.01)	0.98(0.84,1.00)	0.97(0.91,1.00)	0.96(0.83,0.99)
BANG:dead time	0.04(0.02,0.08)	0.01(0.00,0.03)	0.01(0.00,0.05)	0.00(0.00,0.04)
SURGE:K_p	37.85(0.85,184.91)	10.62(0.17,51.40)	12.45(0.21,60.48)	34.02(0.32,189.68)
SURGE:T_I	0.04(0.00,0.05)	0.04(0.01,0.05)	0.03(0.01,0.05)	0.03(0.01,0.05)
SURGE:T_L	3.02(0.06,9.18)	2.95(0.06,9.28)	0.73(0.05,2.81)	0.52(0.00,4.28)
SURGE:T_n	5.61(0.68,9.55)	6.56(1.32,9.94)	6.29(0.94,9.97)	6.02(1.00,9.96)
SURGE:τ	0.21(0.07,0.46)	0.28(0.01,0.50)	0.22(0.00,0.49)	0.13(0.00,0.50)
SURGE:threshold	0.04(0.00,0.05)	0.04(0.01,0.05)	0.03(0.01,0.05)	0.03(0.01,0.05)
SURGE:gain	6.29(2.00,16.00)	3.33(1.00,7.00)	4.04(1.00,12.00)	3.62(1.00,8.00)
SURGE:undershoot	0.99(0.98,1.01)	0.98(0.84,1.00)	0.97(0.91,1.00)	0.96(0.83,0.99)
SURGE:dead time	0.06(0.01,0.12)	0.04(0.00,0.08)	0.03(0.00,0.07)	0.03(0.00,0.09)

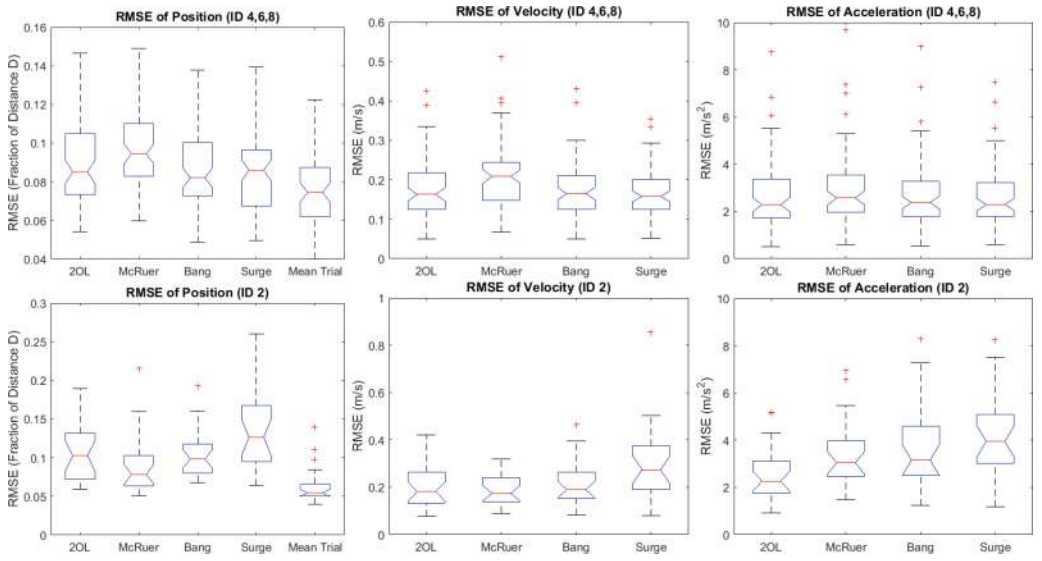

Fig. 9. RMSE of position (left column), velocity (middle column), and acceleration (right column) of different models. IDs 4,6, and 8 are shown together in the top row and ID 2 is shown separately in the bottom row. For position, RMSE is shown as a fraction of distance between targets, for velocity in ms^{-1} and for acceleration in ms^{-2}. Differences among models are small, but Bang-bang shows best fit and McRuer the worst. Note the difference in scale between the top and bottom rows. For position, ID 2, one outlier of Surge is not shown.

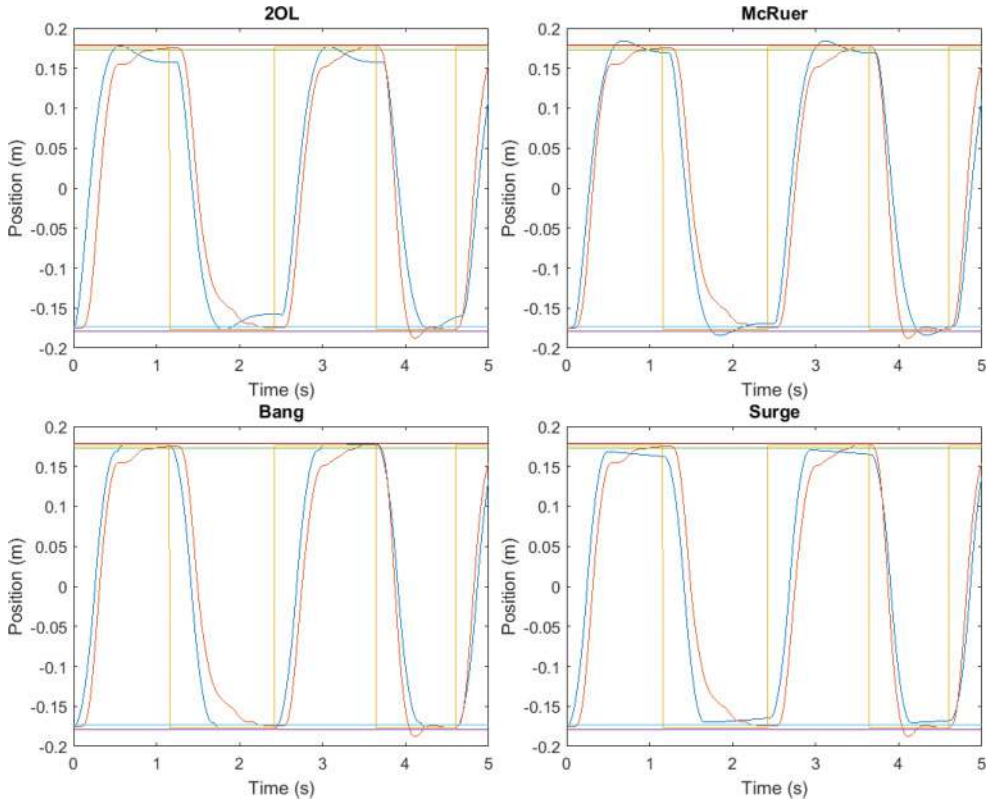

Fig. 10. Time series of pointer position predicted by the four models (blue), as well as the actual pointer position (red). The target position is shown for orientation (yellow). Bounds of the targets are shown as horizontal coloured lines (P1, ID 8). All models predict position well.

Figure 13 (left) shows the maximum velocity of different models and the participants. All models have a lower maximum velocity than the participants, with McRuer being lowest.

