HAL
archives-ouvertes.fr

An adaptive FIR filter for trajectory prediction and latency reduction in direct Human-Computer interactions

Stanislav Aranovskiy, Rosane Ushirobira, Denis Efimov, Géry Casiez

▶ **To cite this version:**

Stanislav Aranovskiy, Rosane Ushirobira, Denis Efimov, Géry Casiez. An adaptive FIR filter for trajectory prediction and latency reduction in direct Human-Computer interactions. Control Engineering Practice, Elsevier, In press, 10.1016/j.conengprac.2019.07.011 . hal-02189181

HAL Id: hal-02189181
https://hal.inria.fr/hal-02189181

Submitted on 19 Jul 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

An adaptive FIR filter for trajectory prediction and latency reduction in direct Human-Computer interactions

S. Aranovskiy[a,b,*], R. Ushirobira[c], D. Efimov[c], G. Casiez[d]

[a] *Equipe Automatique (AUT), CentraleSupélec – IETR, CS 47061 Avenue de la Boulaie, 35576 Cesson-Sévigné, France*
[b] *Department of Control Systems and Informatics, ITMO University, Kronverkskiy pr., 49, St. Petersburg, 197101, Russia*
[c] *Inria, Univ. Lille, CNRS, UMR 9189 – CRIStAL - Centre de Recherche en Informatique Signal et Automatique de Lille, F-59000 Lille, France.*
[d] *Univ. Lille, CNRS, Inria, UMR 9189 – CRIStAL - Centre de Recherche en Informatique Signal et Automatique de Lille, F-59000 Lille, France.*

Abstract

The problem of latency reduction in direct Human-Computer interactions is considered. The proposed method is based on a frequency-domain approximation of a non-causal ideal predictor with a finite impulse response filter. Given a sufficiently rich dataset, the parameters of the filter can be either optimized off-line or tuned on-line with the proposed adaptive algorithm. The performance of the proposed solution is evaluated in an experimental study consisting of drawings on a touchscreen.

Keywords: human-computer interaction, latency reduction, adaptive FIR filter, human motion prediction, touchscreen

1. Introduction

Typically, human-computer interactions can be divided into two categories: direct and indirect interactions. In indirect interactions, the input

*Corresponding author

Email addresses: stanislav.aranovskiy@centralesupelec.fr (S. Aranovskiy), rosane.ushirobira@inria.fr (R. Ushirobira), denis.efimov@inria.fr (D. Efimov), gery.casiez@univ-lille1.fr (G. Casiez)

device (*e.g.* a mice or a trackball) and the output device (*e.g.* a display) are separated. In contrast, for direct interactions, the input and the output devices are coupled together, and the input and the system output (observed by the user) share the same screen. Examples of direct interactions include smartphones, pads and touch screens.

For any type of interaction some time is required for the input device to process the user input, to transfer it to the operating system, then to the specific application, to the graphical layer, and finally to display the output. This route gives rise to the end-to-end latency, which is the time lag between the input action performed by the user and the reaction output displayed by the hardware. The diagram of this route in direct interactions is given in Fig. 1. For modern touch-screen devices, it is reasonable to expect the end-to-end latency about 60 ms or more, as measured by Ng et al. (2012a). As shown by Casiez et al. (2017), even for high-end touch interfaces, end-to-end latency is at least around 50 ms in an application simply changing the background color, where most of the latency comes from the display. End-to-end latency can be reduced with higher input and output frequencies (see Table 6 by Casiez et al. (2017)), but this comes at the price of more expansive hardware and higher power consumption.

Measuring the end-to-end latency and its variations is not a trivial task itself, and several studies were taken on this issue recently (Bérard and Blanch (2013); Casiez et al. (2015)). However, the detrimental impact of latency to user performance has been known for a long time, see MacKenzie and Ware (1993). Direct interactions are more sensitive to latency, and Jota et al. (2013) found that latency greater than 25 ms can significantly affect the user performance in touch dragging tasks. At the same time, Ng et al. (2012b) show that latency as small as 10 ms can be perceived in direct interactions.

There are two ways to reduce the impact of latency. The first is at the hardware level, *e.g.* by using more reactive and advanced elements and by making the signal flow as fast as possible. This approach has two drawbacks, the high cost of advanced components and the increased energy consumption, which play a significant role for portable devices.

The second way to reduce latency is at the software level by using latency reduction algorithms. From the control point of view, this problem can be formulated as a trajectory prediction or forecasting problem, and convenient prediction methods can be used. However, methods based on underlying dynamic models cannot be easily applied for latency reduction since dynamic models of user behavior are not typically available. There exist

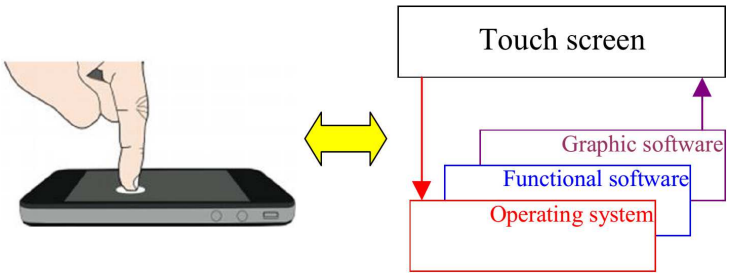


Figure 1: Direct interaction diagram (Ushirobira et al. (2016)).

several dynamic models of human movements, *e.g.* the open-loop *minimum-jerk* optimal model used by Lank et al. (2007), and the recently proposed dynamic models for indirect human-computer interaction (Aranovskiy et al. (2016); Varnell et al. (2016)). However, Aranovskiy et al. (2016) and Varnell et al. (2016) study the problem of modeling of human movements for a specific type of human-computer interaction, namely pointing motions in indirect interaction, and do not propose any solution for movement forecasting. Moreover, no systematic parameter identification procedure is known for these models restricting their applicability to the latency reduction problem. In contrast, this research is devoted to the problem of the user's trajectory prediction for general motions in direct human-computer interaction and it is not based on a underlying dynamic model.

The lack of models motivates the use of model-free prediction methods. For instance, a trajectory prediction using Kalman filter for a chain of integrators was proposed by Wu and Ouhyoung (2000), and a method based on the first-order Taylor series was used by Cattan et al. (2015), where the velocity was estimated using the two most recent position measurements. In the recent paper by Ushirobira et al. (2016) a forecasting algorithm based on the Taylor series expansion was proposed, where the derivatives were estimated using either *algebraic* (Mboup et al. (2009)), or *homogeneous* finite-time (Perruquetti et al. (2008)) differentiators. The shortcoming is that, as it was stated by Ushirobira et al. (2016), high non-linearity and interrelations between differentiator parameters make the tuning of these algorithms rather complicated. Also, some model-free approaches motivated by Kalman filter, curve fitting, and heuristic considerations can be found in the patents Moussavi (2014), Qingkui (2014), and Kim and Lim (2014), respectively.

