Understanding the variability of pointing tasks with event-driven intermittent control [★]

J. Alberto Alvarez-Martin [*] Thomas Doublein [*] Henrik Gollee [*]
Jörg Müller [**] Roderick Murray-Smith [***]

[*] *James Watt School of Engineering, University of Glasgow, Glasgow, UK (e-mails: alberto.alvarez-martin@glasgow.ac.uk, t.doublein.1@research.gla.ac.uk, henrik.gollee@glasgow.ac.uk)*
[**] *Institute for Computer Science, University of Bayreuth, Bayreuth, Germany (e-mail: joerg.mueller@uni-bayreuth.de)*
[***] *School of Computing Science, University of Glasgow, Glasgow, UK (e-mail: roderick.murray-smith@glasgow.ac.uk)*

Abstract: Event-driven Intermittent control (IC) has been used as a framework to explain relevant aspects of human movement. The events, which are generated by states crossing predefined thresholds, give rise to state trajectories that mimic those of a continuous controller, by only using feedback information at event times. Here we present the results of using an optimisation approach to identify the parameters of an intermittent controller from experimental data, where users performed one dimensional mouse movements in a reciprocal pointing task. The results show that IC is able to reproduce both, the dynamical features and the variability of the pointing task across participants. We then introduce probabilistic elements in the IC framework in the form of Gaussian processes as an additional method to represent human movement variability.

Keywords: Intermittent control, human motor control, variability, pointing, gaussian processes.

1. INTRODUCTION

Understanding the control strategy that humans implement to interact with computers in order to change their state is at the core of the field of Human-Computer interaction (HCI). Most of these interactions come through the use of pointer devices, where the user moves a mouse (input device) using the hand to control the position of a pointer on a screen. The process of interacting with a computer is dynamic by nature and it happens under the influence of a feedback loop. The user observes the current state of the pointer and adjusts their strategy to steer the system to the desired state. Control theory is the mathematical framework for systems in the presence of feedback loops and it has been used in the past to better understand human-motor control and HCI in general (Gawthrop et al. (2015), Müller et al. (2017), Murray-Smith (2018)).

The purpose of this paper is to introduce event-driven Intermittent Control (IC) as a special type of control and as a plausible framework to explain the experimental results of a pointing task as well as the observed variability of the recorded movements, by presenting the results from Müller et al. (2017) and Martín et al. (2021). Additionally, initial results based on extensions made to the IC framework are shown as an alternative method to represent human movement. These extensions are probabilistic in nature due to the use of Gaussian Processes (GP) that replace key elements in IC, introducing variability in the observed trajectories in a simple way without the need to introduce noise or disturbances.

[★] Funding support provided by EPSRC through grant EP/R018634/1 (Closed-loop Data Science) and a PhD Scholarship.

2. INTERMITTENT CONTROL

Traditional feedback control can be seen as applying an input signal that is generated using observations which are often combined into an estimate of the state, to a dynamic system cf. Fig. 1. In continuous control (bottom left in Fig. 1), the input signal is computed based on continuously observed information. In contrast, IC (Gawthrop et al. (2011), Gawthrop et al. (2015)), shown in the bottom right of Fig. 1, relies on an event-trigger to use feedback at specific moments in time in order to compute a control trajectory that is applied in an open-loop configuration.

The IC framework is described with more detail as a block diagram in Fig. 2, based on the versions presented in Gawthrop et al. (2011) and Gawthrop et al. (2015). Together, the *Hand-Arm Biomechanics* and *Input device PTF* (Pointing Transfer Function) blocks from Fig. 2 represent a mathematical model of the neuromuscular dynamics of the user and the user interface. We assume that the overall dynamics can be expressed as a linear state-space model of order n that can be written as follows:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)\,, \quad \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t)\,, \tag{1}$$

with $\mathbf{x}(0) = \mathbf{x}_0$ as initial condition and where $\mathbf{x} \in \mathbb{R}^n$ corresponds to the system state such as pointer position and velocity, and muscle activation, $\mathbf{y} \in \mathbb{R}^{n_y}$ corresponds to the output such as pointer position on the display, and $\mathbf{u} \in \mathbb{R}^{n_u}$ corresponds to the input such as muscle excitation. $\mathbf{A}$ is a $n \times n$ matrix representing how the system evolves without control input, $\mathbf{B}$ is a $n \times n_u$ matrix, representing the impact of control signals such as muscle excitation on the system, and $\mathbf{C}$ is a $n_y \times n$ matrix, representing how the system state maps to the users' observation, such as pointer position on the display.