8.3 Acceleration

All models show characteristic deviations in acceleration behaviour from human behaviour. One major difference is that the models, because they are of second order, can generate a step change in acceleration. Because muscle activation and thus force and acceleration builds up over time, and also because of cognitive constraints, humans do not have this ability. Figure 9 (right column) shows the RMSE (in ms^{-2}) of acceleration of the four models. 2OL is the best in the ID 2 condition, with Surge as the worst. In the other ID conditions, the models show a similar level of fit, with McRuer slightly worse than the rest. Figure 13 shows the maximum acceleration of different models and the participants (middle) and the maximum deceleration of different models and the participants (right). This plot shows clear differences in acceleration and deceleration patterns among the models. In particular, 2OL has the highest maximum acceleration and Bang-bang the lowest for IDs 4, 6, and 8.

On the right, we see the maximum deceleration of different models and the participants. Also here, 2OL has the highest maximum deceleration for ID 2, while McRuer, Bang-bang, and Surge have maximum decelerations at or below 5 ms^{-2}. For the participants, Bang-bang, and Surge,

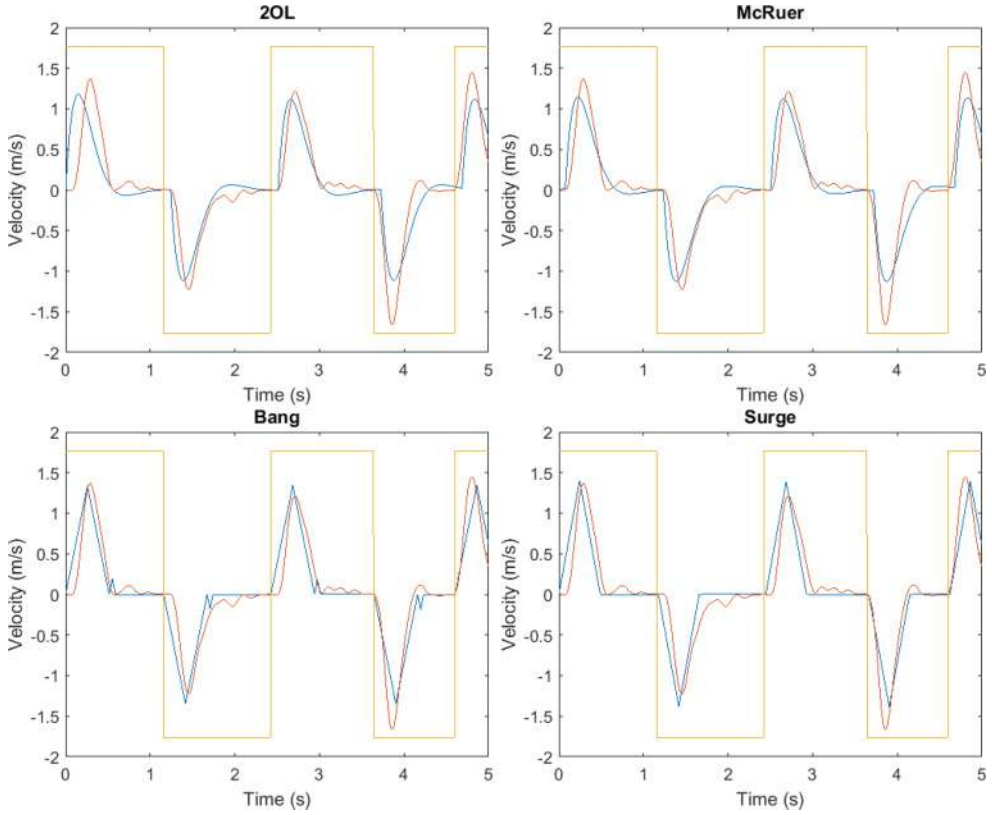

Fig. 11. Time series of pointer velocity predicted by the four models (blue), as well as actual pointer velocity (red). Predictions match observations well except for model-specific characteristics discussed in text. The changes of target position (yellow) are shown for orientation (P1, ID 8).

acceleration and deceleration are rather symmetric. In contrast, because of their linear nature, 2OL and McRuer show higher maximum acceleration than maximum deceleration values. Bang-bang and Surge clearly underestimate maximum acceleration and deceleration values. 2OL is the only model that reaches similar maximum acceleration and deceleration values as the participants. For McRuer maximum acceleration values are determined by the filtering of target position that we apply. Without filtering, maximum acceleration values of McRuer would be infinite.

Figure 14 shows time series of pointer acceleration predicted by the four models. 2OL and McRuer overestimate the acceleration, while underestimating the deceleration. It is further clear that 2OL and McRuer do not show any corrections. Bang-bang and Surge show a symmetric block-shaped acceleration behaviour. The acceleration pattern of this participant is in between these models, being more symmetric than the linear models, but less symmetric than Bang-bang and Surge. In this particular participant/distance/width combination, our optimisation process did not tune the parameters of the McRuer part of the Surge model to converge on the target. Instead, the model does not apply any acceleration after the surge and the pointer slowly drifts.

Figure 15 shows Hooke plots for all four models. These plots show clearly that the shape of the acceleration behaviour of the participants is in between the linear (2OL, McRuer) and maximum effort (Bang-bang, Surge) models. Both types of models show characteristic deviations from the human Hooke plot.