To summarize, the problem of latency reduction can be translated to the

problem of estimating the user's trajectory or, equivalently, to the problem of predicting further points on this trajectory. To this end, the state of the art is to apply a model-free estimation/prediction algorithm, which performs a prediction based on the past measurements. In this paper, a novel model-free frequency-domain based approach for human movements forecasting is proposed. It can be shown that the ideal predictor is a *non-causal* linear time-invariant operator, which obviously cannot be implemented. However, assuming that human movements can be sufficiently well approached with a finite number of low-frequency components, the implementable forecasting algorithm is constructed as a *causal* low-frequency approximation of the ideal one. As it is shown in this paper, such an approximation can be either computed analytically with *a priori* knowledge about movement characteristics or obtained numerically as an optimization task that can be efficiently solved given a sufficiently rich dataset. Next, an adaptive modification of the design is proposed, which is intended to adapt to the changes of users and/or movement types. Experimental results illustrate the applicability of the proposed forecasting algorithm and its performance compared to other methods.

Preliminary results of this study were reported in Aranovskiy et al. (2017b). Extending that work, this manuscript contains i) the proof of the forecasting accuracy bound (15), see Proposition 1, ii) the detailed description of the experimental setup and iii) comparison to the methods currently available in patents Moussavi (2014) and Qingkui (2014).

The rest of the paper is organized as follows. First, the problem statement is given in Section 2. Next, the main forecasting algorithm is proposed in Section 3, and its adaptive version is considered in Section 4. Experimental results are described in Section 5, and concluding remarks are given in Section 6.

2. Problem statement

Consider a movement along a single axis. Motivated by physical reasoning, it is assumed that the trajectory $x(t)$ is smooth and bounded. The raw measurement $x(t_k)$ is the coordinate x measured at the time instance t_k for an integer $k \geq 0$. Given the latency value $t_L > 0$, the goal is to construct the forecasting algorithm

$$\hat{x}(t_k + t_L) = \mathcal{F}\left(\bar{\mathbf{x}}(t_k), t_k\right),$$

where $\bar{\mathbf{x}}(t_k) := \left[x(t_k), x(t_{k-1}), \ldots, x(t_0)\right]^\top$, and $\hat{x}(t_k + t_L)$ is the estimate of $x(t_k + t_L)$. In other words, given the history $\bar{\mathbf{x}}(t_k)$ up to the time instant t_k, the goal is to forecast the *future* value $x(t_k + t_L)$. The forecasting algorithm $\mathcal{F}(\cdot)$ should be constructed in such a way that the forecasting error $e_x := \hat{x} - x$ is small in the sense of a certain norm.

It is also assumed that the coordinate x is measured with the constant sampling interval T_s and $t_k = kT_s$. Moreover, the latency value t_L is the integer number $L > 0$ of sampling intervals, $t_L = LT_s$, and $t_k + t_L = t_{k+L}$.

3. The forecasting algorithm

3.1. An ideal predictor approximation

The ideal predictor in discrete-time domain is the L steps ahead time shift, *i.e.* $\mathcal{F}(\bar{\mathbf{x}}, t_k) = x(t_{k+L}) = q^L x(t_k)$, where q is the one step forward time shift operator. The ideal predictor has the following linear transfer function

$$F_{id}(z) = z^L,$$

where z is a complex variable. In the frequency domain, one obtains

$$F_{id}(\mathrm{i}\omega) = e^{\mathrm{i}\omega t_L},$$

where i is the imaginary unit and ω is the frequency (Goodwin et al. (2001)). It follows

$$|F_{id}(\mathrm{i}\omega)| \equiv 1,$$
$$\arg F_{id}(\mathrm{i}\omega) = \omega t_L.$$

Obviously, such a transfer function is not causal and cannot be implemented. However, this concept of the ideal predictor motivates the construction of the forecasting algorithm as an approximation of the ideal one in the frequency domain. Due to the assumptions on the trajectory $x(t)$, for any time instant t_k and any window size T_W, with $t_k > T_W > 0$, one can represent $x(t)$ for $t \in [t_k - T_W,\ t_k]$ as

$$x(t) = \sum_{i=0}^{\infty} A_i \sin(\omega_i t) + B_i \cos(\omega_i t), \tag{1}$$

where A_i and B_i are some constants and

$$\omega_i = \frac{2\pi i}{T_W}.$$

Next assume that there exists a known value $\omega^\dagger$, such that any reasonable trajectory generated by a user in human-computer interaction can be sufficiently well approximated with the first $N^\dagger + 1$ elements of the series expansion (1), where $N^\dagger$ is chosen such that $\omega_{N^\dagger} \leq \omega^\dagger$ and $\omega_{N^\dagger+1} > \omega^\dagger$; this approximation is further denoted as $x^\dagger$,

$$x^\dagger(t) = B_0 + \sum_{i=1}^{N^\dagger} A_i \sin(\omega_i t) + B_i \cos(\omega_i t). \tag{2}$$

To support this assumption, note that a user cannot provide infinite accelerations or decelerations to his hand (wrist). Consider a curve plotted by the user in the $x-y$ plane without lifting up the stylus. Even if this curve is not smooth, the trajectory $x(t)$ is necessary continuous and sufficiently smooth.

