# Next-Point Prediction Metrics for Perceived Spatial Errors

Mathieu Nancel, Daniel Vogel, Bruno de Araujo, Ricardo Jota, Géry Casiez

# Next-Point Prediction Metrics for Perceived Spatial Errors

**Mathieu Nancel[1,2], Daniel Vogel[1], Bruno De Araujo[3,4], Ricardo Jota[4], Géry Casiez[5]**
[1]University of Waterloo, [2]Aalto University [3]University of Toronto, [4]Tactual Labs, [5]Université de Lille
{mnancel, dvogel}@uwaterloo.ca, {brar, jotacosta}@dgp.toronto.edu, gery.casiez@univ-lille1.fr

## ABSTRACT
Touch screens have a delay between user input and corresponding visual interface feedback, called input "latency" (or "lag"). Visual latency is more noticeable during continuous input actions like dragging, so methods to display feedback based on the most likely path for the next few input points have been described in research papers and patents. Designing these "next-point prediction" methods is challenging, and there have been no standard metrics to compare different approaches. We introduce metrics to quantify the probability of 7 spatial error "side-effects" caused by next-point prediction methods. Types of side-effects are derived using a thematic analysis of comments gathered in a 12 participants study covering drawing, dragging, and panning tasks using 5 state-of-the-art next-point predictors. Using experiment logs of actual and predicted input points, we develop quantitative metrics that correlate positively with the frequency of perceived side-effects. These metrics enable practitioners to compare next-point predictors using only input logs.

### Author Keywords
touch input; latency; lag; prediction;

### ACM Classification Keywords
H.5.2 User Interfaces: Input devices and strategies

## INTRODUCTION
Most interactive systems have a delay between user input and corresponding interface feedback, referred to as input "latency" (or "lag") [26]. End-to-end latency is the total time required by dependent subsystems (e.g. sensing, recognition, stored memory, rendering) to convert an input action into application commands with user feedback [12, 29]. Current touch devices have end-to-end latencies between 60 and 200 ms [33]. Some subsystems like touch sensors can be very fast [23], but complete suppression of latency from all sources in general purpose computing is unlikely: some feedback takes time to compute and the 15 ms per display frame is noticeable [33]. Even very small latencies are noticeable with a direct input device like a touch screen [7]. Any discrepancy between visual feedback and input is more noticeable with direct input, and even more so when continuous on-surface input [7, 33] like drawing, dragging, scrolling, and panning. In fact, studies show people detect latency as low as 2 ms [17] and latency above 10 ms degrades performance [14, 17, 33].

Low-latency visual feedback can be displayed by predicting near-future input actions. For continuous on-surface input, this means predicting the most likely path for the next few input locations. Such "next-point" prediction techniques have been the topic of research [14, 45], described in many patents [4, 5, 19, 25, 30, 40, 43, 46, 47], and implemented in some operating systems [1, 2, 39]. Ideally, input should be predicted far enough in the future to cancel out total end-to-end latency, but spatial accuracy typically degrades with greater prediction time. Next-point techniques have attempted up to 75 ms [14], but most predict only a single frame (up to $\sim$ 16 ms). Designing next-point techniques to cancel out end-to-end latency is a challenging goal requiring principled evaluation methods. Previous work compared conservative prediction techniques using metrics like root mean square error (RMSE) [22] and worst Euclidean error distance [21]. The question is whether these general-purpose metrics effectively capture the degree to which people *perceive* different spatial accuracy errors when predicting far into the future.

In this paper, we contribute metrics to calculate the magnitude of 7 classes of spatial inaccuracies caused by next-point prediction methods: *"lateness"*, *"over-anticipate"*, *"wrong distance"*, *"wrong orientation"*, *"jitter"*, *"jumps"*, and *"spring effect"*. We identified these "side-effects" using a thematic analysis of comments gathered in a 12 participant study in which they performed open-ended continuous on-surface input actions simulating typical drawing, dragging, and panning tasks. They performed these tasks with a no-prediction control, and 5 state-of-the-art predictors configured to predict 68 ms to compensate for the full end-to-end latency of our apparatus. Using experiment logs of actual and predicted input points, we developed our metrics to model the severity of different side-effects. Then, using linear regression models on aggregate experiment data, we show that these metrics accurately predict the probability of people perceiving these side-effects. In contrast, we show that current previous measures do not consistently capture all perceived side-effects.

Our work enables practitioners to design new predictors without running studies at the earliest steps of the process, to efficiently compare new and existing predictors based on actual and predicted data, and will encourage systematic benchmarking of next-point prediction techniques.

## BACKGROUND AND RELATED WORK

Using various measurement techniques [9, 13, 16, 18, 33] end-to-end latency in consumer touch screen devices has been quantified. For example, Ng et al. [33] found latencies between 50 and 200 ms, and the Agawi TouchMark latency benchmarks [16] report latencies between 72 and 168 ms. This has motivated research on the impact of latency on task performance, human perception of touch latency, and prediction techniques to compensate for latency. We review all these areas below, focusing on touch input, but with examples from other input methods when relevant.

### Impact of Latency on Task Performance

End-to-end latency has long been an important issue for immersive graphical systems like Virtual and Augmented reality, causing potential motion sickness [31] and reducing performance. For example, Ware and Balakrishnan [41] found latency greater than 100 ms affected some augmented reality task performance. Early work evaluating latency in graphical user interfaces by Miller [28] and Schneiderman [38] supported this 100 ms latency rule-of-thumb. MacKenzie and Ware [26] include latency in pointing task models, and show latencies above 75 ms have an effect. In real-time games, Pavlovych and Gutwin [35] found accuracy acquiring fast moving targets dropped significantly with more than 50 ms latency. Pavlovych and Stuerzlinger [36, 37] found latency, jitter, and drop-outs can affect performance differently.

For touch input, Anderson et al.'s [6] qualitative evaluation of high-level tasks like web browsing and ebook reading concluded that latencies above 580 ms were unacceptable. However, subjective comments miss quantitative differences. For example, Jota et al. [17] found latency greater than 25 ms significantly reduced user performance in touch dragging tasks and Cattan et al. [14] found latency greater than 25 ms increased dragging task time. This is troubling since current touch devices have much greater latency.

### Human Perception of Input Latency

Regardless of performance, increasing evidence suggests people can perceive very low latency in direct input. With touch, Jota et al. [17] found people detected latencies above 24 ms in touch tapping actions. For touch dragging tasks, where there is more time to perceive offsets between feedback and input, Ng et al. [33] found people noticed latencies greater than 5 to 10 ms. Deber et al. [15] even found relative latency changes as small as 8.3 ms are noticeable, particularly for dragging actions. With pen input, the perception thresholds can be even smaller. Ng et al. [32] report 2 ms for dragging and 6 ms for scribbling, though Annett et al. [7] found people only perceive latencies greater than 50 ms for higher-level tasks like writing and drawing.

Reducing latency below the level of perception is an important goal. This would not only mitigate effects on task performance, but increase the sense of directness [38] when using touch input. While improvements to touch sensors [23] and general processing speed will help, there will always be sources of latency that are hard to eliminate such as more complex application functionality, higher fidelity graphics, and network delays. Partly for this reason, researchers and practitioners have proposed input prediction techniques.

### Input Prediction Techniques and Metrics

In Virtual Reality, compensating for latency in head rotation input is an important goal. LaViola [22] compared double exponential, Kalman, and extended-Kalman filters for predicting 100 ms in the future. Using Root Mean Square Error (RMSE) as the comparison metric, he concluded that double exponential filter performed best. Wu et al. [44] compared a Kalman filter, linear extrapolation, and a theory-based predictor when predicting 150 ms in the future. Using average Euclidean distance error as a metric, they found the theory-based predictor and the Kalman filter were more accurate. In addition, participants ranked Kalman filtering highest even though they said it had high spatial jitter. LaValle et al. [21] compared algorithms based on constant velocity and acceleration to predict the next 20 ms of head rotation. They used two metrics, average and worst Euclidean distance errors.

RMSE and average Euclidean distance may provide an overall measure of accuracy. A worst Euclidean distance metric may capture an infrequent behaviour like jumps, but it is very sensitive. Most importantly, none of these metrics are associated with what people actually perceive. Given Wu et al.'s finding that participant's preferred Kalman in spite of high jitter, perceived errors are a critical comparison factor.

With graphical user interfaces, methods have been proposed for predicting the end point of a touch screen tap or mouse click. For indirect mouse input, the goal is not to reduce latency, but improve selection performance. Various features and models have been evaluated, such as direction and peak velocity [8], motion kinematics [20], kinematic template matching [34], and neutral networks and Kalman filters [10]. These techniques rely on monitoring the movement of the mouse cursor in between two clicks, something that is not possible with current touch input since the finger is in the air between taps. Xia et al. [45] use a motion tracking system to log intermediate finger positions in the air between taps. They were able to reduce touch latency to 0 ms by accurately predicting the end point up to 100 ms in the future.

These techniques all use a Euclidean distance error metric to compare the predicted end point and the target end point. In addition, end-point predictors leverage pointing movement theory [27], so they do not translate to continuous on-surface movements where movement direction, speed, and acceleration can change drastically without any relationship to the end point. In the following section, we survey next-point prediction methods for continuous on-surface movements.