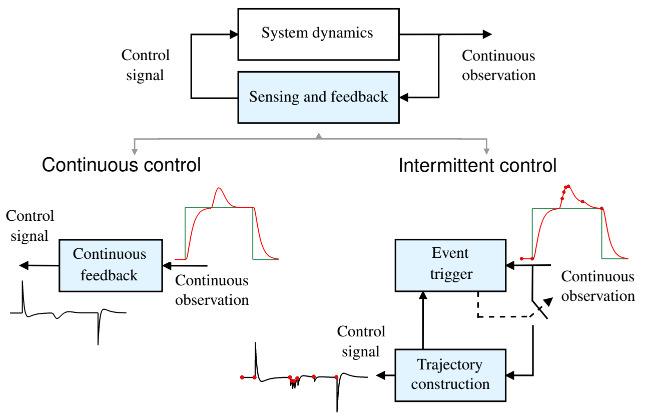

Fig. 1. Feedback control of a dynamic system: In continuous control (bottom left block diagram), the control signal (time-series in black) is generated continuously, based on observations of the system (time-series in red). In intermittent control (bottom right), feedback information from continuous observations is only used intermittently (determined by an *event trigger*) to reset an open-loop control signal trajectory. The instances when feedback is used in intermittent control are indicated by circles on the red and black time-series, representing the system output and the control signal, respectively. The control signal trajectories between two consecutive red circles (or *events*) are generated open-loop, i.e. without the use of feedback.

The goal of the user is to steer the output $\mathbf{y}(t)$ as close as possible to the target signal $w(t)$ (e.g., the pointing target). A recurrent problem in control is that finding a suitable state-feedback control input might not be possible using only the observations. For example, in pointing, the user observes the pointer position. If the pointer velocity and muscle activation states can not be observed directly, they must be estimated in order to compute a control input signal. One way to solve this is to feed the input $\mathbf{u}(t)$ and the output $\mathbf{y}(t)$ to a linear *observer*, which is a dynamical model based on (1), that reconstructs the full state of the system $\hat{\mathbf{x}}(t)$ using only output measurements and information on the input that is being applied. In IC, once the target signal $w(t)$ has been introduced as $\mathbf{x}_w(t) = \hat{\mathbf{x}}(t) - \boldsymbol{x}_{ss}w(t)$, the resulting observed states $\mathbf{x}_w(t)$ get sampled at discrete points in time t_i. In this case, $\boldsymbol{x}_{ss}$ corresponds to the steady-state version of the states and it can be computed offline according to equations 8 and 9 in Martín et al. (2021).

The *predictor* block in Fig. 2 is to compensate for transmission time-delays which might be present in the feedback loop both from the users neural system and from the computer system, by predicting the future states $\mathbf{x}_p(t_i)$ based on the state estimates $\mathbf{x}_w(t_i)$. These predictions are used by the *hold*, a dynamical model that mimics the behaviour of the overall closed-loop system. The *hold* mechanism is a central feature of IC and its main purpose is to generate the state vector $\mathbf{x}_h(t)$ using feedback information only when a sample is taken at t_i. The hold states $\mathbf{x}_h(t)$ follow the trajectories of an ideal closed-loop system in the absence of disturbances or noise, so the *hold* is sometimes seen as an internal model that is used by IC to control the system when it is evolving in an open-loop configuration.

The *trigger* compares the hold states $\mathbf{x}_h(t)$ and the observed states $\mathbf{x}_w(t)$ to generate a prediction error e_p. An event is generated by the trigger at t_i once e_p exceeds a predefined threshold q. The hold states $\mathbf{x}_h(t)$ are used to compute a *state feedback* control input $\mathbf{u}(t)$ continuously, which is the signal that drives the neuromuscular system producing the final input to the system $\mathbf{u}_e(t)$. The *State feedback* block represents the control gain vector $\mathbf{k}$ which can be designed offline using traditional methods such as pole-placement or linear optimal control (Goodwin et al. (2001)). Both the input $\mathbf{u}(t)$ and the output $\mathbf{y}(t)$ can be affected by motor noise $\mathbf{v}_u(t)$ and sensory noise $\mathbf{v}_y(t)$ respectively.

The time between event instants is known as the *intermittent interval* Δ_i. The ith intermittent interval is defined as $\Delta_{ol} = \Delta_i = t_{i+1} - t_i$. Also, IC makes use of a a continuous variable τ that is restarted every Δ_i according to $\tau = t - t_i$. A lower limit $\Delta_{ol}^{\min}$ can be specified within a given intermittent interval as $\Delta_i > \Delta_{ol}^{\min} > 0$. The lower limit $\Delta_{ol}^{\min}$, also known as *minimum open-loop interval*, has been used to model the *psychological refractory period* observed in human motor control (Gawthrop et al. (2011)). The formal definition and implementation details of the predictor, observer, hold, state-feedback and trigger blocks are established in Martín et al. (2021).