Then the forecast $\hat{x}^\dagger(t_k + t_L)$ of this approximation can be used as the forecast of the coordinate $x(t_k + t_L)$. The rest of the expansion (1)

$$\delta_x(t) := x(t) - x^\dagger(t) = \sum_{i=N^\dagger+1}^{\infty} A_i \sin(\omega_i t) + B_i \cos(\omega_i t)$$

is bounded and represents both negligible high-frequency trajectory deviations and possible measurement noise (Goodwin et al. (2001)). Assuming that the coefficients of the approximation (2) are slow-varying during the latency interval t_L, *i.e.* almost constant, the forecast $\hat{x}^\dagger$ of the approximation $x^\dagger$ is given by

$$\hat{x}^\dagger(t_k + t_L) = B_0 + \sum_{i=1}^{N^\dagger} A_i \sin(\omega_i(t_k + t_L)) + B_i \cos(\omega_i(t_k + t_L)).$$

This forecast can be constructed by applying a linear operator F_{ap} to the signal $x^\dagger$, where F_{ap} satisfies

$$F_{ap}(\mathrm{i}\omega_i) = F_{id}(\mathrm{i}\omega_i), \quad i \in \{0, 1, \ldots, N^\dagger\}. \tag{3}$$

The operator F_{ap} is thus an approximation of the ideal predictor F_{id}: its frequency response coincides with the frequency response of F_{id} for frequencies $\omega_i \leq \omega^\dagger$, and it does not necessary coincide for others. However, since the approximation $x^\dagger$ is not known, *i.e.* it is not measured, the operator F_{ap} is

applied to the measured signal x, and the frequency-domain approximation-based forecasting algorithm takes the form

$$\hat{x}(t_k + t_L) = F_{ap}[x(t)] = \hat{x}^\dagger(t_k + t_L) + F_{ap}[\delta_x(t)], \tag{4}$$

where $F_{ap}[(\cdot)(t)]$ denotes the operator F_{ap} applied to the signal $(\cdot)(t)$. The term $F_{ap}[\delta_x(t)]$ in the right-hand side of (4) represents the high-frequency distorting components and its impact should be minimized when possible.

3.2. Approximation using a finite impulse response filter

For any fixed $\omega^\dagger$ the operator F_{ap} can be chosen as a causal stable linear time-invariant system of order $N \geq 2N^\dagger$. One particular choice widely accepted in digital signal processing is a Finite Impulse Response (FIR) filter

$$F_{ap}(z) = c_0 + c_1 z^{-1} + \ldots + c_N z^{-N} = \sum_{i=0}^{N} c_i z^{-i}$$

yielding the forecasting algorithm

$$\hat{x}(t_k + t_L) = c_0 x(t_k) + c_1 x(t_{k-1}) + \ldots + c_N x(t_{k-N}).$$

Given $N^\dagger + 1$ distinct frequencies ω_i, $i = 0, 1, \ldots, N^\dagger$, where

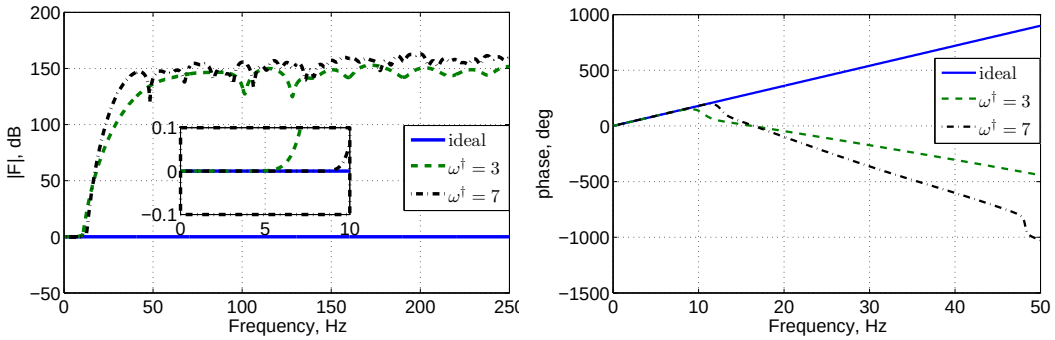
$$\omega_{N^\dagger} < \frac{\pi}{T_s}, \tag{5}$$

there always exists a unique FIR filter F_{ap} of order $N = 2N^\dagger$ satisfying (3), which can be constructed by solving a system of the following $N + 1$ linear equations:

$$\begin{cases} \mathbf{1}_{N+1}^\top \bar{\mathbf{c}} = 1, \\ \mathrm{Re}\left(\mathbf{\Omega}\right) \bar{\mathbf{c}} = \mathrm{Re}\left(\bar{\mathbf{f}}_{id}\right), \\ \mathrm{Im}\left(\mathbf{\Omega}\right) \bar{\mathbf{c}} = \mathrm{Im}\left(\bar{\mathbf{f}}_{id}\right), \end{cases} \tag{6}$$

where $\bar{\mathbf{c}} := \begin{bmatrix} c_0 & c_1 & \ldots & c_N \end{bmatrix}^\top$, $\mathrm{Re}(\cdot)$ and $\mathrm{Im}(\cdot)$ are the real and the imaginary parts of a complex value $(\cdot)$, respectively, $\mathbf{1}_{N+1}$ is the vector of 1's of dimension $N + 1$, $\bar{\mathbf{f}}_{id}$ is a $N^\dagger$-dimensional vector with complex entries,

$$\bar{\mathbf{f}}_{id} := \begin{bmatrix} e^{\mathrm{i}\omega_1 t_L} & e^{\mathrm{i}\omega_2 t_L} & \ldots & e^{\mathrm{i}\omega_{N^\dagger} t_L} \end{bmatrix}^\top,$$



(a) Magnitude (in dB) versus frequency (in Hz).

(b) Phase (in deg) versus frequency (in Hz).

Figure 2: Comparison of the ideal predictor and its approximations given by FIR filters for $\omega^\dagger = 3$Hz and $\omega^\dagger = 7$Hz

and a $N^\dagger \times (N+1)$ matrix with complex entries

$$\mathbf{\Omega} := \begin{bmatrix} 1 & e^{-\mathrm{i}\omega_1 T_s} & e^{-\mathrm{i}\omega_1 2T_s} & \dots & e^{-\mathrm{i}\omega_1 NT_s} \\ 1 & e^{-\mathrm{i}\omega_2 T_s} & e^{-\mathrm{i}\omega_2 2T_s} & \dots & e^{-\mathrm{i}\omega_2 NT_s} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & e^{-\mathrm{i}\omega_{N^\dagger} T_s} & e^{-\mathrm{i}\omega_{N^\dagger} 2T_s} & \dots & e^{-\mathrm{i}\omega_{N^\dagger} NT_s} \end{bmatrix}$$

is the complex generalization of the Vandermonde matrix. It can be shown (see the proof of Theorem 1 in Aranovskiy and Freidovich (2013)) that the system (6) has a unique solution if for all $i \in \{1, \dots, N^\dagger\}$ it holds $0 < \omega_i T_s < \pi$. The latter is satisfied due to (5).

An example of frequency responses of such approximating FIR filters with $N = 2N^\dagger$ for $T_W = 2$s, $L = 50$ and $T_s = 1$ms is given in Fig. 2 for $\omega^\dagger = 3$Hz and $\omega^\dagger = 7$Hz.