## NEXT-POINT PREDICTION TECHNIQUES

Existing techniques for next-point prediction can be classified using their underlying principle: Taylor series, Kalman filtering, curve fitting, and heuristic approaches. Most have been published only as patents [4, 5, 19, 25, 30, 40, 43, 46, 46, 47] which can be difficult to read for the uninitiated. We use six of these representative predictors in our evaluation, so providing details with consistent terminology and notation will

decrease effort to replicate our work. For these reasons, we describe existing next-point prediction techniques in detail.

### Taylor series

Basing a predictor on the Taylor series implies trajectories can be modelled as an infinitely differentiable function. Theoretically, a point $\hat{P}(t)$ around some future event time $t_0$ may be predicted through an infinite sum of its derivatives $P^{(n)}$:

$$\hat{P}(t) = \sum_{n=0}^{\infty} \frac{P^{(n)}(t_0)}{n!}(t-t_0)^n \quad (1)$$

Without loss of generality, we assume $t_0 = 0$ so $t = t - t_0$ and $t$ becomes the amount of time into the future to predict. We use this notational simplification in all following equations.

#### *First order Taylor series*

If only the first derivative is considered, Eq. 1 becomes:

$$\hat{P}(t) \approx P(0) + P'(0)\,t \quad (2)$$

where $P(0)$ is the current finger position and $P'(0)$ is the instantaneous velocity of the finger. This prediction is used by Lincoln [25] to predict pen input 50 ms in the future, Cattan et al. [14] to predict touch points between 25 and 75 ms in the future, and by Zhao et al. [46] to predict one frame in the future (approx. 16 ms assuming 60 frames-per-second input). Only Cattan et al. report the results of an evaluation. Using a straight-line dragging docking task, they found movement time decreased when fully compensating for 25 ms latency. At 75 ms, undershooting and overshooting became noticeable (p.5), presumably reducing performance. Our experiment examines the perception of these kinds of "side-effects."

When using only the first derivative, the assumption is that the velocity remains constant over the time frame of the prediction. Using higher order derivatives relaxes this assumption.

#### *Second order Taylor series*

Adding the second order derivative, Eq. 1 becomes:

$$\hat{P}(t) \approx P(0) + P'(0)\,t + P''(0)\,\frac{t^2}{2} \quad (3)$$

where $P''(0)$ is the acceleration of the finger. Wang [40], Zhou [47], and Zhao et al. [46] use this formulation for predicting the next frame. Both Wang and Zhou determine $P'(0)$ and $P''(0)$ based on the instantaneous speed and acceleration over the last few input frames. If the prediction time is a multiple of the frame period, Eq. 3 can be written as:

$$\hat{P}(t) = 2.5\,P(0) - 2\,P(-t) + 0.5\,P(-2t) \quad (4)$$

This is Zhao et al.'s [46] formulation where $t$ is the period of a single frame (approx. 16 ms assuming 60 Hz input).

In theory, Taylor series prediction will work if the infinite sum of derivatives really models future movements and those derivatives can be estimated accurately.

### Kalman filter

Kalman filters are composed of a process model (a transition matrix between previous and current states, and a process noise covariance matrix), and a measurement model (a measurement matrix combining the information from different sensors and a measurement noise matrix). The standard Kalman filter uses a discrete-time linear stochastic equation for the process model and assumes process and measurement noises are independent, white, and normally distributed [42]. Kalman combines the raw information from the sensors and the prediction of the model to obtain the best estimation of a state taking into account model and measurement noise. If model noise exceeds measurement noise, measurement will dominate (and vice versa).

Kalman filters are excellent for prediction when equations can precisely describe a system's behaviour. There is no complete model for finger motion, so simple models based on Taylor series are used [22, 24, 44]. Moussavi [30] predicts the next touch point frame (approx. 16 ms) using a second order Taylor's series for the process model and a covariance matrix and Kalman gain measurement model. Luo et al. [4] use a semi-Kalman filter without the covariance matrix calculation to also predict the next frame.

### Curve fitting

Another approach is to fit a curve to recent touch points, and predict using extrapolation. Qingkui et al. [5] fit a polynomial to the last 50 to 60 input points, and use the curve tangent and polynomial derivative to predict the next touch point frame (approx. 16 ms). They fit polynomials with order between 2 and 7 using the least square method.

### Heuristic Approaches

Kim et al. [19] use either speed or acceleration as the primary factor for prediction. Magnitude of direction change (the angular difference between vectors formed from the new point to previous point, and the previous point to the next previous point) is a heuristic to choose from two formulas. If direction change $d$ is less than 15°, then velocity dominates, otherwise acceleration dominates. This is used to predict the next touch point frame (approx. 16 ms) with the following equation:

$$\hat{P}(t) = \begin{cases} P(0) + 2\,P'(0) + 0.5\,P''(0) & if d < 15^\circ \\ P(0) + 0.5\,P'(0) + 5\,P''(0) & otherwise \end{cases} \quad (5)$$

## EXPERIMENT

Current next-point prediction methods are not perfect, all exhibit some degree of spatial accuracy errors. Previous work evaluating and comparing predictors [14, 45] have focused on task time, but that does not capture how people perceive prediction errors. Consider Wu et al.'s [44] finding that people are less bothered by jitter in head input prediction, and Cattan et al.'s [14] observation that spatial errors like overshoots and undershoots become a problem.

Therefore, the goal of this experiment is to classify and quantify spatial accuracy prediction errors that people notice with next-point prediction for touch input. These visible errors are the "side-effects" of imperfect prediction. We use a thematic

analysis of comments from participants as they performed typical continuous on-surface touch input tasks with 5 state-of-the-art next-point prediction methods. All predictors are configured to predict 68 ms in the future to compensate for the perceivable end-to-end latency of our apparatus and to increase the frequency of observable side-effects. Using our results, we develop metrics to measure the magnitude of side-effects and estimate the probability of perceiving them.

### Participants

We recruited 12 participants: 1 left-handed, 4 female, 23 to 34 years old ($\mu$28.3 $\sigma$3.7). Six participants identified as Human Computer Interaction professionals or students (P0-P5). This group was recruited to see if "experts" were more capable of perceiving and describing side-effects. As we report below, no significant effect on perception was found, but experts use more precise language for descriptions. All participants were frequent computer users (min 4 hours per day, $\mu$9.3 h, $\sigma$2.9 h) and all but one used touch input more than one hour per day ($\mu$2.3, $\sigma$1.8).

### Apparatus

Experiment software was implemented in Java 8 on a Microsoft Surface Pro tablet (Windows 8.1 Pro with no OS-level trajectory prediction, dual-core 1.7 GHz CPU, 126 Hz input). Using Ng et al.'s method [33], we determined an average touch latency of 72.6 ms (SD 7.9). A camera captured finger movements, tablet feedback, and audio comments.

### Next-Point Prediction Techniques

We included five of the prediction approaches described in the previous section (source-code is available[1]). All predictors were configured to predict 68 ms in the future. This is to reduce the perceivable portion of the 73 ms end-to-end latency of our apparatus down to an unperceivable 5 ms latency [33].

FIRST – A first-order Taylor series (Eq. 2), based on Lincoln [25] and Cattan et al. [14]. The instantaneous velocity is estimated using the two most recent finger positions.

SECOND[2] – A second-order Taylor series, based on Zhao et al. [46] (Eq. 4). This technique was designed to predict one frame in the future using a linear combination of three previous input frames. To predict 68 ms in the future, we found that scaling the entire linear combination approach to 3 past input positions spaced 68 ms apart performed best.

KALMAN – A Kalman filter, based on Moussavi [30]. We used the OpenCV Kalman implementation with process and measurement model matrices provided on page 10 of the patent. The measurement noise was hand-tuned until obvious prediction errors were minimal.

CURVE – Curve fitting using a second order polynomial, based on Qingkui et al. [5] and least square fitting over the last three points. To eliminate singularities (*e.g.* three points aligned on the Y axis), we determine a reference frame corresponding to the principal axis of the points' inertia matrix and work in this reference frame, then transform the interpolated points back to the world reference frame.

HEURISTIC – The heuristic approach to emphasize speed or acceleration by Kim et al. [19]. Like SECOND, we found scaling the linear combination approach to 3 past input positions spaced 68 ms apart performed best.

During a continuous movement, each predicted point is replaced with the actual input point once the latter is processed by the system (after the 73 ms end-to-end latency time in our case). This reflects the objective of next-point prediction to make visual feedback more responsive, not to filter all input. By ultimately reverting to actual positions, prediction errors do not alter final input like drawing strokes. We also found predicted positions could fall far outside the display (due to near singularities in acceleration calculations for example). Since these large errors typically last less than 3 frames, we suppress points predicted outside the display by reusing the previous predicted point. We believe this was not noticed since no participant comments related to "feedback freezing."

### Tasks

We designed three generic on-surface input tasks spanning different levels of visual feedback: drawing a shape, dragging a square, and panning the background. The purpose is not to measure error or time, but to provide a stimulus with which participants explore different movement directions, curvatures, lengths, speeds, and acceleration profiles.

#### *Drawing*

Moving the finger created a black .5 mm stroke ending with the prediction. Pressing 'C' on a physical keyboard cleared the canvas. First, participants traced over light coloured shapes – a house, an ellipse, a five-pointed star, and a zigzag – spanning most of the display (Fig. 1 a). Initially, shapes were presented in random order, but after the participant could switch between them by pressing the 'S' key. After, they sketched freely on the canvas ignoring the shape (Fig. 1 b).