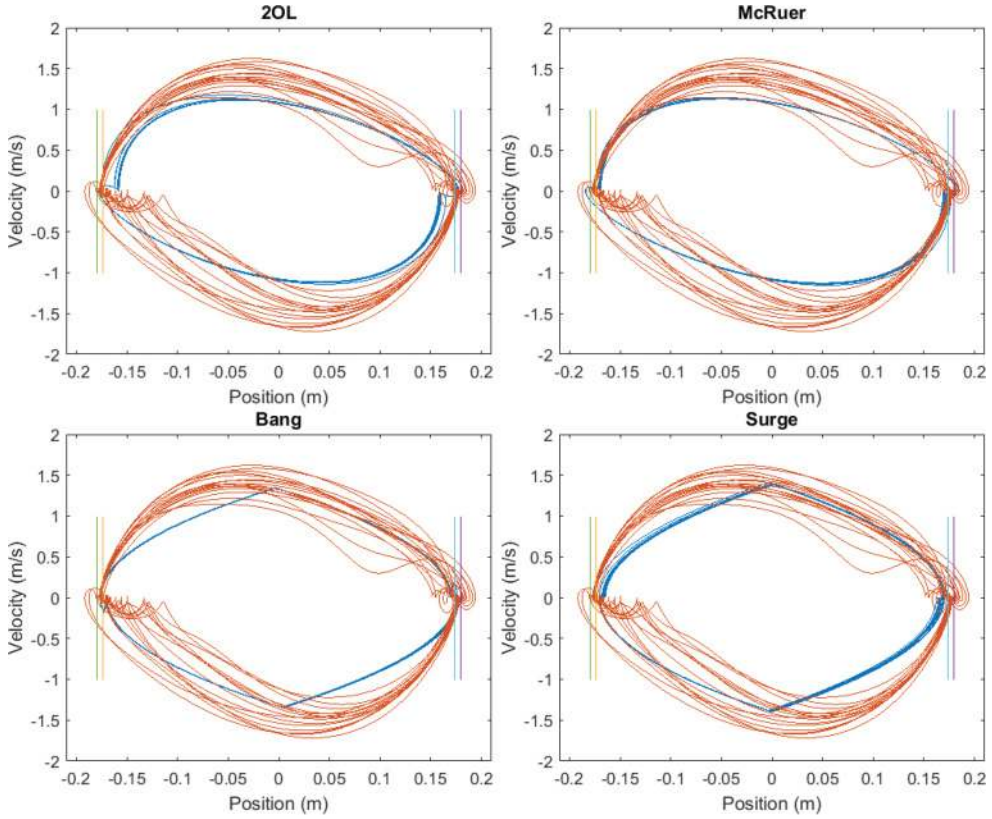

Fig. 12. Phase space plots as predicted by the four models (blue), as well as one participant's phase space plot (red) (P1, ID 8). Phase space plots expose deviations from human behaviour.

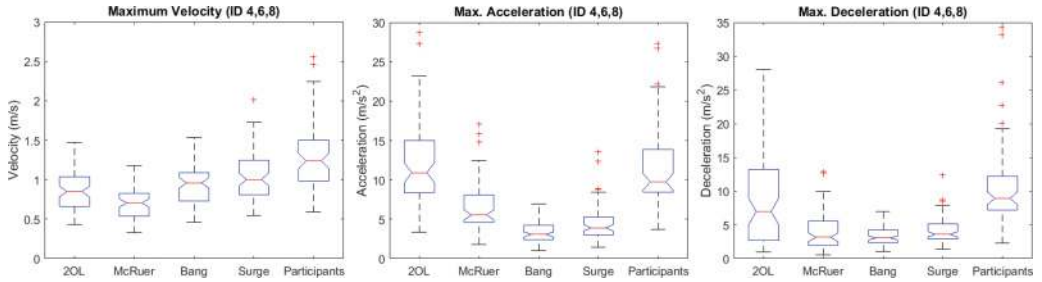

Fig. 13. *Left:* Maximum velocity of different models and participants (ID 4,6,8). *Middle:* Maximum acceleration of different models and participants (ID 4,6,8). *Right:* Maximum deceleration of different models and participants (ID 4,6,8).

8.4 Step Response

Figure 16 shows the step response of all models (red line) against the behaviour of P1 (grey lines) for one particular condition. It can be seen that model behaviour is close to human behaviour. The models do not always models converge to the target, as opposed to the human, as seen with the models here. The issue that models do not converge on the target might be an artefact of the reciprocal pointing task used. Because in this task the target switches very soon after the pointer

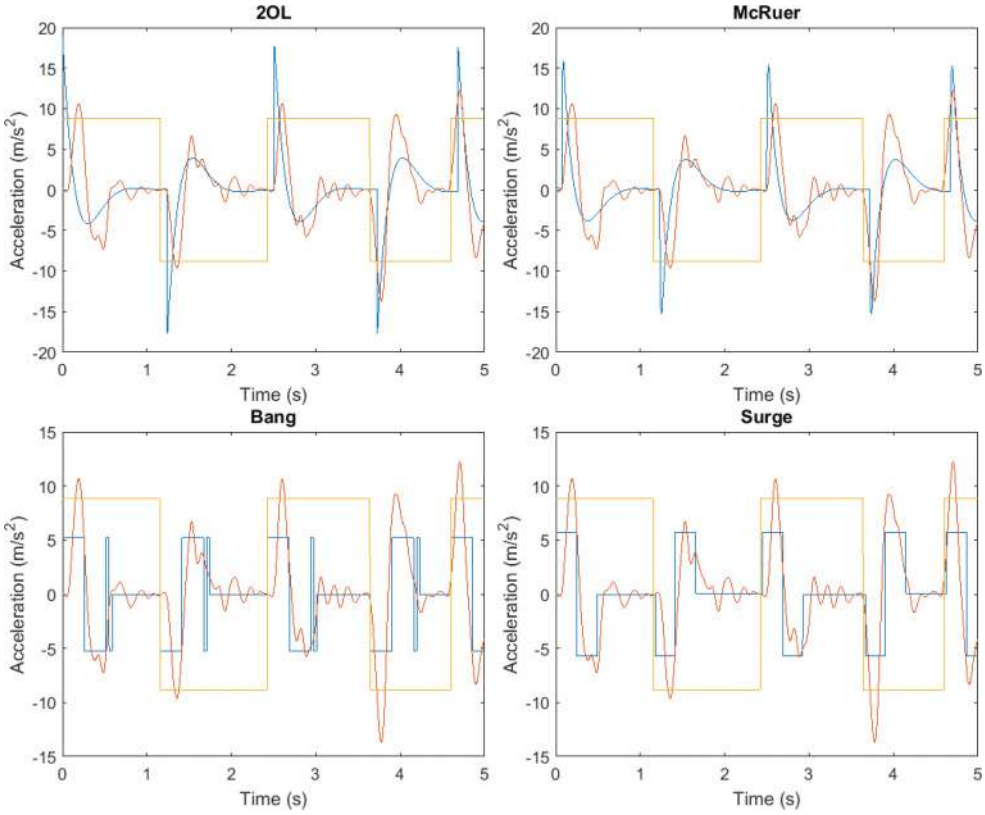

Fig. 14. Time series of pointer acceleration predicted by the four models (blue) as well as one participant's pointer acceleration (red). The changes of target position (yellow) are shown for orientation (P1, ID 8). For all four models, the acceleration profile shows characteristic deviations from the human. For the 2OL and McRuer, we can see that in particular the deceleration phase is smoother than the sharper human deceleration peak. Bang-bang and Surge show a characteristic block-shaped acceleration profile, where the human profile is more N shaped.

reached the target, it does not generate much training data penalising models which do not settle on the target.

Figure 17 shows an analysis of step response parameters for IDs 4, 6, and 8. From top left to bottom right, overshoot, rise time, settling time, and peak time are shown, respectively. 2OL and McRuer do not settle on the target in most cases. Therefore, they are omitted from the settling time plot. The figure shows differences among the models in overshoot, rise time, settling time, and peak time. Notably, McRuer has the highest overshoot and highest rise time. Bang-bang has no overshoot, but it has realistic settling time and good peak time. Surge has very little overshoot, realistic rise time, but very high settling and peak time. While 2OL has too high overshoot, when compared to participants, it has acceptable rise time and peak time.