As it can be seen from Fig. 2, the filter computed in this way has significant amplification for high frequencies, and it increases impact of the $F_{ap}[\delta_x(t)]$ term, which can lead to unacceptable fast oscillations in the forecast $\hat{x}$, known as *jitter*. One possible way to overcome this drawback is to over-parametrize the FIR filter, *i.e.* to choose $N > 2N^\dagger$. Obviously, in such a case there exists an infinite number of FIR filters satisfying the condition (3), and the predictor can be constructed via optimization procedure. For example, if no *a priori* information about δ_x is known, then it is reasonable to choose $\bar{\mathbf{c}}$ as

$$\bar{\mathbf{c}} = \arg\min_{\bar{\mathbf{c}}} \|F_{ap}(z)\|_\infty \text{ subject to (3).}$$

Otherwise, if some information about δ_x is available, *e.g.* its frequency range, the cost function may be modified to take this information into account.

3.3. Data-based filter tuning

The tuning procedure proposed in Subsection 3.2 requires *a priori* knowledge of $\omega^\dagger$, as well as δ_x frequency range, if available. However, these parameters are not easy to obtain: they might depend on a specific input device and thus are hardware-dependent. That makes the analytical tuning questionable and gives rise to the experiment-based tuning approach.

Assume that a sufficiently large trajectory record $\mathcal{R}$ is available representing all kinds of possible/admissible movements. The record $\mathcal{R}$ consists of N_R measurements

$$\mathcal{R} := \{x(t_k),\ k = 1, 2, \ldots, N_R\}.$$

Given the FIR filter order N the forecasting error for $k > L+N$ is computed as

$$\begin{aligned} e_x(t_k) = \hat{x}(t_k) - x(t_k) = c_0 x(t_{k-L}) + c_1 x(t_{k-L-1}) \\ + \ldots + c_N x(t_{k-L-N}) - x(t_k) = \\ - x(t_k) + \sum_{i=0}^{N} c_i x(t_{k-L-i}). \end{aligned}$$

Define the vector of the forecasting errors as

$$\bar{\mathbf{e}}_x := \begin{bmatrix} e_x(t_{N_R}) & e_x(t_{N_R-1}) & \ldots & e_x(t_{L+N+1}) \end{bmatrix}^\top.$$

Then the FIR filter coefficients $\bar{\mathbf{c}}$ can be tuned in order to minimize a certain cost function $J(\bar{\mathbf{e}}_x)$. Particularly, if the cost function is chosen as a vector norm

$$J(\bar{\mathbf{e}}_x) = \|\bar{\mathbf{e}}_x\|_p,\ p = 1, 2, \infty,$$

then the tuning procedure is translated to the linear programming problem for $p = 1, \infty$ or to the least squares problem for $p = 2$, which can be efficiently solved even for a relatively large record size N_R (Sturm (1999); Currie et al. (2012)).

It is expected that the operator F_{ap} tuned with a sufficiently large record $\mathcal{R}$ represents the best possible approximation of the ideal predictor F_{id} with respect to the $\omega^\dagger$ and δ_x information explicitly contained in the record.

4. Adaptive forecast

The approach considered in Subsections 3.2 and 3.3 proposes to design a single fixed-gain FIR filter and to use it for forecasting regardless the exact movement performed by a user. It is assumed that this fixed FIR filter is able to forecast sufficiently well for *all* possible trajectories and for *all* users. On the other hand, it is reasonable to assume that the $\omega^\dagger$ parameter and the characteristics of $\delta_x(t)$ can vary for different users and/or for different kinds of movements, *e.g.* in drawing or pointing tasks. In this case, the FIR filter tuned for a specific user and/or movement can provide a better forecast accuracy for these specific conditions than a general fixed filter. In this section, on-line tuning of the FIR filter parameters is proposed in order to adapt to the exact user and/or to the specific movement being executed at the moment.

4.1. Recursive least-squares algorithm

To this end, rewrite the forecasting algorithm as

$$\hat{x}(t_{k+L}) = \boldsymbol{\phi}^\top(t_k)\bar{\mathbf{c}}(t_k), \tag{7}$$

where

$$\boldsymbol{\phi}(t_k) := \begin{bmatrix} x(t_k) & x(t_{k-1}) & \dots & x(t_{k-N}) \end{bmatrix}^\top \tag{8}$$

and $\bar{\mathbf{c}}(t_k)$ is the vector of FIR filter coefficients updated at the each step. The goal is to update the coefficients $\bar{\mathbf{c}}(t_k)$ in order to minimize the cost function

$$J(t_k) = \sum_{i=L+N+1}^{k} \lambda^{k-i} \left(x(t_i) - \boldsymbol{\phi}^\top(t_{i-L})\bar{\mathbf{c}}(t_k) \right)^2,$$

where $0 < \lambda \leq 1$ is the forgetting factor, which provides higher weight to recent measurements.

The desired update law is given with the recursive least-squares algorithm (Aström and Wittenmark (1994)):

$$\begin{aligned} \mathbf{g}(t_k) &= \frac{\mathbf{P}(t_{k-1})\boldsymbol{\phi}(t_{k-L})}{\lambda + \boldsymbol{\phi}^\top(t_{k-L})\mathbf{P}(t_{k-1})\boldsymbol{\phi}(t_{k-L})}, \\ \mathbf{P}(t_k) &= \frac{1}{\lambda}\left(\mathbf{P}(t_{k-1}) - \mathbf{g}(t_k)\boldsymbol{\phi}^\top(t_{k-L})\mathbf{P}(t_{k-1}) \right), \\ \bar{\mathbf{c}}(t_k) &= \bar{\mathbf{c}}(t_{k-1}) + \mathbf{g}(t_k)\left(x(t_k) - \boldsymbol{\phi}^\top(t_{k-L})\bar{\mathbf{c}}(t_{k-1}) \right). \end{aligned} \tag{9}$$

At each step k the new measurement $x(t_k)$ is available and the gains $\bar{\mathbf{c}}$ are updated using the earlier measurements $\boldsymbol{\phi}(t_{k-L})$. The updated gains are further used in the forecasting algorithm (7). The drawback of this approach is that the most recent measurements $x(t_{k-1}), \ldots, x(t_{k-L+1})$ are not actually used in the gains update. In other words, if any changes in the movement have occurred, they do not impact the gains for the next L steps.