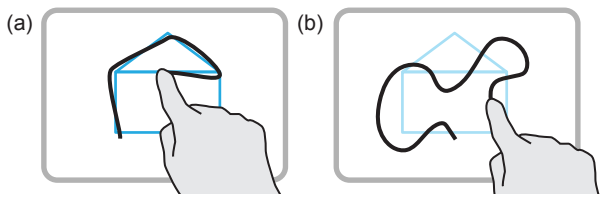


**Figure 1: Drawing task: creating black strokes with the finger: (a) tracing over a shape; (b) free sketching ignoring the shape.**

#### *Dragging*

This task simulates a generic docking action where an object is dragged from one location to another (e.g. moving an icon). A 12.5 mm square is dragged to a 17 mm square dock located 134 mm away (Fig. 2 a). When dragged, the square is rendered at the predicted point. On release, the sensed input

[1] http://ns.inria.fr/mjolnir/predictionmetrics/

[2] We piloted the general second order Taylor method used by Wang [40] and Zhou [47] (Eq. 3), but extrapolating 68 ms using acceleration calculated from the three previous frames, or three previous positions spaced 68 ms apart, exaggerated errors making input uncontrollable and the method completely unusable.

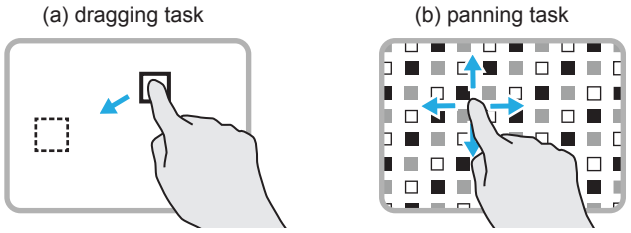


**Figure 2: (a) Dragging task: a square object is moved to dock; (b) Panning task: a background grid is translated in any direction.**

position where the finger was lifted is used. Similar to the sketching portion of the drawing tasks, participants were also encouraged to move the square following arbitrary paths.

*Panning*

This generic task simulates interactions that manipulate a large area of the display, such as two-dimensional panning (e.g. maps), horizontal paging (e.g. viewing photos), and vertical scrolling (e.g. web pages). A repeating grid of multi-coloured squares (each 9-mm, spaced 35-mm) is translated in any direction using finger movements (Fig. 2 b). A multi-coloured grid pattern makes it easier to visually track. Like dragging, the grid is translated using the predicted position during movement and actual position upon release. Participants were asked to pan in different directions and try combining multiple short motions in the same direction.

### Design

With the exception of participant EXPERTISE as a minor between-subject factor (EXPERT, NONEXPERT), the experiment design is within-subject, full factorial. Each participant was exposed to all PREDICTORS and all TASKS. PREDICTOR has 6 levels: the 5 predictors (FIRST, SECOND, HEURISTIC, KALMAN, CURVE) and a control condition with no prediction (CONTROL). TASK has 3 levels (DRAWING, DRAGGING, PANNING). All TASKS are presented in random order for each PREDICTOR. The order of PREDICTOR is balanced with a 6×6 Latin Square. In sum: 6 PREDICTORS × 3 TASKS = 18 conditions per participant.

### Procedure

Participants were given basic explanations of latency and the concept of touch prediction to provide some baseline terminology. They were also explained each TASK. Then they were told their goal is to describe every behaviour that differs from a hypothetically perfectly responsive visual interface. Once this introduction was complete, the recorded portion of the study began with the first TASK using the first PREDICTOR. Participants were asked to explore different movement directions, curvatures, lengths, speeds, and acceleration profiles while interacting with the task stimulus. Once all tasks had been presented for a PREDICTOR, the participant could try any of the tasks again. During, or immediately after this exploration, the participant was prompted to verbally describe problematic behaviours of the task feedback. These descriptions often captured both the behaviours they perceived ("falling behind", "jumping around", etc.) and when they occurred (which task, what kind of motion, what portion of the movement, etc.) They were asked to rate every problematic behavior using a 5-point Likert-type scale: (1) "not disturbing at all"; (2) "a little disturbing"; (3) "disturbing"; (4) "very disturbing"; (5) "unbearable/unacceptable".

Note than absolutely no predictor details were provided, even when requested. The control condition (CONTROL) was not identified, participants were only told there were 6 different predictors. Participants were allowed to take breaks at any time and encouraged to take breaks between predictors. Each session lasted approximately one hour.

## RESULTS

We recorded and transcribed 340 comments: an average of 28.3 (sd 4.4) per participant, 56.7 (sd 2.9) per predictor. Comments from French-speaking participants were translated to English by one of the authors, and then reviewed for accuracy by two other bilingual individuals.

### Categorized Codes

We performed a thematic analysis [11] on all 340 comments. The analysis included the *deductive approach* to classify comments based on patterns observed by the experimenter during the study, and the *semantic approach* to classify comments based on their wording. In accordance with thematic analysis, this was an iterative process during which the whole set of comments was passed through every time a new code was proposed, discarded, split, or merged depending on its relevance and redundancy. This resulted in 30 codes classified into four categories:

SIDE-EFFECT – A spatial accuracy error was observed and at least partially described, like *"wrong orientation"* or *"jitter"*, as opposed to, e.g., *"random"* or *"visible prediction"*. These codes are the main result we are interested in.

CONSEQUENCE – A consequence on overall visual perception or on the feasibility of a task, for example: *"random"*, *"bad for precision"*. These codes may suggest how a prediction method affects perception and task performance.

CONTEXT – The specific circumstances in which an observation occurred, for example during *"direction change"* or during *"fast movements"*. These codes can identify when a SIDE-EFFECT or CONSEQUENCE is more likely to occur.

NON-NEGATIVE – A neutral or positive comment, sometimes to mitigate the impact of a reported problem. For example, "Quite responsive, not many other effects" (P3). Coding comments as neutral or positive helped classify the other more relevant comments. There were 68 occurrences made by all 12 participants, but they were not coded beyond the NON-NEGATIVE classification or used in our analysis.

Table 1 provides descriptions of all SIDE-EFFECT, CONSEQUENCE, and CONTEXT codes with occurrence counts and aggregated disturbance rankings where applicable. In the following results, we focus on SIDE-EFFECT codes, and later use these SIDE-EFFECT codes as the basis for spatial accuracy *metrics* designed to measure the probability of these errors occurring.

| | Code | Occ. Σ | Part. Σ | Disturbance (median, mode) | | Description |
|---|---|---|---|---|---|---|
| SIDE-EFFECT | *"lateness"* | 54 | 12 | 2 | 1 | The prediction was perceived as late, or slow to react to the actual movement. |
| | *"over-anticipate"* | 45 | 11 | 3 | 4 | The prediction was perceived as too far ahead in time, or to over-react to the user input. |
| | *"wrong distance"* | 57 | 11 | 3 | 4 | The prediction was distinguishably far from the finger's actual location. |
| | *"wrong orientation"* | 29 | 12 | 3 | 4 | The prediction was not going in the same direction as the finger motion. |
| | *"jitter"* | 49 | 11 | 3 | 5 | The prediction was perceived as trembling around the finger location. |
| | *"jumps"* | 39 | 12 | 4 | 5 | The prediction appeared to jump away from the finger at times. |
| | *"spring effect"* | 37 | 11 | 3 | 2 | The prediction appeared to "yo-yo" around the finger, possibly in several motions. |
| | *"stick"* | 10 | 5 | 2 | 2 | A short line at the end of the stroke; sometimes of constant orientation, like a pen cursor. |
| CONSEQUENCE | *"random"* | 38 | 11 | 4 | 5 | Could not understand the logic behind some aspects or all of the prediction trajectory. |
| | *"multiple feedback"* | 24 | 6 | 4 | 5 | Seemed like more than one visible feedback (likely caused by jitter and persisitence of vision). |
| | *"blurry"* | 3 | 1 | 1 | 1 | Visual feedback appeared blurry. |
| | *"visible prediction"* | 15 | 6 | 2 | 1 | The prediction was visible (as opposed to perfectly under the finger). |
| | *"disturbing"* | 13 | 7 | 4 | 4 | The prediction was deemed unpleasant or disturbing. |
| | *"bad for completion time"* | 5 | 3 | 3 | 3 | The prediction supposedly harmed time performance. |
| | *"bad for precision"* | 10 | 6 | 4 | 4 | The prediction supposedly harmed precision. |
| | *"bad for task completion"* | 14 | 7 | 3 | 3 | The prediction made it difficult to perform the current task, or an aspect of it. |
| CONTEXT | *"beginning of movement"* | 9 | 2 | | | An effect, whatever it was, was observed at the beginning of the stroke movement. |
| | *"end of movement"* | 34 | 11 | | | An effect was observed at the end of the stroke movement. |
| | *"new targets"* | 7 | 3 | | | The participants were under the impression that the prediction was target-aware. |
| | *"straight movements"* | 7 | 5 | | | An effect was observed during straight movements. |
| | *"angles"* | 23 | 9 | | | An effect was observed during sudden direction changes. |
| | *"curves"* | 16 | 8 | | | An effect was observed during curved trajectories. |
| | *"direction change"* | 40 | 10 | | | An effect was perceived as dependent on direction changes (includes all of Angles and Curves). |
| | *"speed change"* | 5 | 4 | | | An effect was observed specifically during a change of speed. |
| | *"fast movements"* | 64 | 12 | | | An effect was observed during fast movements. |
| | *"slow movements"* | 30 | 10 | | | An effect was observed during slow movements. |
| | *"speed dependent"* | 93 | 12 | | | An effect was perceived as dependent on input speed (includes all of Fast and Slow). |
| | *"short strokes"* | 4 | 3 | | | An effect was perceived for short strokes only. |
| | *"long strokes"* | 2 | 1 | | | An effect was perceived for long strokes only. |

**Table 1: Codes by category: total occurrences (Occ.); total participants (Part.) mentioning; disturbance rating (Dist.) (lower is better).**

### Effect of Participant on Codes

We first examine effects of participant and EXPERTISE on the probability that a given code is reported. For each participant comment and each code, we create an indicator variable assigning '1' if the comment matches the code, '0' otherwise. This data is not normally distributed, so we use one-way Kruskal-Wallis tests with two null hypotheses: (i) EXPERTISE has no effect on the response; and (ii) participant has no effect on the response. Post-hoc tests are Steel-Dwass all-pair (non-parametric) tests with a significance level of 5 %.