8.5 Comparison of Task Demands (ID)

ID 2 turned out to be more difficult to model than IDs 4, 6, or 8. The inspection of model parameters of 2OL and Bang-bang for different IDs allows us to understand the effect of ID on a spring–mass

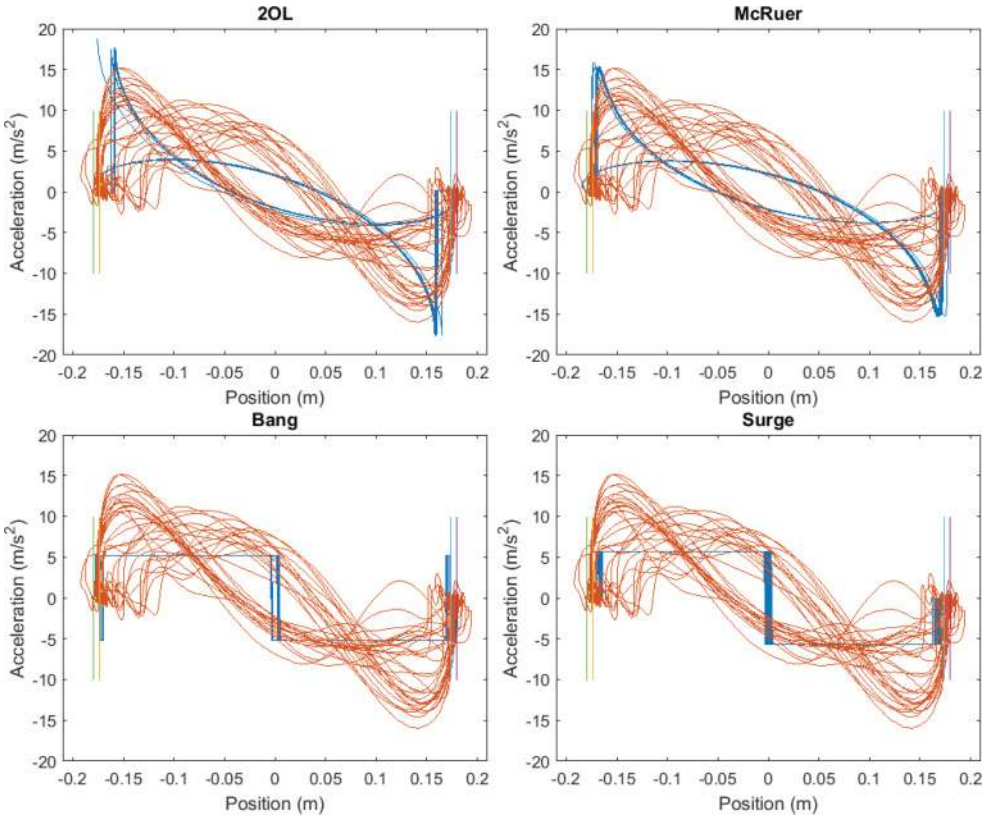

Fig. 15. Hooke plots as predicted by the four models (blue) as well as one participant's Hooke plots (red) (P1, ID 8). 2OL and McRuer are characterised by a steep acceleration and then slow deceleration towards the target (as control action is proportional to error, weakening while approaching the target). For Bang-bang and Surge, the acceleration and deceleration are more symmetric, similar to the human, but these models show the characteristic block-shaped behaviour, rather than the human 'N-shaped' form, which relates to the dynamic properties of the human arm. Note that the accelerations of movements to the right as well as movements to the left are plotted simultaneously.

system (2OL) and maximum effort model (Bang-bang), respectively. Figure 20 (left) shows the median RMSE of pointer position by ID.

8.5.1 Second-Order Lag. Figure 18 shows how the parameters of the 2OL change when the ID is increased. It can be seen that the mass m decreases with rising ID. The damping factor d is lower for ID 2, but relatively constant for IDs 4, 6, and 8. The spring constant k decreases with increasing ID. These parameters can be interpreted as the participants oscillating more, like a heavier mass with a stronger spring and less damping, with ID 2. As the ID increases, there is more damping, a lighter mass and a weaker spring, resulting in less oscillations. The dead time of the model decreases with increasing ID, in contrast to the participants' reaction time shown in Figure 8, top left.

8.5.2 Bang-bang. Figure 19 shows how the parameters of the Bang-bang model change when the ID is increased. In particular, the gain g, dead time, and undershoot all decrease with increasing ID. The decreasing gain can be interpreted as participants accelerating and decelerating the mouse more aggressively with ID 2.

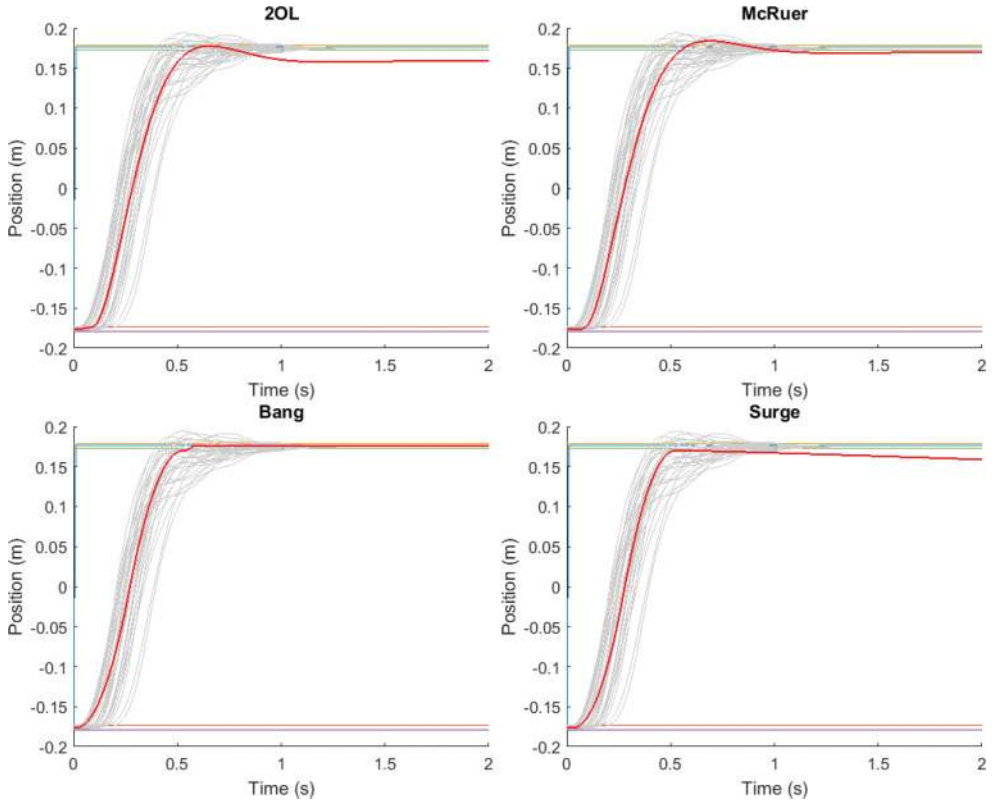

Fig. 16. Step response of the four models (red lines) as well as a number of trials of one participant (grey lines, P1, ID 8). The models are trained only for the times where participant data is available (before the click), such that behaviour after this time (about one second) is not necessarily realistic. For example, the models do not necessarily settle on the target.