4.2. Adaptive predictor

To take into account the most recent measurements, the following procedure is proposed. First, the one-step-ahead adaptive predictor is constructed. Next, the gains of this predictor are used to compute the gains $\bar{\mathbf{c}}$. The one-step-ahead predictor is given by

$$\hat{x}(t_{k+1}) = \boldsymbol{\phi}^\top(t_k)\boldsymbol{\theta}(t_k), \tag{10}$$

where $\boldsymbol{\theta} \in \mathbb{R}^{N+1}$ is the gains vector. The gains $\boldsymbol{\theta}$ can be adaptively tuned using the recursive least square algorithm (9) and substituting $L = 1$ and $\boldsymbol{\theta}$ in the place of $\bar{\mathbf{c}}$.

Next, to compute $\bar{\mathbf{c}}(t_k)$ it is assumed that $\boldsymbol{\theta}(t_k)$ is slow-varying for a time window of at least L steps ahead, and the trajectory x can be approximated on this window with the following *autoregressive* model with time-varying gains

$$x(t_{k+1}) = \boldsymbol{\phi}^\top(t_k)\boldsymbol{\theta}(t_k), \tag{11}$$

where for all $i \in \{0, 1, \ldots, L\}$ it holds

$$\|\boldsymbol{\theta}(t_{k+i}) - \boldsymbol{\theta}(t_k)\| \leq \epsilon_\theta, \tag{12}$$

and $\|\cdot\|$ is the Euclidean norm.

Proposition 1. *Given the autoregressive model* (11) *under assumption* (12)*, set the vector* $\bar{\mathbf{c}}(t_k)$ *as*

$$\bar{\mathbf{c}}(t_k) = \begin{bmatrix}\boldsymbol{\theta}(t_k) & \mathbf{e}_1 & \ldots & \mathbf{e}_N\end{bmatrix}^L \mathbf{e}_1, \tag{13}$$

where $\mathbf{e}_i$ *denotes the* i*-th Euclidean basis vector,* i.e. *the* $N+1$ *dimensional vector with a* 1 *in the* i*-th coordinate and* 0*'s elsewhere. Then it holds*

$$x(t_{k+L}) = \boldsymbol{\phi}^\top(t_k)\bar{\mathbf{c}}(t_k) + \sigma_x(t_k), \tag{14}$$

and

$$|\sigma_x(t_k)| \leq \sqrt{N+1} x_\infty H(\epsilon_\theta, \|\boldsymbol{\theta}(t_k)\|)\epsilon_\theta, \tag{15}$$

where the hardware-dependent value x_∞ is the maximum admissible position value, the function

$$H(\epsilon_\theta, \|\boldsymbol{\theta}(t_k)\|) := \sum_{i=1}^{L-1} \binom{L-1}{i} \epsilon_\theta^{i-1} \left(\|\boldsymbol{\theta}(t_k)\| + 1\right)^{L-i}, \tag{16}$$

and $\binom{n}{k}$ is the binomial coefficient.

Proof. Define the d-steps-ahead shift vector $\boldsymbol{\beta}(d, t_k) \in \mathbb{R}^{N+1}$ such that for the autoregressive model (11) the following holds

$$x(t_{k+d}) = \boldsymbol{\phi}^\top(t_k)\boldsymbol{\beta}(d, t_k). \tag{17}$$

Obviously, $\boldsymbol{\beta}(1, t_k) = \boldsymbol{\theta}(t_k)$ and $\bar{\mathbf{c}}(t_k) = \boldsymbol{\beta}(L, t_k)$. Moreover, for all $k > N$ it holds $\boldsymbol{\beta}(0, t_k) = \mathbf{e}_1$, $\boldsymbol{\beta}(-1, t_k) = \mathbf{e}_2$, and so on up to $\boldsymbol{\beta}(-N, t_k) = \mathbf{e}_{N+1}$. Next define a $(N+1) \times (N+1)$ matrix $\mathfrak{B}(d, t_k)$ as

$$\mathfrak{B}(d, t_k) := \begin{bmatrix} \boldsymbol{\beta}(d, t_k) & \boldsymbol{\beta}(d-1, t_k) & \dots & \boldsymbol{\beta}(d-N, t_k) \end{bmatrix},$$

where by definition

$$\mathfrak{B}(1, t_k) = \begin{bmatrix} \boldsymbol{\beta}(1, t_k) & \mathbf{e}_1 & \dots & \mathbf{e}_N \end{bmatrix}. \tag{18}$$

Recalling (8) it can be verified that

$$\begin{aligned} x(t_{k+d+1}) &= \boldsymbol{\phi}^\top(t_{k+d})\boldsymbol{\beta}(1, t_{k+d}) \\ &= \begin{bmatrix} x(t_{k+d}) & x(t_{k+d-1}) & \dots & x(t_{k+d-N}) \end{bmatrix} \boldsymbol{\beta}(1, t_{k+d}) \\ &= \boldsymbol{\phi}^\top(t_k) \begin{bmatrix} \boldsymbol{\beta}(d, t_k) & \dots & \boldsymbol{\beta}(d-N, t_k) \end{bmatrix} \boldsymbol{\beta}(1, t_{k+d}) \\ &= \boldsymbol{\phi}^\top(t_k)\mathfrak{B}(d, t_k)\boldsymbol{\beta}(1, t_{k+d}). \end{aligned}$$

On the other hand, from (17) it follows

$$x(t_{k+d+1}) = \boldsymbol{\phi}^\top(t_k)\boldsymbol{\beta}(d+1, t_k),$$

and thus

$$\boldsymbol{\beta}(d+1, t_k) = \mathfrak{B}(d, t_k)\boldsymbol{\beta}(1, t_{k+d}). \tag{19}$$

Note that by definition the first column of $\mathfrak{B}(d+1,t_k)$ is $\boldsymbol{\beta}(d+1,t_k)$, and the i-th column of $\mathfrak{B}(d+1,t_k)$ coincides with the $(i-1)$-th column of $\mathfrak{B}(d,t_k)$ for i varying from 2 to $N+1$. Then from (19) and recalling (18) one obtains

$$\mathfrak{B}(d+1,t_k)=\mathfrak{B}(d,t_k)\mathfrak{B}(1,t_{k+d}).$$

The latter implies

$$\begin{aligned}
\mathfrak{B}(2,t_k)&=\mathfrak{B}(1,t_k)\mathfrak{B}(1,t_{k+1}),\\
\mathfrak{B}(3,t_k)&=\mathfrak{B}(1,t_k)\mathfrak{B}(1,t_{k+1})\mathfrak{B}(1,t_{k+2}),\\
\mathfrak{B}(4,t_k)&=\mathfrak{B}(1,t_k)\mathfrak{B}(1,t_{k+1})\mathfrak{B}(1,t_{k+2})\mathfrak{B}(1,t_{k+3}),
\end{aligned}$$

and

$$\mathfrak{B}(d,t_k)=\prod_{i=0}^{d-1}\mathfrak{B}(1,t_{k+i}). \tag{20}$$

Therefore

$$x(t_{k+d})=\boldsymbol{\phi}^\top(t_k)\mathfrak{B}(d,t_k)\mathbf{e}_1=\boldsymbol{\phi}^\top(t_k)\boldsymbol{\beta}(d,t_k).$$