Participant EXPERTISE had no significant effect on any codes: perceiving these core issues did not require an expert's eye. NON-EXPERTS used vague terms more often leading to *"random"* and *"annoying"* codes, and expressed more comments related to the consequences of *"curve"* trajectories and on *"precision"*; NON-EXPERTS were more likely to comment on the *"beginning of movement"* context (all $p < .05$).

PARTICIPANT had a significant effect on *"lateness"*, *"jitter"*, and *"spring-effect"* (among SIDE-EFFECT codes, all $p < 0.5$). Post-hoc tests only found P3 more likely than P4 to notice *"lateness"*. Overall, this indicates reasonable consistency.

### Effect of Task and Predictor on Side-effects

Of interest is whether there is an effect of TASK or PREDICTOR on the probability that a given SIDE-EFFECT code is reported. If there is, then estimating probability could be a method to evaluate predictors, perhaps even when used for different tasks. Similar to above, we created an indicator variable for each code and each TASK $\times$ PREDICTOR by assigning '1' if *any* corresponding comments mentioned the code and '0' otherwise. As before, this data is not normally distributed, so one-way Kruskal-Wallis tests are used with the two null hypotheses: (i) TASK has no effect on the response; (ii) PREDICTOR has no effect on the response. Significant effects are reported in subsections below, post-hoc tests are Steel-Dwass all-pair tests. Significance levels are reported as follows: * $p < .05$, ** $p < .01$, *** $p < .001$, **** $p \leq .0001$.

TASK had a significant effect on all SIDE-EFFECT codes except *"lateness"* and *"spring effect"* (all $p < 0.05$). Table 2 lists significant differences: all but *"wrong distance"* separate DRAWING from one or both PANNING and DRAGGING.

| Code | Effects |
|---|---|
| *"over-anticipation"* | PANNING < DRAWING (***) |
| *"wrong distance"* | PANNING < DRAWING (****), DRAGGING (**) |
| *"wrong orientation"* | DRAWING > PANNING, DRAGGING (**) |
| *"jitter"* | DRAWING < PANNING, DRAGGING (****) |
| *"jumps"* | DRAWING < PANNING (*), DRAGGING (***) |
| *"stick"* | DRAWING > PANNING, DRAGGING (**) |

**Table 2: Steel-Dwass tests for TASK.**

PREDICTOR had a significant effect on all SIDE-EFFECT codes except *"spring effect"* and *"stick"* (all $p < 0.05$). Table 3 lists significant differences. Unsurprisingly, participants reported significantly less side-effects with CONTROL, except for *"lateness"*. In fact *"over-anticipate"*, *"wrong orientation"*, *"jitter"*, *"jumps"*, and *"stick"* were never reported with CONTROL. This was further supported by overall comments suggesting that latency was normal, and better than bad prediction errors.

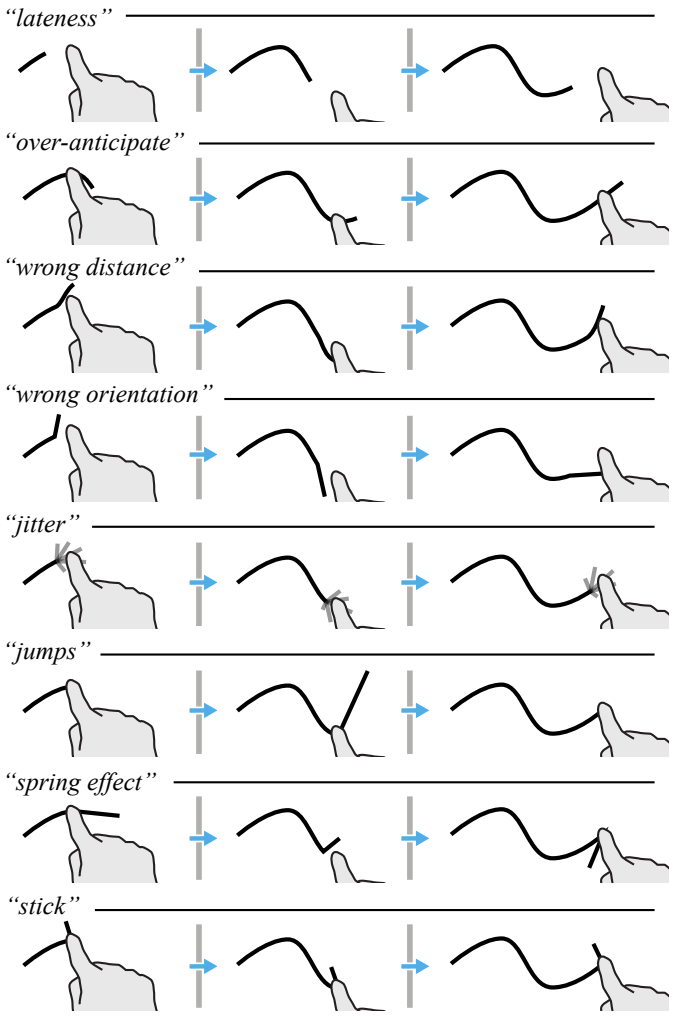


**Figure 3: Conceptual illustration of how side-effects are perceived (see also the accompanying video for side-effect demonstrations).**

Similar to CONTROL, participants noticed more *"lateness"* with KALMAN. There is also a possible trend of fewer comments leading to *"over-anticipation"*, *"wrong distance"*, *"jitter"* and *"jumps"* with KALMAN. This may be due to the Kalman filters' self-correcting mechanism. Assuming most predictions were observed to have a low accuracy (they were based on second order Taylor's series, similar to SECOND), the predictor would rely very little on its own predictions, therefore behaving similarly to CONTROL.

### Disturbance Ratings

Contingency analyses revealed significant effects of SIDE-EFFECT (Pearson $\chi^2 = 179.4$, **), TASK (42.6 ****), and PREDICTOR (251.3 ****) on the overall disturbance rankings: *"lateness"* was rated less disturbing (median 2, mode 1) than other SIDE-EFFECT codes (medians $\geq 3$, modes $\geq 4$); DRAWING was rated less disturbing (median 3, mode 2) than DRAGGING and PANNING (3, 5); and CONTROL (median 2 mode 1) and KALMAN (1, 1) were rated less disturbing than other PREDICTORS (medians 3 or 4, modes 4 or 5).

| Code | Effects |
|---|---|
| *"lateness"* | CONTROL > FIRST (**), HEURISTIC (*)<br>CONTROL > SECOND (***), CURVE (****)<br>KALMAN > CURVE (****), SECOND (**), FIRST (*) |
| *"over-anticipation"* | CONTROL < SECOND (****), HEURISTIC (**)<br>SECOND > CURVE (*), KALMAN (**) |
| *"wrong distance"* | CONTROL < SECOND, FIRST, HEURISTIC (**)<br>KALMAN < HEURISTIC, SECOND (*) |
| *"wrong orientation"* | CONTROL < HEURISTIC (**) |
| *"jitter"* | CONTROL < SECOND (**), CURVE, FIRST (***)<br>KALMAN < SECOND (**), CURVE, FIRST (***)<br>HEURISTIC < CURVE, FIRST (*) |
| *"jump"* | CURVE, FIRST > CONTROL, KALMAN (*) |

**Table 3: Steel-Dwass all-pair tests for Predictor.**

| Code 1 | Code 2 | $\rho$ | p |
|---|---|---|---|
| *"wrong distance"* | *"lateness"* | -.109 | * |
| *"wrong distance"* | *"over-anticipate"* | .429 | **** |
| *"wrong orientation"* | *"over-anticipate"* | .129 | * |
| *"wrong orientation"* | *"wrong distance"* | .145 | ** |
| *"jitter"* | *"lateness"* | -.178 | ** |
| *"jitter"* | *"over-anticipate"* | -.16 | ** |
| *"jitter"* | *"wrong distance"* | -.184 | *** |
| *"jumps"* | *"lateness"* | -.131 | * |
| *"spring effect"* | *"jitter"* | -.143 | ** |
| *"stick"* | *"wrong orientation"* | .196 | *** |

**Table 4: Spearman correlation between SIDE-EFFECTS. Positive values mean those two codes were frequently observed together; negative values mean those two codes were frequently observed separately.**

### Correlations

To test if relationships exist between SIDE-EFFECT codes, we examined Spearman's rank correlations ($\rho$) (Table 4). The strongest correlation is between *"wrong distance"* and *"over-anticipation"*. Although thematic analysis did not group these together, this suggests a similarity. In fact, the following section describes how one metric models the magnitude of both of these SIDE-EFFECT codes quite well. We also calculated Spearman's rank correlations between all codes and TASKS. This is provided in Appendix I for completeness.