8.6 Generalisability Across Users

For the analysis in preceding sections models are identified for specific participants and distance/width pairs. However, we are also interested how well models generalise. First, we are interested how well a model identified for a particular participant generalises to other participants. Second, we are interested how well a model identified for a specific ID generalises to other IDs. Figure 20 (middle and right) shows the results of this analysis. The middle graph shows how well models identified for a particular ID generalise to other IDs. For this analysis, the models identified for participant 1 and one condition (ID 8, target size 1.38mm) are tested against data from different IDs from the same participant. It can be seen that for most models, RMSE is relatively stable for ID 8, 6, and 4, for both target distances. The models do not generalise well to ID 2, with increasing RMSE.

The right graph shows how well models identified for a particular participant generalise to other participants. For this analysis, the models identified for participant 1 in one condition (ID 8, target size 1.38mm) are tested against data from different participants for the same distance/width pair. It can be seen that for some participants (participants 4, 5, 7), the models identified for participant 1 seem to fit even better than for participant 1. For other participants, in particular participants 2 and 9, the models identified for participant 1 do not seem to be a good fit.

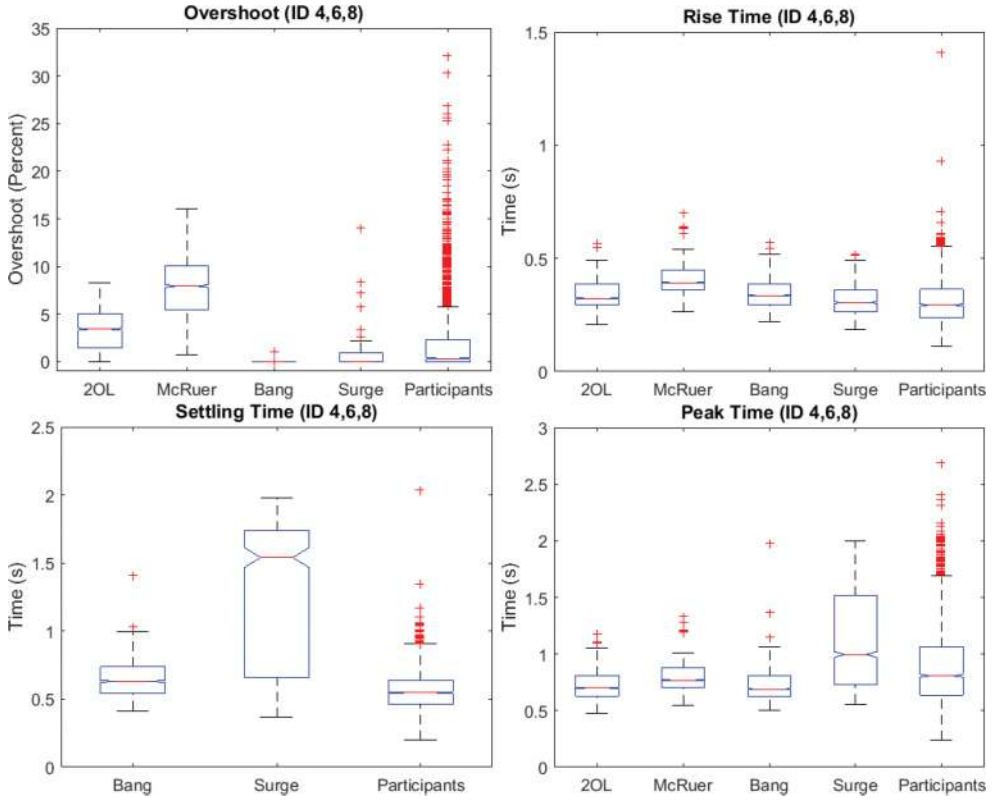

Fig. 17. Step response parameters of the four models and the participants. The plot exposes distinct patterns among the models. From top left to bottom right: Overshoot, rise time, settling time, and peak time, respectively. Overshoot has six hidden outliers.

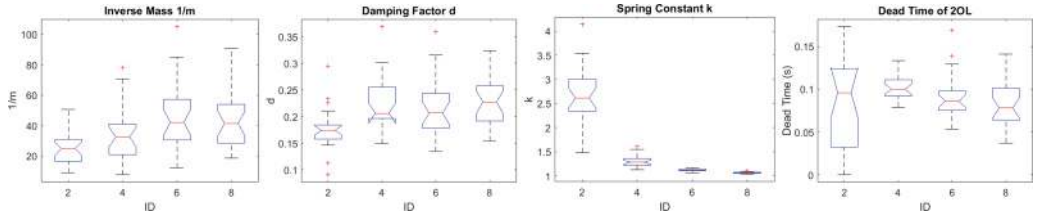

Fig. 18. Parameters of 2OL by ID. From left to right: $\frac{1}{m}$, d, k, and dead time. "Mass" m decreases, the damping factor d increases and the spring constant k decreases with increasing ID. This explains the slower reaction but reduced overshooting and less oscillations as the ID increases, leading in total to faster settling on small targets.

9 DISCUSSION

This article has explored the use of classical manual control models on pointing, a fundamental task in HCI. The article demonstrates that we can use recorded human pointing behaviour to optimise the parameters of classical control models, and that the identified control models can generate a range of dynamic behaviours that captures the human pointing behaviour to varying degrees. The analysis of the full trajectory, rather than just the end time and error rates brought

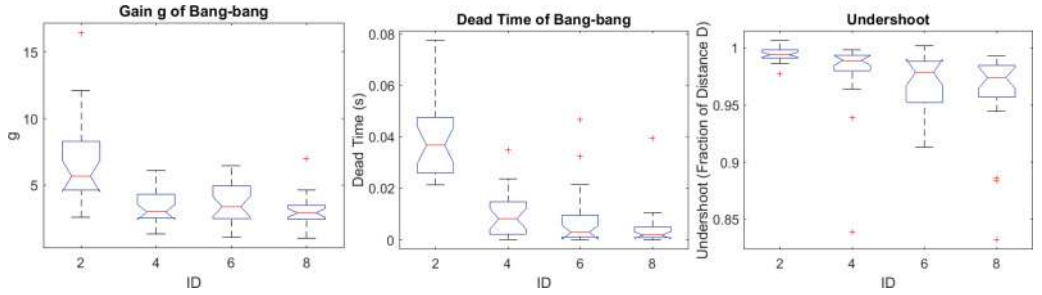

Fig. 19. Parameters of Bang-bang model by ID. From left to right: gain, dead time, and undershoot. Gain, dead time, and undershoot decrease with increasing ID.