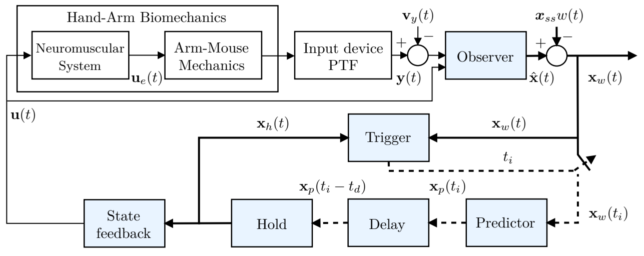

Fig. 2. Block diagram of intermittent control. The hold states $\mathbf{x}_h$ are compared with the state-estimates $\mathbf{x}_w$ in the trigger block. If the difference exceeds a predefined threshold, then the block creates events at times denoted by t_i. The hold states $\mathbf{x}_h$ are used to generate the control signal $\mathbf{u}$ during the open-loop period, and it is reset only at times t_i by the predictor block. The dashed lines represent signals that are defined only at t_i. The Hand-Arm Biomechanics block contains the Neuromuscular System which generates the force that is applied to the input device and the Arm-Mouse Mechanics block that translates it into a position for the pointing transfer function (PTF).

3. EXPERIMENT SETUP

The IC framework from the previous section was used to identify the parameters of a standard intermittent controller based on experimental data collected from a pointing experiment [1] (Müller et al., 2017), where 12 participants were asked to control a pointer shown on a screen. The pointer reflected the changes in the x-dimension of the mouse and it was restricted to move only horizontally.

The task consisted of clicking a number of one-dimensional targets that were displayed in sequence. During a block of trials, the active target was presented in a different colour compared to the previous target, and the distance between them, as well as the width of the target, stayed constant throughout the block. Each block uses a specific combination of distance and width; a total of 8 combinations were applied for each participant: 1) distance of 212 mm and widths of 0.83, 3.32, 14.1 and 70.6 mm, 2) distance of 353 mm and widths of 1.38, 5.54, 23.5, 118 mm. According to Fitts' law (Fitts, 1954), the corresponding

[1] `http://joergmueller.info/controlpointing/`

Index of Difficulty (ID) for each distance and target width combination are 8, 6, 4 and 2. Each participant had to complete a full block of trials per ID for the two distance conditions. The experiment blocks were designed to have 102 trials each, divided into 22 trials for training and 80 for the rest of the task.

4. MODEL ASSUMPTIONS AND CONTROL DESIGN

The block labelled as *Neuromuscular system* in Fig. 2 was implemented as a second order system with time-constants of 50 ms (Gawthrop et al., 2011),

$$G_{nms}(s) = \frac{1}{(0.05s+1)^2}\,, \tag{2}$$

where $G(s)$ is the transfer function in the Laplace domain. This transfer function was converted to an equivalent state-space system in the form of equation (1) with

$$\mathbf{A}_{nms} = \begin{bmatrix} -20 & 20 \\ 0 & -20 \end{bmatrix}, \quad \mathbf{B}_{nms} = \begin{bmatrix} 0 \\ 20 \end{bmatrix}, \quad \mathbf{C}_{nms} = \begin{bmatrix} 1 & 0 \end{bmatrix}. \tag{3}$$

Similarly, the block shown in Fig. 2 as *Arm-Mouse Mechanics* was established as a double integrator in the state-space representation of (1) with

$$\mathbf{A}_{sys} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad \mathbf{B}_{sys} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \mathbf{C}_{sys} = \begin{bmatrix} 1 & 0 \end{bmatrix}, \tag{4}$$

where the associated state vector is $\mathbf{x}_{sys}(t) = \begin{bmatrix} \mathbf{x}_{pos}(t) & \mathbf{x}_{vel}(t) \end{bmatrix}^T$, comprised of the pointer position $\mathbf{x}_{pos}(t)$ and velocity $\mathbf{x}_{vel}(t)$. This assumes that the output of the neuromuscular system, depicted as $\mathbf{u}_e(t)$ in Fig. 2, is a force generated by the participant, applied to a unit mass in a low friction environment, and that the output of the overall system, $\mathbf{y}(t)$, is the position of the mass, which in this case is equivalent to the position of the pointer on the screen. The combination of (3) and (4) yields a fourth order system based on the full state vector $\mathbf{x}(t)$, which is then used to design the overall controller. The feedback gain vector $\mathbf{k}$ (defined in Martín et al. (2021)), needed to generate the final control law $\mathbf{u}(t)$ was designed using optimal control via the LQR design method (Goodwin et al., 2001). This involves the minimisation of the LQR cost function

$$J_{LQR} = \int_0^\infty \left[\mathbf{x}(t)^T\mathbf{Q}_c\mathbf{x}(t) + \mathbf{u}(t)^T\mathbf{R}_c\mathbf{u}(t)\right] dt\,, \tag{5}$$

and the solution of its associated algebraic Riccati equation. Both $\mathbf{x}(t)$ and $\mathbf{u}(t)$ in (5) are weighted by design matrices $\mathbf{Q}_c$ (an $n \times n$ diagonal matrix that must be positive semi-definite) and $\mathbf{R}_c$ (an $n_u \times n_u$ diagonal positive definite matrix, where n_u corresponds to the number of inputs in the system). Using the same LQR method on the dual problem of state estimation, the closed-loop observer gain matrix $\mathbf{L}$, can be computed to force the estimation error to vanish asymptotically, by choosing a design matrix $\mathbf{Q}_o$. In this case, only the first state of the state vector $\mathbf{x}(t)$, corresponding to the pointer position $\mathbf{x}_{pos}(t)$, was used by the observer to estimate the rest to the unknown states. The matrices $\mathbf{Q}_c$ and $\mathbf{Q}_o$ were defined as parameters to be optimised, whereas $\mathbf{R}_c$ was fixed for the entire process with a value of 1. The time-delay t_d remained constant at 0.01 sec for all participants and conditions. To detect events, a threshold q was applied only to $\mathbf{x}_{pos}(t)$. However, the actual value of q was obtained via the optimisation process.