## SPATIAL ACCURACY METRICS

Our experiment found that people perceive different kinds of spatial error side-effects caused by next-point predictor inaccuracies. Moreover, we found evidence that the frequencies of several side-effects are significantly affected by the predictor and task. This suggests that estimating the probability of perceiving side-effects, especially the most disturbing ones, could be an effective way to evaluate and compare different predictors. However, conducting an experiment with qualitative coding to evaluate each new predictor requires significant time, effort, and skill. As an alternative, we developed a set of metrics to estimate the magnitude of 7 side-effects using input logs (metric source code is provided[1]).

### Definition and Procedure

Each spatial accuracy metric computes a scalar value estimating the side-effect magnitude (the degree to which predicted points will cause a specific side-effect). A metric consumes a single touch input stroke from finger-down to finger-up with timestamped x-y positions for both the predicted position and

the actual (zero-latency) finger position. We estimate this ground-truth finger position using the position of the input event occurring 68 ms after the prediction (the future prediction time). The last 68 ms of the stroke is truncated since ground-truth positions cannot be estimated. Most of our metrics first identify pairs of finger and predictor positions exhibiting the side-effect and then use those positions to compute the magnitude. If no pairs are found, the metric returns 0. This avoids smoothing out spurious side-effects like RMSE.

The average metric magnitude across strokes is transformed into the probability of noticing a side-effect using linear regression models. The probabilities to model are frequencies of side-effect occurrence as reported by participants for each predictor. Reports are treated as an indicator variable (i.e. a participant reporting the same side-effect three times for a predictor and task counts as one report). The sum of these report indicator variables are converted to a probability by dividing by 36 (12 participants x 3 tasks). We calculate the average metric magnitude for strokes associated with each predictor condition and find the linear regression for each metric.

The models are built using 5,955 strokes from the experiment above. Note that 6,454 strokes were logged, but we removed the first 5$^{th}$ duration percentile ($<$ 47 ms) assuming participants could not see any side-effects in such a short time, and we removed the last 5$^{th}$ percentile ($>$ 6290 ms) because extremely long strokes added additional latency to display all the points.

This procedure and the final metrics below were developed after significant trial and error to satisfy multiple success criteria: formulas should relate to side-effect characteristics and description; simple formulas should be used, with minimal number of parameters; parameters should be robust to variation; correlation to the modeled side-effect should be positive and linear, to facilitate interpretation.

### Metrics and models

Our metrics model the 7 most common side-effects: *"lateness"*, *"over-anticipate"*, *"wrong distance"*, *"wrong orientation"*, *"jitter"*, *"jumps"*, and *"spring effect"*. We did not model *"stick"* since it was noted by only 5 participants (all others were identified by 11 or 12 participants) and it only had 10 occurrences (all others occurred 29 to 57 times).

#### *Lateness*

The lateness metric measures side-effects perceived as "late, or slow to react to the actual movement." The characteristic captured is whether the predicted point is *behind* the finger. This is done by first defining two vectors (Fig. 4 a): the finger direction $\mathbf{f} = F_i - F_{i-1}$, where $F_i$ and $F_{i-1}$ are the current and previous finger points; and the direction from current finger position to current predicted position $\mathbf{d} = P_i - F_i$. If the absolute angle between $\mathbf{f}$ and $\mathbf{d}$ is greater than a threshold $\alpha$, the distance $||\mathbf{d}||$ contributes to a sum. The metric is the average of these "late prediction distances":

$$L(\alpha) = \frac{1}{m}\sum_{i=1}^{n-1} ||\mathbf{d}||, \text{ if } |\text{angle}(\mathbf{f},\mathbf{d})| > \alpha \qquad (6)$$

where $m$ is the number of positions meeting the criteria and $\alpha = 90^\circ$ is used for simplicity. Transforming this metric into the probability of noticing a side-effect using a linear regression produces a significant model ($F_{1,4}$=37.3, p= 0.003) with an $r^2$ of 0.90. The $\alpha$ parameter is robust, values between 60° to 120° produced models with $r^2$ between 0.89 and 0.91.

#### *Over-Anticipation*

The over-anticipation metric measures side-effects perceived as "too far ahead in time, or over-react to the actual movement." It is like lateness, except the characteristic is whether the predicted point is *in front* of the finger. Similar to lateness, if the angle between vectors $\mathbf{f}$ and $\mathbf{d}$ (Fig. 4 b) is below a threshold $\beta$, the distance $||\mathbf{d}||$ contributes to an average sum. Again, $\beta = 90^\circ$ for simplicity.

In practice, considering only angle can result in points being misclassified when the stroke has acute direction changes and the predicted position lags behind the finger (Fig. 4 c). To address this, sequences of finger points are matched with predicted points using Dynamic Time Warping (DTW) [3]. DTW finds an association between sequences so their total distance is minimized. Then, for a predicted point $P_i$, the associated finger point index according to DTW ($DTWindex(P_i)$) can be tested. Note this extra DTW condition was not needed with the lateness metric.

$$OA(\beta) = \frac{1}{m}\sum_{i=1}^{n-1} ||\mathbf{d}||, \begin{cases} \text{if } |\text{angle}(\mathbf{f},\mathbf{d})| < \beta \\ \text{and DTWindex}(P_i) > i \end{cases} \qquad (7)$$

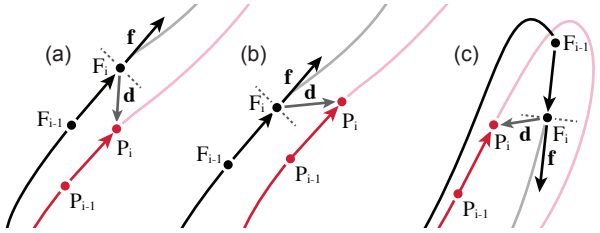


**Figure 4: Conceptual illustration showing current and previous finger positions $F_i$ and $F_{i-1}$ in black and current and previous predicted positions $P_i$ and $P_{i-1}$ in red for: (a) lateness metric; (b) over-anticipation metric; (c) case where a predicted point lagging behind the finger is mistakenly considered as over-anticipated.**

Transforming this metric into a probability with linear regression found a significant model ($F_{1,4}$=33.4, p= 0.004) with $r^2$ 0.89. The $\beta$ parameter is robust, values between 60° to 120° produced models with $r^2$ between 0.88 and 0.89.

#### *Wrong Distance*

Despite extensive trial and error, we could not formulate a specific metric for *"wrong distance"* that outperforms the over-anticipation metric above. However, this is consistent with correlation results (Table 4): the strongest correlation we found is between *"over-anticipate"* and *"wrong distance"* ($\rho$ = .429 ****). We suspect these terms might be interchangeable, despite hinting at different behaviours. Transforming this metric with linear regression found a significant

model ($F_{1,4}$=13.9, p= 0.02), with $r^2$ 0.77. The $\beta$ parameter remains robust, values between 60° to 120° produced models with $r^2$ remaining at 0.77.

*Wrong Orientation*

The wrong orientation metric measures side-effects perceived as "not going in the same direction as the finger motion". The characteristic we capture is when the predicted point is both over-anticipated or slightly lagging and traveling in a direction away from the finger path. We accomplish this by adjusting the over-anticipation metric to include points when the angle between **f** and **d** is below a threshold $\gamma$. We use $\gamma = 90°$ for simplicity. The resulting value is the average absolute angle between **f** and **d** (Equation 8).

$$WO(\gamma,k) = \frac{1}{m}\sum_{i=1}^{n-1} |\text{angle}(\mathbf{f},\mathbf{d})|, \begin{cases} \text{if } |\text{angle}(\mathbf{f},\mathbf{d})| < \gamma \\ \text{and DTWindex}(P_i) > i-k \end{cases} \quad (8)$$

We set $\gamma = 90°$ and $k = 10$. Transforming this metric with linear regression found a significant model ($F_{1,4}$=12.5, p= 0.02), with an $r^2$ of 0.76. For $\gamma = 90°$, $r^2$ ranges from 0.67 ($k = 0$) to 0.39 ($k = 14$). For $k = 10$, $r^2$ ranges from 0.67 ($\gamma = 60°$) to 0.72 ($\gamma = 120°$).

*Jitter*

The jitter metric measures side-effects perceived as "trembling around the finger location". To measure jitter in predicted points, we first create a "jitter-free" version of the predicted points as a baseline. We follow an approach by LaViola [22], and apply a zero phase shift filter to remove high frequency noise with a first order low pass filter (we used signal.filtfilt and a first order Butterworth filter from SciPy). The magnitude of jitter is the average Euclidean distance between the raw predicted points and the filtered predicted points.

We use a cutoff frequency ($f_{cuttoff}$) of 0.15 Hz. Transforming this metric with linear regression found a significant model ($F_{1,4}$=21.5, p= 0.01) with an $r^2$ of 0.84. $r^2$ ranges between 0.77 for $f_{cuttoff} = 0.1$ Hz to 0.68 for 1.0 Hz, but $r^2$ drops to 0 above 2 Hz.