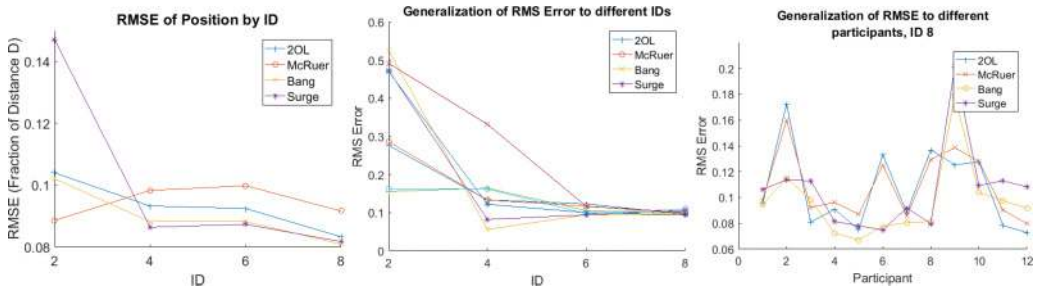

Fig. 20. *Left:* RMSE of Position by ID for all models. *Middle:* Shows how well a model identified for ID 8 generalises to other IDs. *Right:* Shows how well models identified for one participant generalise to other participants. In the middle and right case, models are identified for participant 1 from a condition with ID 8 and a target size of 1.38mm.

some new insight. To make this more accessible, we presented detailed analyses of our ground truth dataset using visualisations commonly used in control theory. They illuminate some dynamic behaviours not often analysed when modelling pointing in HCI, and we hope that they will provide many readers with new insight into the common mouse pointing task. The visualisations and metrics used make clear, e.g., that the classical models had a better fit with position and velocity characteristics than acceleration. A further key observation was the capture and analysis of the closed and open-loop (surge) stages in mouse pointing.

The detailed analysis of user surge behaviour, the first, ballistic stage of movement, is a novel contribution from this article. This provided insight into how people move towards different types of target, changing their strategy with different indices of difficulty. Participant time in the surge phase increased reliably with target width, from a median of 35% of the time in the surge for the smallest targets up to 100% for the largest. This gives insight into the likely advantages of multi-model approaches such as Costello's Surge model for harder targets, while simple bang-bang models may suffice for easier targets. Reaction time delays were lower for the easiest (ID=2) case. The proportion of distance to target covered in the surge stage in each case typically had a median greater than 95%, with more variability for easier indices of difficulty, but strongly biased to undershoot the target, rarely going beyond the target midpoint.

The notion of the *effective width* of a target has been used in Fitts' studies to adapt the value of W to fit actual user behaviour. We therefore investigated the distribution of end points for different indices of difficulty. We found a reliable increase of error variance as targets became larger, with the distribution of clicks matching the targets well, until $W = 23.5$mm, after this, for $W = 70.6$

and 118mm, the distribution was smaller than the actual target. Increasing target width also led to participants systematically clicking short of the target centre (up to a median of 17% of W short for the largest). This increase in variability of end point due to casual behaviour by participants in the underconstrained cases highlighted the limitations of a control behaviour assuming consistent optimal behaviour.

The choice of model depends on what aspects of user behaviour should be modelled.

- The second-order lag (2OL) is a very simple linear model, and has an intuitive physical interpretation as spring–mass–damper system with dead time. In terms of RMSE, it is comparable to the other models, despite its simplicity.
- McRuer's model performs very similarly to the 2OL in our case of step tracking. McRuer's model does, however, have the ability to react to the derivative of the error, which might be a strength with moving targets. A fundamental limitation of linear models, such as 2OL or McRuer, is that acceleration is necessarily proportional to error, so they cannot replicate the symmetric phase space and Hooke plots that we observe with some participants. There were also some mundane practical experiences in our data capture – both 2OL and McRuer models did not settle on the target when the parameters are identified through our optimisation process, as a side effect of the data acquired in the final stages – a good example of the constraints associated with data-driven models.
- The Bang-bang controller can generate a symmetric, maximum-effort phase space and Hooke plot characteristic for open-loop behaviour. The acceleration behaviour of the model is block-shaped, assuming the ability to instantly reach full acceleration and deceleration, without taking into account the dynamic constraints of a human body. This means that the model had a different acceleration profile, and then also underestimated maximum acceleration and deceleration compared to the participants. In contrast to the participants, in our implementation this model never overshoots. We did augment the model to take into account the reaction time and systematic undershoot bias that we observed in participants.
- For the reciprocal pointing task we investigated, the Surge model does not behave very differently from the Bang-bang model. The Surge model would however better model situations in which the target moves (tracking tasks), where the pure Bang-bang model would be likely to perform poorly.

9.1 Future Work

This article has focussed on relatively simple, parametric, deterministic models and has used them to control objects with no internal dynamics on a task requiring movement to a static area. These models also have potential for more extensive use than just pointing, as they are well suited for general control tasks, such as tracking trajectories and for disturbance rejection. The next steps will be (1) more powerful, general models, (2) application of control concepts for analysis of interaction loops for more complex systems, and (3) use of control theory to stimulate novel approaches to interaction design.

Future developments in *more powerful models of pointing* are likely to explore (1) explicit representation of human variability and (2) more powerful non-parametric or highly flexible, non-linear black box models from the machine learning literature. Non-linear models will be able to generate a more accurate representation of the dynamics, and can potentially also capture the variability of human behaviour. Mottet and Bootsma [37] show how polynomial models can capture the 'N-shaped' human dynamic behaviour for a specific reciprocal tapping task, although generalising the model to arbitrary distances and target sizes would require more work and data. It would be relatively straightforward to use non-parametric models such as non-parametric Bayesian Gaussian

Process priors [46] as part of the modelling framework presented in this article, e.g., to represent a more 'human-like', smoothed bang-bang surge behaviour, rather than the simple discrete representation used here. The combination of dynamic generative models with a representation of variability becomes a powerful tool for predictive interfaces, which can anticipate targets during the trajectory.

The *design and analysis of interaction with controlled objects with more complex dynamic behaviour* will help us strengthen the conceptual foundations of HCI. As mentioned in the related work section, while this article does not vary the dynamics of controlled objects, it does provide an engineering framework which is well-suited for such analysis. The insight into how different human control strategies vary can inform the design of system dynamics to improve the overall closed-loop behaviour. Techniques such as 'sticky mouse' dynamics, non-linear mouse transfer functions, adaptive magnification or speed dependent zooming can all be readily represented by dynamic models, allowing rigorous analysis of their closed-loop behaviour. Similarly, the impact on closed-loop dynamics by sensor noise, or user uncertainty about the system as something that combines statistical analysis and dynamic systems, as in e.g., [54].

We are also optimistic that *the control perspective can inspire designers* to propose novel interaction styles. An example of this is interfaces which can infer the user's intent based on detection of control behaviour, as developed in [58] and built on by [21] and [9] and [55].

We conclude that pointing dynamics and control theoretic models are a rich area for future work in HCI. Beyond modelling, the approach may have value in enhancing the bandwidth of human-computer interaction and allowing us to make better use of novel sensing and feedback mechanisms which would not be trivial to work with using current approaches. We observe that modern GUIs primarily exploit buttons and other spatially expanded targets for communication of intent. The widespread use of touchscreen smartphones has increased the use of dynamic gestures, such as swipes, flicks and taps. However, these are often nothing more than 'glorified buttons', where a discrete gesture is recognised and a discrete event is triggered, meaning that the information content, and hence the interaction potential expressed in continuous motion has been disregarded. Being able to use dynamic state changes to control interaction could open up new styles of interaction which go beyond button-pushing, and we believe that use of control concepts could strengthen the foundations of our understanding of human-computer interactions, and offer the basis for explorations which can move us beyond the predominant 'discrete' user interface paradigm.