5. OPTIMISATION APPROACH

Based on the combination of (3) and (4), an optimisation procedure was used to fit a set of controller parameters to the experimental data. The following parameters were identified as a result of the optimisation process: The LQR design matrix $\mathbf{Q}_c$ (the four elements in its diagonal since the off-diagonal terms are all zero), the observer gain $\mathbf{Q}_o$, the prediction error threshold q for triggering purposes, the minimum open-loop interval $\Delta_{ol}^{\min}$ and a mismatch gain $\mathbf{A}_p = 1-p$, for a total of 8 free parameters. The purpose of $\mathbf{A}_p$ is to model the cases where the control input that is applied to system is different by a fixed amount from what it should be by design. This is implemented as follows: from Fig. 2, the output of the neuromuscular system $\mathbf{u}_e(t)$, which serves as the input to the system in (4), is multiplied by a quantity p, resulting in $\dot{\mathbf{x}}_{sys}(t) = \mathbf{A}_{sys}\mathbf{x}_{sys}(t) + \mathbf{B}_{sys}\mathbf{u}_{sys}(t)$, where $\mathbf{u}_{sys}(t) = p\mathbf{u}_e(t)$. When $p = 1$, the full input is applied. A mismatch is generated when p is different than 1. The optimisation process identified the value of p directly; however, the mismatch gain $\mathbf{A}_p$ is reported in the following sections as it gives a better insight on the difference between $\mathbf{u}_e(t)$ and $\mathbf{u}_{sys}(t)$. A pattern-search method was used for all subjects and conditions as implemented in Matlab's Optimisation Toolbox.

5.1 *Data partitioning*

The data was partitioned according to the following criteria: the time elapsed between two successive changes in the target position would constitute a full *slice*. Essentially, each slice is composed of the pointer trajectories made by the subject to reach consecutive targets (two consecutive trials). An individual IC is then fitted to each slice. In Fig. 3, a prototypical target signal is used to illustrate how the data was partitioned into individual slices. Each block had 80 trials, this resulted in 40 full slices per block. The optimisation was carried out using the information of the last 20 slices in the block, leaving the rest for evaluation. 20 controllers were optimised for each of the 8 conditions in the experiment. This was repeated for all 12 participants. Altogether, 1920 individual ICs were obtained.

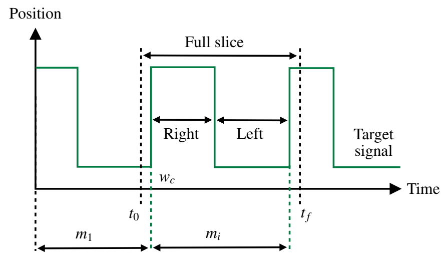

Fig. 3. Data partitioning diagram. A slice starts at t_0, which is 10 samples before a change in reference w_c (movement to the target on the right), and ends 10 samples after the next change in reference in the same direction (t_f). The second portion of the slice (shown as *Left*) corresponds to the movement from the right to the left target. As described in section 5.3, m_1 is the best controller in bank m and m_i is a randomly selected controller from the same bank.

5.2 *Model fitness via cost functions*

To establish how well the intermittent controller fits the experimental data, we use a general cost function J measuring the difference between pointer movement predicted by the model and actual pointer movement, which also includes the difference in terms of the pointer velocity. The optimiser considers the root mean squared error (RMSE) of these quantities as follows

$$J = c_p\sqrt{\frac{\sum_{i=1}^{n_j}(\hat{y}_i - y_i)^2}{n_j}} + c_v\sqrt{\frac{\sum_{i=1}^{n_j}(\hat{v}_i - v_i)^2}{n_j}}\,, \tag{6}$$

where $\hat{y}$ and $\hat{v}$ are the simulated pointer position and velocity respectively. The number of samples considered for each slice is shown as n_j and the constants c_p and c_v denote values that act as weights for each of the errors. For the optimisation procedure, the values of c_p and c_v were set to 0.5, resulting in equal weights for each of the error terms in (6).