However, at the step k the only available value is $\boldsymbol{\theta}(t_k)$, that gives $\boldsymbol{\beta}(1,t_k)$ and $\mathfrak{B}(1,t_k)$, but not $\mathfrak{B}(d,t_k)$. With the definitions in use, (13) can be rewritten as

$$\bar{\mathbf{c}}(t_k)=\left(\mathfrak{B}(1,t_k)\right)^L\mathbf{e}_1,$$

and $\sigma_x(t_k)$ in (14) is thus

$$\begin{aligned}
\sigma_x(t_k)&=x(t_{k+L})-\boldsymbol{\phi}^\top(t_k)\left(\mathfrak{B}(1,t_k)\right)^L\mathbf{e}_1\\
&=\boldsymbol{\phi}^\top(t_k)\left(\mathfrak{B}(L,t_k)-\left(\mathfrak{B}(1,t_k)\right)^L\right)\mathbf{e}_1.
\end{aligned}$$

Define

$$\Delta\mathfrak{B}(d):=\mathfrak{B}(1,t_{k+d})-\mathfrak{B}(1,t_k)$$

and note that

$$\begin{aligned}
\|\mathfrak{B}(1,t_k)\|&=\sup_{\|\mathbf{x}\|=1}\|\mathfrak{B}(1,t_k)\mathbf{x}\|=\sup_{\|\mathbf{x}\|=1}\left\|\begin{bmatrix}\boldsymbol{\theta}(t_k) & \mathbf{e}_1 & \dots & \mathbf{e}_N\end{bmatrix}\mathbf{x}\right\|\\
&=\sup_{\|\mathbf{x}\|=1}\left\|\boldsymbol{\theta}(t_k)x_1+\begin{bmatrix}x_2 & x_3 & \dots & x_{N+1} & 0\end{bmatrix}^\top\right\|\leq\|\boldsymbol{\theta}(t_k)\|+1,
\end{aligned}$$

and for all $i\in\{0,1,\dots,L\}$

$$\|\Delta\mathfrak{B}(i)\|\leq\epsilon_\theta.$$

From (20) it follows

$$\mathfrak{B}(L,t_k) = \mathfrak{B}(1,t_k)\prod_{i=1}^{L-1}\left(\mathfrak{B}(1,t_k)+\Delta\mathfrak{B}(i)\right)$$

and

$$\begin{aligned}&\mathfrak{B}(L,t_k) - \left(\mathfrak{B}(1,t_k)\right)^L\\ &\quad= \mathfrak{B}(1,t_k)\left(\prod_{i=1}^{L-1}\left(\mathfrak{B}(1,t_k)+\Delta\mathfrak{B}(i)\right) - \left(\mathfrak{B}(1,t_k)\right)^{L-1}\right).\end{aligned}$$

Therefore

$$\begin{aligned}&\|\mathfrak{B}(L,t_k) - \left(\mathfrak{B}(1,t_k)\right)^L\|\\ &\quad\le \left(\|\boldsymbol{\theta}(t_k)\|+1\right)\left(\sum_{i=1}^{L-1}\binom{L-1}{i}\epsilon_\theta^i\left(\|\boldsymbol{\theta}(t_k)\|+1\right)^{L-i-1}\right)\\ &\quad= \epsilon_\theta H(\epsilon_\theta, \|\boldsymbol{\theta}(t_k)\|),\end{aligned}$$

where $H(\cdot,\cdot)$ is defined in (16). Noting that

$$\|\boldsymbol{\phi}(t_k)\| \le \sqrt{N+1}x_\infty,$$

the upper bound (15) follows. □

Remark 1. *The upper bound* (15) *allows to establish some qualitative conclusions, e.g. a relation between the upper bound and the maximum admissible position* x_∞*. One particularly important observation is that if* $\boldsymbol{\theta}(k)$ *remains constant and* ϵ_θ *equals zero, then* $|\sigma_x(k)|$ *is zero as well.*

Finally, the adaptive forecasting algorithm that uses the most recent measurements can be formulated as follows.

Algorithm 1:

Step 1. At the k-th step get the new measurement $x(t_k)$.

Step 2. Update the one-step-ahead predictor gains $\boldsymbol{\theta}(t_k)$ using algorithm (9) with $L=1$ and substituting $\boldsymbol{\theta}$ in the place of $\bar{\mathbf{c}}$.

Step 3. Compute $\bar{\mathbf{c}}(t_k)$ as (13) using the one-step-ahead predictor gains $\boldsymbol{\theta}(t_k)$.

Step 4. Compute forecast $\hat{x}(t_{k+L})$ as (7).

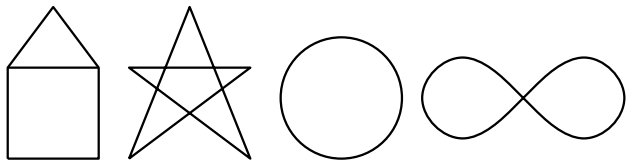


Figure 3: Geometric shapes used in the experiment.

5. Experimental results

An experiment where three participants were asked to draw some predefined shapes on a screen using a stylus is considered. The shapes were divided into three groups: a) the geometric shapes given in Fig. 3, b) the Latin capital letters from A to F, and c) the digits from 0 to 9.

The hardware used in the experiment is the Touch X by 3D Systems (2019), previously known as Phantom Desktop, which is a pen-type haptic device. The Touch X is coupled with a horizontally oriented 15" display and calibrated to get the stylus tip position co-located with the display, using an optimal affine transformation between two 3D point sets using the RANdom SAmple Consensus (RANSAC) algorithm (Fischler and Bolles (1981)). Thus, this experimental setup mimics a touchscreen with a pen.

The measurements were captured with the sampling interval $T_s = 1$ms, and the expected latency value $t_L = 50$ ms, that is $L = 50$. The movements were performed in the plane, and the x and y axes are considered separately (human movements along different axes typically have different magnitudes and velocities, and the predictors for x and y may have different order N and operate over time windows of different length L).