9.2 Conclusions

Taken together, the results suggest that control-theoretical models warrant serious reconsideration as a theoretical and practical tool in the analysis and design of human-computer interfaces. Control theoretic models are a complement, not a replacement, to Fitts' law. Being able to predict and understand the dynamics of user behaviour at finer granularity is worthwhile for HCI for a number of reasons. While Fitts' law's strength is that it abstracts from variability in data, the control theoretic approach allows us to explore the sources of variability in detail, potentially leading to a deeper understanding of interaction dynamics for the complete human-computer interaction loop.

We believe one major reason why control theoretic approaches have not received a lot of attention in the HCI literature is the difficulty of applying models, obtaining datasets, as well as lack of tools and procedures. Previously, mathematical training and extensive knowledge were needed to engage in control-theoretical modelling, and Fitts' law based models were simply easier to deploy. Now, however, there are software tools that help practitioners to obtain, represent, and analyse control theoretical models, and by making the Simulink models and experimental data freely

available, we hope that others can more easily build on this work, significantly lowering the barrier for deploying control theoretical models in HCI.

ACKNOWLEDGMENTS

We would like to thank Yves Guiard, Shumin Zhai, Henrik Gollee, John Williamson, David Murray-Smith, Myroslav Bachynskyi, and the reviewers of this and an earlier version of this article submitted to CHI 2016 for their helpful comments.

REFERENCES

[1] Anand Agarawala and Ravin Balakrishnan. 2006. Keepin'it real: Pushing the desktop metaphor with physics, piles and the pen. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1283–1292.

[2] Rob C. Barrett, Edwin J. Selker, Joseph D. Rutledge, and Robert S. Olyha. 1995. Negative inertia: A dynamic pointing function. In *Proceedings of the Conference Companion on Human Factors in Computing Systems (CHI'95)*. ACM, New York, NY, 316–317. DOI: http://dx.doi.org/10.1145/223355.223692

[3] Reinoud J. Bootsma, Laure Fernandez, and Denis Mottet. 2004. Behind fitts law: Kinematic patterns in goal-directed movements. *International Journal of Human-Computer Studies* 61, 6 (2004), 811–821.

[4] Daniel Bullock and Stephen Grossberg. 1988. Neural dynamics of planned arm movements: Emergent invariants and speed-accuracy properties during trajectory formation. *Psychological Review* 95, 1 (1988), 49.

[5] David J. Cannon. 1994. Experiments with a target-threshold control theory model for deriving Fitts' law parameters for human-machine systems. *IEEE Transactions Systems Man and Cybernetics* 24, 8 (1994), 1089–1098.

[6] Géry Casiez and Nicolas Roussel. 2011. No more bricolage!: Methods and tools to characterize, replicate and compare pointing transfer functions. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST'11)*. ACM, New York, NY, 603–614. DOI: http://dx.doi.org/10.1145/2047196.2047276

[7] Géry Casiez, Daniel Vogel, Ravin Balakrishnan, and Andy Cockburn. 2008. The impact of control-display gain on user performance in pointing tasks. *Human–Computer Interaction* 23, 3 (2008), 215–250.

[8] Sung-Jung Cho, Roderick Murray-Smith, and Yeun-Bae Kim. 2007. Multi-context photo browsing on mobile devices based on tilt dynamics. In *Proceedings of the 9th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI'07)*. ACM, New York, NY, 190–197. DOI: http://dx.doi.org/10.1145/1377999.1378006

[9] Christopher Clarke, Alessio Bellino, Augusto Esteves, Eduardo Velloso, and Hans Gellersen. 2016. TraceMatch: A computer vision technique for user input by tracing of animated controls. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp'16)*. ACM, New York, NY, 298–303. DOI: http://dx.doi.org/10.1145/2971648.2971714

[10] Edward M. Connelly. 1984. A control model: An alternative interpretation of Fitts' law. In *Proceedings of the 28th Annual Meeting of the Human Factors Society*.

[11] Richard G. Costello. 1968. The surge model of the well-trained human operator in simple manual control. *IEEE Transactions on Man-Machine Systems* 9, 1 (1968).

[12] Kenneth J. W. Craik. 1947. Theory of the human operator in control systems: 1. The operator as an engineering system. *British Journal of Psychology. General Section* 38, 2 (1947), 56–61.

[13] Kenneth J. W. Craik. 1948. Theory of the human operator in control systems: 2. Man as an element in a control system. *British Journal of Psychology. General Section* 38, 3 (1948), 142–148.

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

[15] Gavin Doherty and Mieke Massink. 1999. Continuous interaction and human control. In *Proceedings of the XVIII European Annual Conference on Human Decision Making and Manual Control*. 80–96.

[16] Parisa Eslambolchilar and Roderick Murray-Smith. 2004. Tilt-based automatic zooming and scaling in mobile devices – a state-space implementation. In *Mobile HCI 2004*, S. Brewsterand M. Dunlop (Eds.), Vol. 3160. Springer LNCS, 120–131.

[17] Parisa Eslambolchilar and Roderick Murray-Smith. 2006. Model-based, multimodal interaction in document browsing. In *Proceedings of the Machine Learning for Multimodal Interaction, LNCS Volume 4299*. Springer, 1–12. DOI: http://dx.doi.org/10.1007/11965152

[18] Parisa Eslambolchilar and Roderick Murray-Smith. 2008. Control centric approach in designing scrolling and zooming user interfaces. *International Journal of Human-Computer Studies* 66, 12 (2008), 838–856.

[19] Parisa Eslambolchilar and R. Murray-Smith. 2008. Model-based target sonification in small screen devices: Perception and action. *Handbook of Research on User Interface Design and Evaluation for Mobile Technology* (2008), 478–506.

[20] Giorgio P. Faconti and Mieke Massink. 2001. Continuous interaction with computers: Issues and requirements. In *Proceedings of the HCI*. Citeseer, 301–305.

[21] Jean-Daniel Fekete, Niklas Elmqvist, and Yves Guiard. 2009. Motion-pointing: Target selection using elliptical motions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'09)*. ACM, New York, NY, 289–298. DOI: http://dx.doi.org/10.1145/1518701.1518748

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

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

[24] Peter Gawthrop, Henrik Gollee, and Ian Loram. 2015. Intermittent control in man and machine. In *Event-Based Control and Signal Processing*, Marek Miskowicz (Ed.). CRC Press, 281–350. DOI: http://dx.doi.org/DOI: 10.1201/b19013-16

[25] Peter Gawthrop, Martin Lakie, and Ian Loram. 2008. Predictive feedback control and Fitts' law. *Biological Cybernetics* 98 (2008), 229–238. DOI: http://dx.doi.org/doi:10.1007/s00422-007-0206-9

[26] Yves Guiard. 1993. On Fitts's and hooke's laws: Simple harmonic movement in upper-limb cyclical aiming. *Acta Psychologica* 82, 1 (1993), 139–159.

[27] Yves Guiard and Olivier Rioul. 2015. A mathematical description of the speed/accuracy trade-off of aimed movement. In *Proceedings of the 2015 British HCI Conference*. ACM, 91–100.