5.3 Controller switching

The resulting models of the optimisation procedure can be used to carry out a simulation over a collection of trials for each participant. To create a controller that captures some of the inherent variability of human control, we used a multiple-model approach, instead of a single optimised model for all trials. A bank of optimised models m was generated for each participant. This bank is essentially a ranked list according to the value of (6), which contains the intermittent controllers that were derived using the optimised parameters for each slice, resulting in 20 controllers per condition. At the start of the simulation, the best controller in the bank, or m_1, is used against the first trial, which corresponds to an initial pointer movement towards the right target followed by a second movement to the left target. Fig. 3 shows the moment in time when the target changes (w_c), which is when a new controller m_i is selected randomly from m and applied for the duration of the next trial only. This selection mechanism starts after the first trial has finished and is kept in use for the remaining trials.

6. OPTIMISATION RESULTS

The optimised parameters for all participants are shown in Fig. 4 when grouped by ID: the threshold q (A), the gain $\mathbf{A}_p$ (B), the minimum open-loop interval $\Delta_{ol}^{\min}$ (C), the observer gain $\mathbf{Q}_o$ (D) and the four elements in the diagonal of matrix $\mathbf{Q}_c$ (E, F, G, H). The data distribution for each parameter is shown as a violin plot for the two target distances of 212 mm (left side) and 353 mm (right side) respectively.

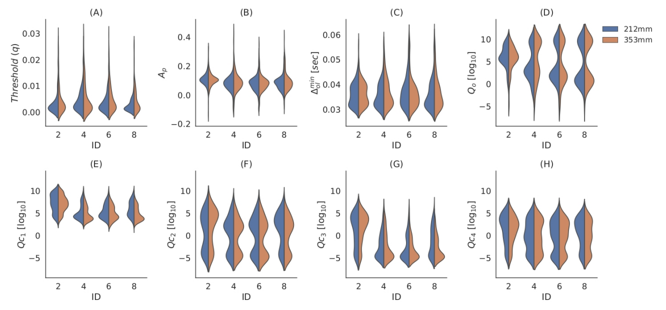

Fig. 4. Optimised controller parameters for each condition. (A) the threshold q, (B) mismatch gain $\mathbf{A}_p$, (C) the minimum open-loop interval $\Delta_{ol}^{\min}$, (D) the observer gain $\mathbf{Q}_o$, and the four elements in the diagonal of matrix $\mathbf{Q}_c$ (E, F, G, H), are shown as violin plots including data of all subjects and categorised by ID (horizontal axis). Both $\mathbf{Q}_o$ and $\mathbf{Q}_c$ are shown in log scale. The results are also grouped according to the two values of distance between targets used in the experiment (left: 212 mm and right: 353 mm). The shape of the distributions for ID 2 is slightly different compared to the rest of the conditions.

For the threshold q (Fig. 4A), all conditions exhibit long tails and similar distributions, indicating the presence of higher thresholds in some of the models. The mismatch gain $\mathbf{A}_p$ in Fig. 4B shows that the data distributions are wider for ID 4, 6, and 8, suggesting that there is less of a mismatch compared to ID 2. The minimum open-loop interval $\Delta_{ol}^{\min}$ in Fig. 4C is similar for all conditions, with the distributions covering the 0.03 to 0.05 sec range. The optimised parameters in matrix $\mathbf{Q}_c$ are shown in 4E, F, G, and H, using a log scale. The first and second elements of $\mathbf{Q}_c$ (Fig. 4E and Fig. 4F) are of particular importance since $\mathbf{Q}_{c1}$ is associated to the pointer position state and $\mathbf{Q}_{c2}$ to the pointer velocity state. $\mathbf{Q}_{c1}$ shows higher values in all conditions compared to the other elements in $\mathbf{Q}_c$, this suggests that the controller puts more emphasis on the position state in order to reduce the overall error. The higher values of $\mathbf{Q}_{c1}$, $\mathbf{Q}_{c2}$, and $\mathbf{Q}_{c3}$ in ID 2 reveal that as the ID increases, a higher degree of precision is needed which forces the controller to become more cautious by producing smaller feedback gains.

7. DYNAMIC RESPONSES

In Fig. 5, the phase planes and Hooke plots of participant 10 are shown and compared to a simulation generated by the multiple-model IC approach. Each row corresponds to a specific ID, starting from ID 2 at the top and ending with ID 8 at the bottom. The first two columns from left to right contain data for a distance of 212 mm, the third and fourth columns correspond to 353 mm. The simulated phase planes (in red) follow the behaviour of the human response (in blue); however, the IC response captures the variability from the experimental result in all conditions, being slightly less accurate as the ID increases.

The phase planes in Fig. 5E and Fig. 5O, are a reflection of the type of control strategy applied by the participant. Once the trajectory approaches the target, the control policy that is used results in a reduction in velocity followed by a sequence of corrections to reduce the error (more frequent in difficult conditions), or in an error in the opposite direction in the form of overshoot. The simulated IC not only covers almost the entire range of possible trajectories generated by this participant, but also reproduces the overshoot behaviour that was previously mentioned. The Hooke plots (second and fourth columns) compare how the pointer acceleration changes with time. In Fig. 5F and Fig. 5P, the acceleration trajectory generated by the IC is smooth (in red), and follows the experimental result (blue) in terms of its overall shape. For ID 2 in Fig. 5A and Fig. 5B, the control strategy changes significantly and a more fluid motion is observed, which requires less corrections as the pointer approaches the target, which agrees with the parameter distributions for ID 2 in Fig. 4.