*Jumps*

The method used in the jitter metric also captures side-effects perceived as "jumping away from the finger at times". Setting $f_{cuttoff} = 0.2$ Hz we found a significant regression model ($F_{1,4}$=1638, p< 0.001) with $r^2$ of 0.99. The sensitivity of the cutoff frequency for jumps is similar to jitter: $r^2$ ranges between 0.82 for $f_{cuttoff} = 0.1$ Hz, to 0.89 for 1.0 Hz, and then drops to 0 above 2 Hz.

*Spring Effect*

The spring metric measures side-effects that "yo-yo around the finger". We experimented with using acceleration directly, but found that using the second order partial derivative of distances between each finger and predicted points to detect local maxima and minima worked better. The metric reports the average number of maximums and minimums over the number of points of each stroke.

$$S = \frac{1}{n}\sum_{i=1}^{n-1} \begin{cases} 1 \\ 0 \;\; if \;\; \frac{\partial^2 \mathbf{d}[\mathbf{i}]}{\partial t^2}\frac{\partial^2 \mathbf{d}[\mathbf{i+1}]}{\partial t^2} < 0 \end{cases} \quad (9)$$

Transforming this metric into a probability with regression found a significant model ($F_{1,4}$=13.7, p= 0.02) with $r^2$ 0.77.

### Accuracy of the resulting models

Table 5 summarizes the regression models. For each intended side-effect, our metric provides the best results for the intended side-effect in terms of correlation. Note the slope ($m$) is always positive for the intended side-effect. This makes the metrics intuitive since higher values correspond to greater chances of noticing a problem (and vice versa).

Our metrics provide better results than state-of-the-art metrics. RMSE was used by LaViola [22] and it is very similar to average Euclidean distance used by Wu et al. [44] and LaValle et al. [21]. 95$^{th}$ percentile of distance between finger points and predicted points is a more principled way to measure LaValle et al.'s [21] "worst Euclidean error distance."

### Application

Our results serve three main purposes.

First, the magnitude of a metric indicates the likelihood of people noticing the corresponding side-effect. This can be used to compare current predictors on relevant, user-defined criteria, or to inform the design of new prediction methods to avoid or balance perceived side-effects.

Second, our metric models can be used with our input logs and experiment protocol to develop new predictors without collecting and coding participant comments. Our input logs (available at[1]) are a representative sample of touch input for different tasks. Practitioners can use these logs to simulate new prediction algorithms at different latencies to estimate the likelihood that people will notice each side-effect. This would be useful for exploring different approaches, parameter tuning, and initial validation. In later stages, practitioners can use a simplified version of our experiment protocol to gather input logs when participants use a predictor prototype with different tasks and latencies. The simplification is that participant comments do not need be recorded and coded, our metrics can be used with these more specific input logs to estimate side-effects more accurately.

Finally, our metric-based models can be used to benchmark predictor side-effects behaviour across different levels of prediction time and end-to-end latencies. This can be used to establish practical prediction time thresholds for different predictors, reveal side-effect trade-offs implicit in different prediction approaches, and guide practitioners to refine algorithms to flatten side-effect probability curves. We provide the results of such a simulation in Appendix II.

Our metrics have the potential to streamline next-point predictor design, but most importantly, they enable practitioners to explore predictor design systematically.

## DISCUSSION

Our experiment results and metric development motivate several topics for discussion.

### Noticing Latency is Least Disturbing

A general finding is that perceiving latency is less disturbing than other types of spatial error side-effects introduced by total-prediction methods. This is shown by lower disturbance ratings for the CONTROL condition (see p. ) and the *"lateness"* side-effect (Table 1). Some participants explicitly stated a preference for latency over other errors. To be clear, this does not mean latency is preferred over no latency (refer to research surveyed earlier in this paper) nor does it signal the end for prediction research.

Our results suggest that predictors that reduce some latency without side effects are preferred to those that remove all latency with visible side effects. We emphasize the importance of designing prediction methods that keep other types of side-effects below a perceivable threshold.

### Traditional metrics model side-effects poorly

Our work reveals that RMSE and max Euclidean distance (tested using 95$^{th}$ percentile) do not quantify the kinds of prediction errors that really disturb people. We found they have reasonable correlations with the least disturbing *"lateness"* side-effect (thought not as well as our lateness metric), but they poorly capture most other side-effects (see Table 5 and Appendix II). Moreover, RMSE and max Euclidean distance correlate negatively with *"jitter"* and *"jumps"* side-effects (slope $m < 0$ and $r^2 > .65$ in Table 5) which means selecting a predictor based on their lower scores could actually increase the risk of these side-effects emerging.

### Similar side-effects, metrics, and models

Some SIDE-EFFECTS are positively correlated in their response (Table 4) or in their metric, which raises the question of whether they are redundant. We identify three possible causes, informed by our observations during the study.

*Perceptual framework* – Thematic analysis relies on the participants' capacity to describe an observation accurately and in a consistent way. In effect, it emerged that our SIDE-EFFECT codes can be categorized by the perspective they express, e.g. temporal errors (*"latency"* and *"over-anticipate"*), geometric errors (*"wrong orientation"* and *"wrong distance"*), instability (*"jitter"* and *"jumps"*), or metaphors (*"spring effect"* and *"stick"*). Interestingly, these perspectives also emerged in our metrics: *"latency"* and *"over-anticipate"* are best modeled by similar formulae with different settings; the same goes for *"jitter"* and *"jumps"*. In both cases, the side effects are not correlated (Table 4).

*Hierarchical formulations* – However these perspectives are not mutually exclusive, and similar phenomena can be expressed differently under different perspectives. For instance, while *"spring effect"* is a clear case of both *"wrong orientation"* and *"wrong distance"*, no significant correlation was found. Similarly, while *"latency"* and *"jumps"* both imply that the distance between finger and prediction must be wrong, we found respectively a significant negative correlation and no correlation with *"wrong distance"*. Overall, we propose that users perceive or describe prediction errors first as metaphors (e.g. a "spring" or a "stick") or known phenomena (e.g. "latency" and "jumps"), and then resort to geometric descriptions if necessary. This would explain why arguably trivial geometric phenomena prove challenging to formulate, since they were not systematically reported as such.

*Causal relationships* – We propose that some SIDE-EFFECTS are consequences of others, which affects their correlations and models. For instance, *"over-anticipate"* expresses that predictors were too prompt to react to changes in speed or orientation, making the reaction disproportionate. This could result in the prediction being far from the finger or going in the wrong direction. Some instances of *"wrong distance"* and *"wrong orientation"* could therefore be consequences of *"over-anticipation"*. This would explain the high correlation between the three (Table 4), and the fact that we could not produce a metric for *"wrong distance"* that could beat the *"over-anticipation"* metric.

Note that despite these similarities we did not merge similarly-modelled or correlated side-effects. Our primarily objective remains to predict what users *perceive* as disturbing prediction behaviours.

### Visual feedback

The nature of the reported side-effects can be strongly affected by the visual feedback during different tasks (Table 2). Some terms are directly linked to the visual feedback: *"stick"* requires a trace (DRAWING), and *"jumps"* is more fitting to a translation metaphor (DRAGGING, PANNING). Others terms, while not systematically linked to a given task, might still be affected. For instance, having a thick line linking the last detected point to the prediction (DRAWING) likely emphasized

| | *"lateness"* | | | *"over-anticipate"* | | | *"wrong distance"* | | | *"wrong orientation"* | | | *"jitter"* | | | *"jumps"* | | | *"spring effect"* | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | m | b | $r^2$ | m | b | $r^2$ | m | b | $r^2$ | m | b | $r^2$ | m | b | $r^2$ | m | b | $r^2$ | m | b | $r^2$ |
| RMSE metric | 0.3 | -10.0 | 0.81 | -0.1 | 31.4 | 0.14 | -0.1 | 43.7 | 0.49 | -0.1 | 22.8 | 0.35 | -0.2 | 49.6 | 0.76 | -0.2 | 37.2 | 0.82 | -0.0 | 21.8 | 0.22 |
| 95$^{th}$ percentile metric | 0.1 | -12.6 | 0.74 | -0.0 | 31.4 | 0.11 | -0.1 | 45.3 | 0.45 | -0.0 | 24.4 | 0.37 | -0.1 | 51.0 | 0.66 | -0.1 | 39.0 | 0.75 | -0.0 | 22.7 | 0.24 |
| Lateness metric | **0.2** | **-5.9** | **0.90** | -0.1 | 34.4 | 0.32 | -0.2 | 43.6 | 0.70 | -0.1 | 22.6 | 0.49 | -0.2 | 44.3 | 0.74 | -0.1 | 33.6 | 0.82 | -0.0 | 21.3 | 0.26 |
| Over-anticipation metric | -0.6 | 43.5 | 0.65 | **0.6** | **1.0** | **0.89** | **0.5** | **8.8** | **0.78** | 0.2 | 5.4 | 0.44 | 0.4 | 7.1 | 0.37 | 0.3 | 7.0 | 0.34 | 0.1 | 13.5 | 0.11 |
| Wrong orientation metric | -2.3 | 79.1 | 0.86 | 0.9 | -2.4 | 0.22 | 1.4 | -9.4 | 0.61 | **1.0** | **-10.6** | **0.76** | 1.7 | -20.6 | 0.59 | 1.4 | -17.3 | 0.78 | 0.1 | 12.9 | 0.03 |
| Jitter metric | -2.2 | 72.4 | 0.85 | 0.9 | -0.7 | 0.24 | 1.5 | -7.7 | 0.71 | 0.7 | -2.5 | 0.42 | **2.0** | **-22.8** | **0.84** | 1.5 | -17.2 | 0.99 | 0.2 | 11.2 | 0.09 |
| Jump metric | -2.0 | 59.5 | 0.76 | 0.7 | 8.4 | 0.13 | 1.3 | 1.8 | 0.59 | 0.6 | 1.9 | 0.35 | 1.9 | -12.9 | 0.82 | **1.5** | **-10.0** | **1.00** | 0.2 | 13.0 | 0.06 |
| Spring effect metric | -538.6 | 25.5 | 0.01 | 986.1 | 16.0 | 0.03 | 607.1 | 22.3 | 0.01 | -35.9 | 13.1 | 0.00 | 7.0 | 20.8 | 0.00 | -65.0 | 16.5 | 0.00 | **2006.9** | **7.3** | **0.77** |