[28] Richard J. Jagacinski and John M. Flach. 2003. *Control Theory for Humans: Quantitative Approaches to Modeling Performance*. Lawrence Erlbaum, Mahwah, New Jersey.

[29] Charles R. Kelley. 1968. *Manual and Automatic Control: A Theory of Manual Control and Its Applications to Manual and to Automatic Systems*. Academic Press.

[30] David L. Kleinman, Sheldon Baron, and W. H. Levison. 1971. A control theoretic approach to manned-vehicle systems analysis. *IEEE Transactions Automatic Control* 16 (1971), 824–832.

[31] Sven Kratz, Ivo Brodien, and Michael Rohs. 2010. Semi-automatic zooming for mobile map navigation. In *Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services (MobileHCI'10)*. ACM, New York, NY, 63–72. DOI: http://dx.doi.org/10.1145/1851600.1851615

[32] Lennart Ljung. 1987. *System Identification — Theory for the User*. Prentice Hall, Englewood Cliffs, New Jersey.

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

[34] Duane T. McRuer and Henry R. Jex. 1967. A review of quasi-linear pilot models. *IEEE Transactions on Human Factors in Electronics* 8, 3 (1967), 231–249.

[35] David E. Meyer, Richard A. Abrams, Sylvan Kornblum, Charles E. Wright, and J. E. Keith Smith. 1988. Optimality in human motor performance: Ideal control of rapid aimed movements. *Psychological Review* 95, 3 (1988), 340.

[36] David E. Meyer, J. E. Keith Smith, Sylvan Kornblum, Richard A. Abrams, and Charles. E. Wright. 1990. Speed-accuracy trade-offs in aimed movements: Toward a theory of rapid voluntary action. *Attention and Performance XIII*. M. Jeannerod (Ed.), 173–226.

[37] Denis Mottet and Reinoud J. Bootsma. 1999. The dynamics of goal-directed rhythmical aiming. *Biological Cybernetics* 80, 4 (1999), 235–245.

[38] Winston L. Nelson. 1983. Physical principles for economies of skilled movements. *Biological Cybernetics* 46, 2 (1983), 135–147.

[39] Rejean Plamondon. 1998. A kinematic theory of rapid human movements: Part III. Kinetic outcomes. *Biological Cybernetics* 78, 2 (1998), 133–145. DOI: http://dx.doi.org/10.1007/s004220050420

[40] Réjean Plamondon and Adel M. Alimi. 1997. Speed/accuracy trade-offs in target-directed movements. *Behavioral and Brain Sciences* 20, 2 (1997), 279–303.

[41] Henning Pohl and Roderick Murray-Smith. 2013. Focused and casual interactions: Allowing users to vary their level of engagement. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'13)*. ACM, New York, NY, 2223–2232. DOI: http://dx.doi.org/10.1145/2470654.2481307

[42] E. C. Poulton. 1974. *Tracking Skill and Manual Control*. Academic Press, New York.

[43] William T. Powers. 1973. *Behavior: The Control of Perception*. Aldine de Gruyter.

[44] Philip Quinn, Sylvain Malacria, and Andy Cockburn. 2013. Touch scrolling transfer functions. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST'13)*. ACM, New York, NY, 61–70. DOI: http://dx.doi.org/10.1145/2501988.2501995

[45] Philip Quinn and Shumin Zhai. 2016. Modeling gesture-typing movements. *Human-Computer Interaction* (2016), 1–47. DOI: http://dx.doi.org/10.1080/07370024.2016.1215922

[46] Carl E. Rasmussen and Christopher K. I. Williams. 2006. *Gaussian Processes for Machine Learning*. MIT Press. http://dx.doi.org/10.1080/07370024.2016.1215922

[47] Lionel Rigoux and Emmanuel Guigon. 2012. A model of reward-and effort-based optimal decision making and motor control. *PLoS Comput Biol* 8, 10 (2012), e1002716.

[48] Joseph D. Rutledge and Edwin J. Selker. 1990. Force-to-motion functions for pointing. In *Proceedings of the IFIP TC13 Third International Conference on Human-Computer Interaction*. North-Holland Publishing Co., 701–706.
[49] Abraham Savitzky and Marcel J. E. Golay. 1964. Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry* 36, 8 (1964), 1627–1639.
[50] Richard A. Schmidt and Timothy D. Lee. 2005. *Motor Control and Learning*. Human Kinetics.
[51] Reza Shadmehr. 2010. Control of movements and temporal discounting of reward. *Current Opinion in Neurobiology* 20, 6 (2010), 726–730. DOI: http://dx.doi.org/10.1016/j.conb.2010.08.017 Motor systems, Neurobiology of behaviour.
[52] Thomas B. Sheridan and William R. Ferrell. 1974. *Man-Machine Systems: Information, Control, and Decision Models of Human Performance*. M.I.T. Press, Cambridge, MA.
[53] Emanuel Todorov and Michael I. Jordan. 2002. Optimal feedback control as a theory of motor coordination. *Nature Neuroscience* 5, 11 (2002), 1226–1235.
[54] Dari Trendafilov and Roderick Murray-Smith. 2013. Information-theoretic characterization of uncertainty in manual control. In *Proceedings of the 2013 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 4913–4918.
[55] E. Velloso, M. Carter, J. Newn, A. Esteves, C. Clarke, and H. Gellersen. 2017. Motion correlation: Selecting objects by matching their movement. *ACM Transactions on Computer-Human Interaction (TOCHI)* 24, 3 (2017).
[56] Christopher D. Wickens and Justin G. Hollands. 1999. *Engineering Psychology and Human Performance*. Prentice Hall.
[57] Norbert Wiener. 1948. *Cybernetics: Control and Communication in the Animal and the Machine*. Wiley, New York.
[58] John Williamson and Roderick Murray-Smith. 2004. Pointing without a pointer. In *Proceedings of the ACM SIG CHI 2004 Conference on Human Factors in Computing Systems*. ACM, 1407–1410.
[59] John Williamson, Roderick Murray-Smith, and Stephen Hughes. 2007. Shoogle: Excitatory multimodal interaction on mobile devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'07)*. ACM, New York, NY, 121–124. DOI: http://dx.doi.org/10.1145/1240624.1240642
[60] Jacob O. Wobbrock, Edward Cutrell, Susumu Harada, and I. Scott MacKenzie. 2008. An error model for pointing based on Fitts' law. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1613–1622.
[61] Robert Sessions Woodworth. 1899. Accuracy of voluntary movement.*The Psychological Review: Monograph Supplements* 3, 3 (1899), i.
[62] Shumin Zhai, Jing Kong, and Xiangshi Ren. 2004. Speed–accuracy tradeoff in Fitts' law tasks on the equivalency of actual and nominal pointing precision. *International Journal of Human-Computer Studies* 61, 6 (2004), 823–856.
[63] Brian Ziebart, Anind Dey, and J. Andrew Bagnell. 2012. Probabilistic pointing target prediction via inverse optimal control. In *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces*. ACM, 1–10.

Received May 2016; revised November 2016; accepted January 2017