8. PROBABILISTIC ELEMENTS IN IC

One of the benefits of the approach described in the previous sections, where parameters are optimised for each trial and then a bank of controllers is used to simulate trajectories for a pointing task, is that the observed variability arises from the triggering process that is inherent to IC and the constant switch to a different controller during the block of trials, without the need for added noise. However, having a control framework where the variability can be generated via open-loop trajectories and effective triggering alone is appealing because only one controller would be responsible for a wide range of responses, simplifying the estimation of the full set of parameters. For this reason, we have explored the use of probabilistic elements, in the form of Gaussian processes, to replace the *Hold* element in the IC framework, which is responsible for the production of

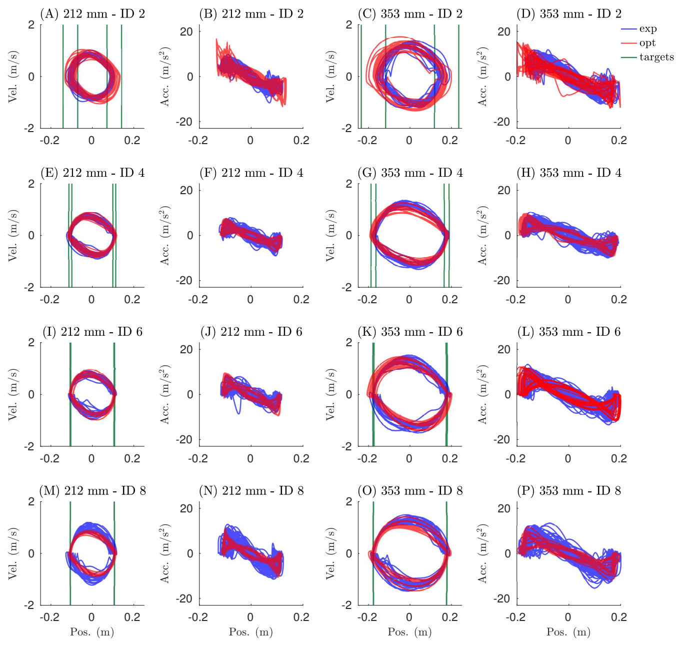

Fig. 5. Phase planes and Hooke plots for participant 10 in all conditions. Each row corresponds to a particular ID, starting with 2 for the top row and ending with 8 at the bottom. The two columns on the left show the phase planes (pointer position vs velocity) and Hooke plots (pointer position vs acceleration) for a distance between targets of 212 mm (A, B, E, F, I, J, M, N). The two columns on the right show the same quantities for a distance of 353 mm (C, D, G, H, K, L, O, P). The participant's response is in blue, while the trajectories generated by the IC are in red. Green vertical lines show the target signal.

open-loop trajectories. We now introduce our initial results in the context of human balancing.

8.1 *Gaussian processes and the hold mechanism*

The *Hold* mechanism in Fig. 2 produces a state vector $\mathbf{x}_h(t)$ that, in combination with the state-feedback gain $\mathbf{k}$, generates the control input $\mathbf{u}(t)$ that is applied to the system to be controlled. In our standard formulation of IC, the process of computing $\mathbf{x}_h(t)$ involves a closed-loop matrix $\mathbf{A}_c = \mathbf{A} - \mathbf{Bk}$, that depends on the system matrices defined in (1). At the start of every intermittent interval i.e., at t_i, and until the next event appears, an autonomous dynamical model is defined as follows:

$$\dot{\mathbf{x}}_h(t) = \mathbf{A}_c\mathbf{x}_h(t)\,, \tag{7}$$

where $\dot{\mathbf{x}}_h$ is the derivative of the hold state vector w.r.t. time. This ensures that in the absence of disturbances or noise, the hold states $\mathbf{x}_h$ would match those of the underlying closed-loop system, with the distinction that they only use feedback at event times (when (7) is restarted). The *hold* formulation in (7) is deterministic in nature and provides a solid frame of reference in order to decide when to trigger an event. However, if (7) is replaced by a GP using:

$$\mathbf{A}_c\mathbf{x}_h(t) \sim \mathcal{N}(0, K)\,, \tag{8}$$

which depends on a kernel matrix K, then a prediction of the hold state for the next time instant can be made:

$$\dot{\mathbf{x}}_h(t) = \mathbf{x}_h(t+1) = \mathcal{N}(\mathbf{x}_h(t)|0, K)\,. \tag{9}$$

This recursive formulation allows the computation of hold state trajectories of finite length. The implementation of a GP in order to do prediction and the obtention of the kernel matrix K, as in (9), depends on an off-line optimisation process that is based on training data generated by the system that is being approximated (Rasmussen and Williams, 2006). A diagram of the proposed hold in (9) is shown in Fig. 6, which is based on the standard IC diagram of Fig. 2. In this case, the *Hold* block has been expanded to show that each component of the state vector $\mathbf{x}_h(t)$ is approximated by an individual GP. The *Arm-Mouse Mechanics* and *Input device PTF* blocks are replaced by a more general representation using a *System* block which corresponds to the actual system being controlled.