**Table 5: Slope (m), intercept (b) and correlation ($r^2$) for linear regressions for each metric and side-effect. The regression models the probability of noticing a side-effect (in %) based on the side-effect magnitude for each predictor computed by the metric. Highest $r^2$ value by row and column in bold.**

the errors in predicted orientation (see *"wrong orientation"* in Table 2) while comparable behaviors in DRAGGING and PANNING could be perceived as non-specific *"jumps"*. Conversely, systematic errors of smaller amplitudes (*"jitter"*) are more noticeable when the visual feedback is larger than the finger (DRAGGING, PANNING) than when it remains mostly under the finger (DRAWING). Finally, feedbacks that are not focused around the finger (PANNING) likely decrease the notability of amplitude errors (*"wrong distance"*), provided that the orientation is correct.

### Links between Predictor Method and Side-effects

Perhaps enabled by the relatively large prediction time in our study, we can observe some links between SIDE-EFFECTS and how prediction methods use available input (Table 3). Using a small number of points (2 for FIRST, 3 for SECOND and CURVE) increases the variability of the prediction, which leads to more observed instability (*"jitter"*); HEURISTIC (5 points) behaved comparably better for *"jitter"*. Using points within a time interval much smaller than the prediction (8 ms for FIRST, 16 ms for CURVE) makes the prediction more sensible to sensing noise and input noise, which become all the more exaggerated by the distant prediction (*"jumps"*). On the contrary, involving older data in the prediction (340 ms for HEURISTIC) can introduce delays, especially regarding prediction direction (*"wrong orientation"*).

### Limitations

When calculating our metrics, we interpolated input points to match predicted points using a constant estimation of the system's end-to-end latency. In doing so, we ignored the variability of the latency [12], which may have introduced minute inaccuracies. Being able to capture strokes with a real time measure of the end-to-end latency would provide better estimations of the finger positions, and possibly better estimations of our metrics.

Thematic analysis, as all methods to identify patterns of meaning, is sensitive to participants' capacity for observation and expression, as well as the practitioner's accuracy in extracting meaningful codes. While we expect to have minimized the former by involving both HCI experts and laymen, there is always a possibility that important effects were missed, either by them, or by us. Similarly, different prediction methods might introduce new side-effects that our study did not capture. Replications of our work, and its applications to other predictors, latencies and contexts of use, will likely answer these questions.

## CONCLUSION

Our work is the first systematic study of different kinds of perceived side-effects caused by spatial inaccuracies in current next-point prediction methods. To make our results immediately applicable, we offer a set of metrics with linear models to estimate the perceptual probability of the seven most common side-effects. Not only are our metrics more intuitive, but they capture nuanced effects between predictor and side-effects better than previous measures like RMSE.

As future work, perception thresholds for each metric could be established using a Just-Noticeable-Difference (JND) experiment [32,33]. Also, our approach to identifying and modelling visual side-effects could be applied to other input contexts like augmented reality (where latency is a significant issue [44]), to other devices like styluses (where a nib will not hide small prediction errors) and to other user groups like digital artists (who may be more aware of prediction errors).

Our goal was not to find the best predictor or suggest new predictor designs. But now, accelerated by our experiment protocol, a corpus of input logs, and metric source code, practitioners can efficiently and scientifically compare and benchmark any next-point predictor. In addition, armed with an understanding of side-effects, next-point prediction designers can now make informed decisions to minimize them, and enabled by our metrics, they have the tools to immediately measure their success.

## ACKNOWLEDGEMENTS

The project was partially supported by ANR (TurboTouch, ANR-14-CE24-0009), the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement 637991), and the Natural Sciences and Engineering Research Council of Canada (NSERC) under Engage Grant 476701-2014. We also thank Daniel Wigdor for facilitating the collaboration between the co-authors.

## REFERENCES

1. TouchPredictionParameters structure documentation. Retrieved July 25, 2016 from https://msdn.microsoft.com/en-us/library/windows/desktop/hh969214(v=vs.85).aspx.
2. What's new in Cocoa Touch. Retrieved April 10, 2016 from https://developer.apple.com/videos/wwdc/2015/?id=107, watch from 13:10 to 15:24.
3. *Information Retrieval for Music and Motion*. Springer Berlin Heidelberg, 2007, ch. Dynamic Time Warping, 69–84.
4. Multi-touch trajectory tracking method, June 15 2011. CN Patent App. CN 201,110,030,430.
5. Curve fitting based touch trajectory smoothing method and system, July 2 2014. CN Patent App. CN 201,210,585,264.
6. Anderson, G., Doherty, R., and Ganapathy, S. User perception of touch screen latency. In *Proc. Design, User Experience, and Usability*, DUXU '11 (2011), 195–202.
7. Annett, M., Ng, A., Dietz, P., Bischof, W. F., and Gupta, A. How low should we go?: Understanding the perception of latency while inking. In *Proc. Graphics Interface 2014*, GI '14, Canadian Information Processing Society (2014), 167–174.
8. Asano, T., Sharlin, E., Kitamura, Y., Takashima, K., and Kishino, F. Predictive interaction using the delphian desktop. In *Proc. 18th Annual ACM Symposium on User*

*Interface Software and Technology*, UIST '05, ACM (2005), 133–141.

9. Bérard, F., and Blanch, R. Two touch system latency estimators: High accuracy and low overhead. In *Proc. 2013 ACM International Conference on Interactive Tabletops and Surfaces*, ITS '13, ACM (2013), 241–250.
10. Biswas, P., Aydemir, G., Langdon, P., and Godsill, S. Intent recognition using neural networks and Kalman filters. In *Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data*, A. Holzinger and G. Pasi, Eds., vol. 7947 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2013, 112–123.
11. Braun, V., and Clarke, V. Using thematic analysis in psychology. *Qualitative Research in Psychology 3*, 2 (2006), 77–101.
12. Casiez, G., Conversy, S., Falce, M., Huot, S., and Roussel, N. Looking through the eye of the mouse: A simple method for measuring end-to-end latency using an optical mouse. In *Proc. 28th Annual ACM Symposium on User Interface Software and Technology*, UIST '15, ACM (2015), 629–636.
13. Cattan, E., Rochet-Capellan, A., and Bérard, F. A predictive approach for an end-to-end touch-latency measurement. In *Proc. 2015 International Conference on Interactive Tabletops & Surfaces*, ITS '15, ACM (2015), 215–218.
14. Cattan, E., Rochet-Capellan, A., Perrier, P., and Bérard, F. Reducing latency with a continuous prediction: Effects on users' performance in direct-touch target acquisitions. In *Proc. 2015 International Conference on Interactive Tabletops and Surfaces*, ITS '15, ACM (2015), 205–214.
15. Deber, J., Jota, R., Forlines, C., and Wigdor, D. How much faster is fast enough?: User perception of latency & latency improvements in direct and indirect touch. In *Proc. 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, ACM (2015), 1827–1836.
16. Dilger, D. E. Agawi TouchMark contrasts iPad's fast screen response to laggy Android tablets, Oct. 2013.
17. Jota, R., Ng, A., Dietz, P., and Wigdor, D. How fast is fast enough?: A study of the effects of latency in direct-touch pointing tasks. In *Proc. SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, ACM (2013), 2291–2300.
18. Kaaresoja, T., and Brewster, S. Feedback is... late: Measuring multimodal delays in mobile device touchscreen interaction. In *International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction*, ICMI-MLMI '10, ACM (2010), 2:1–2:8.
19. Kim, B., and Lim, Y. Mobile terminal and touch coordinate predicting method thereof, Aug. 28 2014. WO Patent App. PCT/KR2014/000,661.
20. Lank, E., Cheng, Y.-C. N., and Ruiz, J. Endpoint prediction using motion kinematics. In *Proc. SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, ACM (2007), 637–646.
21. LaValle, S., Yershova, A., Katsev, M., and Antonov, M. Head tracking for the oculus rift. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on* (May 2014), 187–194.
22. LaViola, J. J. Double exponential smoothing: An alternative to Kalman filter-based predictive tracking. In *Proc. Workshop on Virtual Environments 2003*, EGVE '03, ACM (2003), 199–206.
23. Leigh, D., Forlines, C., Jota, R., Sanders, S., and Wigdor, D. High rate, low-latency multi-touch sensing with simultaneous orthogonal multiplexing. In *Proc. 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, ACM (2014), 355–364.
24. Liang, J., Shaw, C., and Green, M. On temporal-spatial realism in the virtual reality environment. In *Proc. 4th Annual ACM Symposium on User Interface Software and Technology*, UIST '91, ACM (1991), 19–25.
25. LINCOLN, J. Position lag reduction for computer drawing, Oct. 17 2013. US Patent App. 13/444,029.
26. MacKenzie, I. S., and Ware, C. Lag as a determinant of human performance in interactive systems. In *Proc. INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, CHI '93, ACM (1993), 488–493.
27. Meyer, D. E., Keith-Smith, J., Kornblum, S., Abrams, R. A., and Wright, C. E. Speed-accuracy tradeoffs in aimed movements: Toward a theory of rapid voluntary action. *Attention and performance 13: Motor representation and control* (1990), 173–226.
28. Miller, R. B. Response time in man-computer conversational transactions. In *Proc. December 9-11, 1968, Fall Joint Computer Conference, Part I*, AFIPS '68 (Fall, part I), ACM (1968), 267–277.
29. Mine, M. Characterization of end-to-end delays in head-mounted display systems. Tech. rep., University of North Carolina at Chapel Hill, 1993.
30. Moussavi, F. Methods and apparatus for incremental prediction of input device motion, July 1 2014. US Patent 8,766,915.
31. Nelson, W. T., Roe, M. M., Bolia, R. S., and Morley, R. M. Assessing simulator sickness in a see-through hmd: Effects of time delay, time on task, and task complexity. Tech. rep., DTIC Document, 2000.