This section presents a comparison of online forecasting algorithms which do not require a rich dataset to tune them. These algorithms are motivated by the patents Moussavi (2014) and Qingkui (2014) and are based on Kalman filtering and curve fitting. The 3-rd order Kalman filter is chosen, where the process and measurement model matrices are given by Moussavi (2014). For the curve fitting forecast, the 2-nd order polynomial fitted with the last 30 samples is used. These two predictors are compared with the adaptive FIR filter given by *Algorithm 1.* The FIR filters is of order $N = 15$, that implies 16 tunable gains. The forgetting factor is a tuning parameter and is chosen such that it provides the discounting factor $\frac{1}{2}$ for the measurements obtained one second ago yielding

$$\lambda^{\frac{1}{T_s}} = 2 \Leftrightarrow \lambda = \exp(-T_s \log(2)) \approx 0.9993.$$

The performance indices used for comparison are:

- the mean absolute error MAE computed as

$$MAE := \frac{1}{N_{samples}} \sum_{k=1}^{N_{samples}} |e_x(t_k)|,$$

  where $N_{samples}$ is the number of samples used for performance evaluation;

- the mean squared error MSE computed as

$$MSE := \frac{1}{N_{samples}} \sum_{k=1}^{N_{samples}} e_x^2(t_k);$$

- the maximum error ME computed as

$$ME := \max_{k=1,\ldots,N_{samples}} |e_x(t_k)|.$$

Note that MAE and ME are measured in pixels, while MSE is measured in pixels^2.

Comparison results divided by figure groups are given in Table 1, and the same performance indices computed for the whole dataset are given in Table 2. It can be seen from the experimental results that the proposed adaptive FIR prediction algorithm provides the most accurate prediction in the terms of MAE, MSE and ME. The only case when the adaptive FIR has the ME value slightly higher than the competitors is the Latin Letters scenario and probably this can be related to transients in adaptive gains tuning. The Kalman filter predictor has the second best accuracy in terms of MAE and MSE. The ME value of the Kalman filter predictor is, in general, higher than the one of the curve fitting predictor, which, however, has the poorest MAE and MSE accuracy.

To evaluate how does prediction error depend on the movement velocity, the numerical differentiation was performed as

$$v(t_k) = \frac{x(t_{k+d}) - x(t_{k-d})}{2dT_s},$$

Table 1: Prediction performance comparison divided by figure groups.

	MAE pixel	MSE pixel^2	ME pixel
	Geometric shapes		
Kalman filter	2.05	9.89	56.94
Curve fitting	2.27	11.44	50.09
Algorithm 1	1.97	8.22	46.82
	Latin letters		
Kalman filter	2.84	16.22	36.12
Curve fitting	3.21	20.09	37.21
Algorithm 1	2.33	10.48	39.76
	Digits		
Kalman filter	3.13	18.56	34.97
Curve fitting	3.63	24.71	34.12
Algorithm 1	2.32	10.13	25.77

Table 2: Prediction performance comparison for the whole dataset.

	MAE pixel	MSE pixel^2	ME pixel
Kalman filter	2.70	15.15	56.94
Curve fitting	3.08	19.17	50.09
Algorithm 1	2.21	9.66	46.82

where $v(t_k)$ is the velocity estimate, and the differentiation window length $d = 5$. It was found that the majority of the movements is performed with the absolute velocity in the range form zero to 1200 pixels per second. This range is uniformly divided into 20 intervals, and for each interval the mean absolute error (MAE) is computed for the adaptive FIR, Kalman filter, and curve fitting predictors. The resulting plot is given in Figure 4. As it can be seen, the adaptive predictor outperforms it competitors in the considered velocity range.

Remark 2. *As it is discussed in Section 3.3, the fixed-gains FIR filter requires sufficiently rich records to be tuned off-line, and thus it is not included in comparison. However, the frequency response of the fixed-gains FIR filter optimized over the whole dataset is given as an illustration in Fig. 5. The phase response shows that the* $\omega^\dagger$ *value approximated from the given experi-*

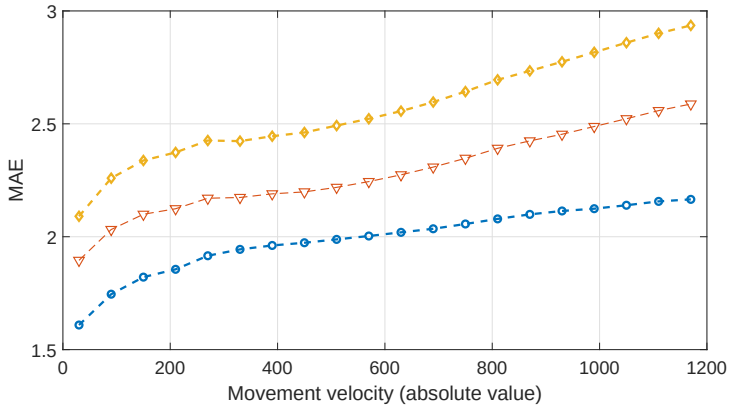


Figure 4: MAE (in pixels) versus the movement velocity (in pixels per second).

mental data is close to 5 *Hz.*

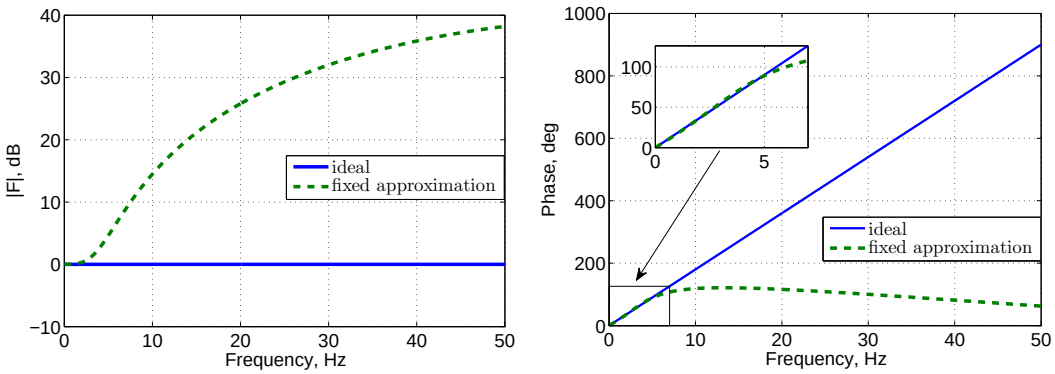
6. Conclusion

The problem of latency compensation in direct human-computer interactions was considered in this paper. This problem can be formulated as a forecasting algorithm design, and a novel approach was proposed, based on a frequency-domain approximation of the ideal predictor. Such an approximation can be either computed analytically or constructed by solving optimization problem for a sufficiently rich dataset. Also, an adaptive modification of the forecasting algorithm was developed allowing possible variations of the user behavior to be taken into account. Experimental studies illustrate the applicability of the proposed solution for a trajectory forecasting with reasonable accuracy.