Two alternatives have been explored to generate the hold states $\mathbf{x}_h(t)$ based on the GP in (9):

(1) To use the mean of the distribution embedded in the GP to compute the prediction for the next open-loop interval.
(2) To draw a sample from the distribution and use it for the next open-loop interval.

These two alternatives produce hold states that eventually lead to stable closed-loop controllers; however, the difference in behaviour obtained from using one or the other is noticeable. The first method results in behaviour that is closer to the one observed when the standard hold in (7) is used. Interestingly, the second alternative adds a richer variety in terms of the range of the responses. These results are illustrated in the next section using a simulation of a human balance model.

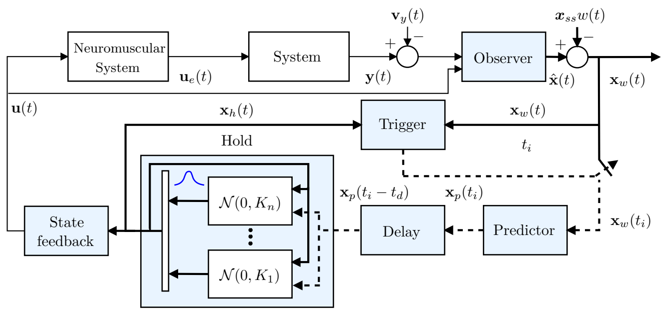

Fig. 6. Intermittent Control using a GP based hold. The *Hold* block is expanded, compared to Fig. 2, to show how individual GPs predict the trajectories of the states in $\mathbf{x}_h(t)$. The control input $\mathbf{u}(t)$ drives the *Neuromuscular System* and *System* blocks, where the latter represents the dynamical model of the system that is being controlled.

8.2 *The inverted pendulum model of human balance*

The single inverted pendulum model has been used to describe the human balance control problem both in simulation and using experimental data. For the purpose of our simulations, the dynamic bias model of human standing described in (Lakie et al., 2003; Gawthrop et al., 2011) is used. This model considers that the control signal applied to maintain a human-size inverted pendulum balanced is generated by a tendon connected in series with a contractile element (in this case the calf muscle) which is in charge of generating a balancing torque. The model can be written in a state space form, as in (1), if the following state vector is defined: $\mathbf{x}(t) = \begin{bmatrix}\dot{\theta} & \theta\end{bmatrix}^T$, where θ and $\dot{\theta}$ are the angular position and velocity. Using $\mathbf{x}$, the final model is:

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} -\frac{V}{J} & \frac{(1-c)mgh}{J} \\ 1 & 0 \end{bmatrix}\mathbf{x}(t) + \begin{bmatrix} \frac{cmgh}{J} \\ 0 \end{bmatrix}\mathbf{u}(t) \tag{10}$$

$$\mathbf{y} = \begin{bmatrix} 0 & 1 \end{bmatrix}\mathbf{x}(t)\,. \tag{11}$$

The constants in (10) are: the tendon stiffness ratio $c = 0.85$, the moment of inertia $J = 77$ kg m^2, the ankle viscosity $V = 2.9$ Nm rad^{-1}, the mass of the pendulum $m = 70$ kg, the distance from the ankle joint to the centre of mass $h = 0.92$ m, and the gravitational acceleration constant $g = 9.81$ m s^{-2}, as reported in Loram et al. (2009). c being smaller than 1 implies that the tendon stiffness is not enough to stabilise the pendulum on its own, requiring additional control effort provided by the muscle.

8.3 Simulation results using a GP based hold

The model in (10) was used to run a comparison simulation between the traditional hold of the IC framework, referred to as *Standard hold* and the GP based hold with the two sampling alternatives described in section 8.1: 1) Mean based GP (mGP) and 2) Sample based GP (sGP). The parameters of the three intermittent controllers were: threshold q of 0.01 rad applied to both θ and $\dot{\theta}$, a minimum open-loop interval $\Delta_{ol}^{\min}$ of 0.25 sec, a 2 by 2 identity matrix as $\mathbf{Q}_c$, and $\mathbf{R}_c = 1$. An initial condition of 0.01 rad for θ was used while $\dot{\theta}$ was set to 0. First, a comparison between the standard hold (blue) and the mean based GP hold (mGP), in red, is presented in Fig. 7, where the two states $\dot{\theta}$ and θ are shown in (A) and (B) respectively and the associated phase plane in (C). Both responses are similar, small differences arise when the mGP based hold is used, specially in the regions close to zero angle and velocity, but overall the same pattern is obtained with the two holds.