One possible direction of further studies is to apply for adaptive parameters tuning some techniques with enhanced transient performance, *e.g.* the one recently proposed by Aranovskiy et al. (2017a). Hopefully, this modification can reduce the maximum prediction error value ME. Another challenging problem is to consider model-based trajectory prediction algorithms that involve the design of user behavior models.

7. Acknowledgments

This work was supported by ANR (TurboTouch, ANR-14-CE24-0009). Stanislav Aranovskiy is supported by the Russian Science Foundation grant



(a) Magnitude (in dB) versus frequency (in Hz).

(b) Phase (in deg) versus frequency (in Hz).

Figure 5: Comparison of the ideal predictor and its approximations given by fixed-gains FIR filter tuned with $N = 15$ using the whole dataset

(project 17-79-20341).

8. Conflict of interest

None declared.

9. References

References

3D Systems, 2019. Touch X specification.
URL `https://www.3dsystems.com/haptics-devices/touch-x`

Aranovskiy, S., Bobtsov, A., Ortega, R., Pyrkin, A., 2017a. Performance enhancement of parameter estimators via dynamic regressor extension and mixing. IEEE Transactions on Automatic Control 62 (7), 3546–3550.

Aranovskiy, S., Freidovich, L. B., 2013. Adaptive compensation of disturbances formed as sums of sinusoidal signals with application to an active vibration control benchmark. European Journal of Control 19 (4), 253–265.

Aranovskiy, S., Ushirobira, R., Efimov, D., Casiez, G., 2016. Modeling pointing tasks in mouse-based human-computer interactions. In: Decision and Control (CDC), 2016 IEEE 55th Conference on. IEEE, pp. 6595–6600.

Aranovskiy, S., Ushirobira, R., Efimov, D., Casiez, G., 2017b. Frequency domain forecasting approach for latency reduction in direct human-computer interaction. In: Conference on Decision and Control. pp. 2623–2628.

Aström, K., Wittenmark, B., 1994. Adaptive Control, 2nd Edition. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Bérard, F., Blanch, R., 2013. Two touch system latency estimators: high accuracy and low overhead. In: Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces. ACM, pp. 241–250.

Casiez, G., Conversy, S., Falce, M., Huot, S., Roussel, N., 2015. Looking through the Eye of the Mouse : A Simple Method for Measuring End-to-end Latency using an Optical Mouse. In: Proceedings of the 28th ACM Symposium on User Interface Software and Technology. ACM Press. pp. 629–636.

Casiez, G., Pietrzak, T., Marchal, D., Poulmane, S., Falce, M., Roussel, N., 2017. Characterizing latency in touch and button-equipped interactive systems. In: Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology. ACM, pp. 29–39.

Cattan, E., Rochet-Capellan, A., Perrier, P., Bérard, F., 2015. Reducing latency with a continuous prediction: Effects on users' performance in direct-touch target acquisitions. In: Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces. ACM, pp. 205–214.

Currie, J., Wilson, D. I., Sahinidis, N., Pinto, J., 2012. Opti: Lowering the barrier between open source optimizers and the industrial matlab user. Foundations of computer-aided process operations 24, 32.

Fischler, M. A., Bolles, R. C., Jun. 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24 (6), 381–395.
URL `http://doi.acm.org/10.1145/358669.358692`

Goodwin, G. C., Graebe, S. F., Salgado, M. E., 2001. Control system design. Upper Saddle River.

Jota, R., Ng, A., Dietz, P., Wigdor, D., 2013. How fast is fast enough?: a study of the effects of latency in direct-touch pointing tasks. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, pp. 2291–2300.

Kim, B., Lim, Y., Aug. 28 2014. Mobile terminal and touch coordinate predicting method thereof. WO Patent App. PCT/KR2014/000,661.

Lank, E., Cheng, Y.-C. N., Ruiz, J., 2007. Endpoint prediction using motion kinematics. In: Proceedings of the SIGCHI conference on Human factors in computing systems (CHI 07). San Jose, California, USA, pp. 637–646.

MacKenzie, I. S., Ware, C., 1993. Lag as a determinant of human performance in interactive systems. In: Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems. ACM, pp. 488–493.

Mboup, M., Join, C., Fliess, M., 2009. Numerical differentiation with annihilators in noisy environment. Numerical algorithms 50 (4), 439–467.

Moussavi, F., Jul. 1 2014. Methods and apparatus for incremental prediction of input device motion. US Patent 8,766,915.

Ng, A., Lepinski, J., Wigdor, D., Sanders, S., Dietz, P., 2012a. Designing for low-latency direct-touch input. In: Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology. ACM, pp. 453–464.

Ng, A., Lepinski, J., Wigdor, D., Sanders, S., Dietz, P., 2012b. Designing for low-latency direct-touch input. In: Proceedings of the 25th annual ACM symposium on User interface software and technology. ACM, pp. 453–464.

Perruquetti, W., Floquet, T., Moulay, E., 2008. Finite-time observers: application to secure communication. IEEE Transactions on Automatic Control 53 (1), 356–360.

Qingkui, Jul. 2 2014. Curve fitting based touch trajectory smoothing method and system. CN Patent App. CN 201,210,585,264.

Sturm, J. F., 1999. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. Optimization methods and software 11 (1-4), 625–653.

Ushirobira, R., Efimov, D., Casiez, G., Roussel, N., Perruquetti, W., June 2016. A forecasting algorithm for latency compensation in indirect human-computer interactions. In: 2016 European Control Conference (ECC). Aalborg, Denmark, pp. 1081–1086.

Varnell, P., Malisoff, M., Zhang, F., 2016. Stability and robustness analysis for human pointing motions with acceleration under feedback delays. International Journal of Robust and Nonlinear Control.

Wu, J.-R., Ouhyoung, M., 2000. On latency compensation and its effects on head-motion trajectories in virtual environments. The visual computer 16 (2), 79–90.