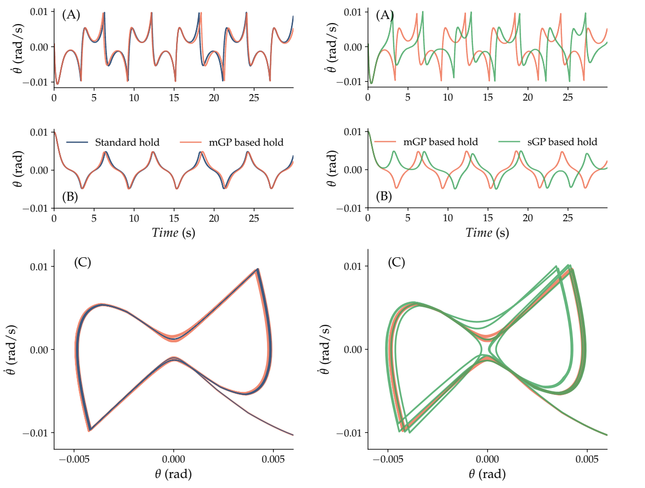

Fig. 7. **Left:** Standard (blue) vs mGP hold (red) comparison. (A) shows the evolution of $\dot{\theta}$, (B) shows the angle θ, and (C) is a phase plane representation of $\dot{\theta}$ vs. θ. Small differences observed between the responses.
Right: mGP hold (red) vs sGP hold (green) comparison. (A) shows the evolution of $\dot{\theta}$, (B) shows the angle θ, and (C) is a phase plane representation of $\dot{\theta}$ vs. θ. The differences between the two responses are more evident, starting to diverge at around 2 seconds. The phase plane in (C) shows this behaviour clearly, especially the max/min values of θ when close to zero.

To illustrate the increased range of the responses obtained when a sample from the distribution is taken (sGP) instead of using just the mean (mGP), a comparison was made between the two holds and the results are shown in Fig. 7(right).The time-series for both states in (A) and (B) show differences in behaviour after only two seconds, which is expected. The overall trend of the trajectories generated by the sGP hold is comparable to both the standard hold and mGP, but due to event times being slightly different in all of them and the fact that in mGP the starting point of the trajectory is determined by the standard deviation of the distribution, a wider arrange of patterns emerge. For instance, in Fig. 7(right, B), around 5 and 24 sec, the angle θ does not cross to the other side of the vertical reference at 0 rad and it bounces back to the same side where it came from.

9. CONCLUSIONS

The results presented in this paper show how IC is a plausible framework to study and understand human movement. In particular, the results of applying IC to the experimental data recorded from pointing task in the HCI context show that the triggering process of IC and the proposed multiple-model approach are capable of reproducing the dynamical features of the trajectories generated by the participants. These results provide a starting point to study the effects of closed vs. open-loop configurations and the role of intermittency in HCI, by looking at methodologies that are not only physiologically compatible but that are also supported by the existing theory of control systems. Also, the GP based extensions mentioned in the previous section add a probabilistic point of view to the IC framework, resulting in a controller that is capable of being precise and at the same time provide a wider range or responses to better represent the variability of human movement.

REFERENCES

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

Gawthrop, P., Gollee, H., and Loram, I. (2015). Intermittent control in man and machine. In M. Miskowicz (ed.), *Event-Based Control and Signal Processing*, Embedded Systems, chapter 14, 1–99. CRC press, London.

Gawthrop, P., Loram, I., Lakie, M., and Gollee, H. (2011). Intermittent control: A computational theory of human control. *Biological Cybernetics*, 104(1-2), 31–51.

Goodwin, G.C., Graebe, S.F., and Salgado, M.E. (2001). *Control System Design*. Prentice Hall, New Jersey, USA.

Lakie, M., Caplan, N., and Loram, I. (2003). Human balancing of an inverted pendulum with a compliant linkage: neural control by anticipatory intermittent bias. *Journal of Physiology*, 551, 357–370.

Loram, I., Lakie, M., and Gawthrop, P. (2009). Visual control of stable and unstable loads: what is the feedback delay and extent of linear time-invariant control? *The Journal of Physiology*, 587(Pt 6), 1343–65.

Martín, J.A.Á., Gollee, H., Müller, J., and Murray-Smith, R. (2021). Intermittent Control as a Model of Mouse Movements. *ACM Transactions on Computer-Human Interaction*, 28(5), 35:1–35:46. doi:10.1145/3461836.

Müller, J., Oulasvirta, A., and Murray-Smith, R. (2017). Control theoretic models of pointing. *ACM Trans. on Computer-Human Interaction*, 24(4), 27:1–27:36.

Murray-Smith, R. (2018). Control theory, dynamics and continuous interaction. In A. Oulasvirta, P.O. Kristensson, X. Bi, and A. Howes (eds.), *Computational Interaction*, 17–42. Oxford University Press, Oxford.

Rasmussen, C.E. and Williams, C.K.I. (2006). *Gaussian Processes for Machine Learning*. MIT Press.