32. Ng, A., Annett, M., Dietz, P., Gupta, A., and Bischof, W. F. In the blink of an eye: Investigating latency perception during stylus interaction. In *Proc. 32Nd Annual ACM Conference on Human Factors in Computing Systems*, CHI '14, ACM (2014), 1103–1112.
33. Ng, A., Lepinski, J., Wigdor, D., Sanders, S., and Dietz, P. Designing for low-latency direct-touch input. In *Proc. 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, ACM (2012), 453–464.
34. Pasqual, P. T., and Wobbrock, J. O. Mouse pointing endpoint prediction using kinematic template matching. In *Proc. SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, ACM (2014), 743–752.
35. Pavlovych, A., and Gutwin, C. Assessing target acquisition and tracking performance for complex moving targets in the presence of latency and jitter. In *Proc. Graphics Interface 2012*, GI '12, Canadian Information Processing Society (2012), 109–116.
36. Pavlovych, A., and Stuerzlinger, W. The tradeoff between spatial jitter and latency in pointing tasks. In *Proc. 1st ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, EICS '09, ACM (2009), 187–196.
37. Pavlovych, A., and Stuerzlinger, W. Target following performance in the presence of latency, jitter, and signal dropouts. In *Proc. Graphics Interface 2011*, GI '11, Canadian Human-Computer Communications Society (2011), 33–40.
38. Shneiderman, B. Response time and display rate in human performance with computers. *ACM Computing Surveys 16*, 3 (Sept. 1984), 265–285.
39. Tsoi, P., and Xiao, J. Advanced touch input on ios. Technical report, Apple Inc., 2005.
40. Wang, W. Touch tracking device and method for a touch screen, Jan. 24 2013. US Patent App. 13/367,371.
41. Ware, C., and Balakrishnan, R. Reaching for objects in VR displays: Lag and frame rate. *ACM ToCHI 1*, 4 (Dec. 1994), 331–356.
42. Welch, G., and Bishop, G. An introduction to the Kalman filter. Tech. rep., 1995.
43. Westerman, W., and Elias, J. Multi-touch contact tracking using predicted paths, Dec. 18 2012. US Patent 8,334,846.
44. Wu, J.-R., and Ouhyoung, M. On latency compensation and its effects on head-motion trajectories in virtual environments. *The Visual Computer 16*, 2 (2000), 79–90.
45. Xia, H., Jota, R., McCanny, B., Yu, Z., Forlines, C., Singh, K., and Wigdor, D. Zero-latency tapping: Using hover information to predict touch locations and eliminate touchdown latency. In *Proc. 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, ACM (2014), 205–214.
46. Zhao, W., Stevens, D., Uzelac, A., Benko, H., and Miller, J. Prediction-based touch contact tracking, Aug. 16 2012. US Patent App. 13/152,991.
47. Zhou, S. Electronic device and method for providing tactile stimulation, June 19 2014. US Patent App. 13/729,048.

## APPENDIX I: CORRELATIONS WITH SIDE-EFFECTS

| | Code 1 | Code 2 | General correlation ($\rho$, p) | | DRAGGING correlation ($\rho$, p) | | PANNING correlation ($\rho$, p) | | DRAWING correlation ($\rho$, p) | |
|---|---|---|---|---|---|---|---|---|---|---|
| Between SIDE-EFFECT | *"wrong distance"* | *"lateness"* | -.109 | * | | | | | | |
| | *"wrong distance"* | *"over-anticipate"* | .429 | **** | .394 | **** | .293 | ** | .441 | **** |
| | *"wrong orientation"* | *"over-anticipate"* | .129 | * | | | | | | |
| | *"wrong orientation"* | *"wrong distance"* | .145 | ** | .228 | * | | | | |
| | *"jitter"* | *"lateness"* | -.178 | ** | -.247 | ** | -.211 | * | | |
| | *"jitter"* | *"over-anticipate"* | -.16 | ** | | | | | | |
| | *"jitter"* | *"wrong distance"* | -.184 | *** | -.217 | * | | | | |
| | *"jumps"* | *"lateness"* | -.131 | * | -.177 | * | | | | |
| | *"jumps"* | *"wrong orientation"* | | | | | .256 | ** | | |
| | *"spring effect"* | *"jitter"* | -.143 | ** | -.108 | * | | | | |
| | *"spring effect"* | *"over-anticipate"* | | | | | .218 | * | | |
| | *"stick"* | *"wrong orientation"* | .196 | *** | | | | | | |
| SIDE-EFFECT vs. CONSEQUENCE | *"over-anticipate"* | *"random"* | | | | | | | .222 | * |
| | *"wrong distance"* | *"random"* | | | | | | | .247 | *** |
| | *"wrong orientation"* | *"random"* | | | | | | | .288 | ** |
| | *"jumps"* | *"random"* | .136 | * | | | | | | |
| | *"over-anticipate"* | *"bad for precision"* | | | .198 | * | | | | |
| | *"wrong distance"* | *"bad for precision"* | .202 | *** | .228 | * | | | | |
| | *"wrong orientation"* | *"bad for precision"* | .196 | *** | .375 | **** | | | | |
| | *"jitter"* | *"bad for task completion"* | | | | | .199 | * | | |
| SIDE-EFFECT vs. NON-NEGATIVE | *"lateness"* | *"neutral"* | -.157 | ** | -.197 | * | | | | |
| | *"over-anticipate"* | *"neutral"* | -.109 | * | | | | | | |
| | *"wrong distance"* | *"neutral"* | -.146 | ** | | | | | -.302 | ** |
| | *"multiple feedback"* | *"neutral"* | -.138 | * | | | | | | |
| | *"jitter"* | *"neutral"* | -.184 | *** | | | -.245 | * | | |
| | *"jumps"* | *"neutral"* | -.157 | ** | | | -.238 | * | | |
| | *"spring effect"* | *"neutral"* | -.128 | * | | | | | | |
| | *"random"* | *"neutral"* | -.154 | ** | | | -.198 | * | | |
| SIDE-EFFECT vs. CONTEXT | *"lateness"* | *"end of movement"* | .258 | **** | .247 | ** | .311 | ** | .197 | * |
| | *"lateness"* | *"direction change"* | -.109 | * | | | | | | |
| | *"lateness"* | *"angles"* | -.117 | * | | | | | | |
| | *"over-anticipate"* | *"direction change"* | .262 | **** | | | | | .325 | *** |
| | *"over-anticipate"* | *"angles"* | .137 | * | | | | | .191 | * |
| | *"over-anticipate"* | *"curves"* | .2 | *** | .236 | ** | | | .194 | * |
| | *"wrong distance"* | *"direction change"* | .252 | **** | .197 | * | | | .229 | * |
| | *"wrong distance"* | *"angles"* | .193 | *** | | | .204 | * | .225 | * |
| | *"wrong distance"* | *"curves"* | .198 | *** | | | .266 | ** | .239 | * |
| | *"wrong distance"* | *"speed-dependent"* | .184 | *** | .3 | *** | | | .246 | * |
| | *"wrong distance"* | *"fast"* | .146 | ** | .256 | ** | | | .24 | * |
| | *"wrong orientation"* | *"end of movement"* | | | | | | | .201 | * |
| | *"wrong orientation"* | *"direction change"* | .248 | **** | .259 | ** | | | .22 | * |
| | *"wrong orientation"* | *"curves"* | .131 | * | .195 | * | | | | |
| | *"wrong orientation"* | *"slow"* | | | | | | | .222 | * |
| | *"multiple feedback"* | *"direction change"* | | | .282 | ** | .204 | * | | |
| | *"multiple feedback"* | *"angles"* | .109 | * | .336 | **** | | | | |
| | *"multiple feedback"* | *"speed-dependent"* | .192 | *** | .295 | *** | | | | |
| | *"multiple feedback"* | *"fast"* | .161 | ** | .258 | ** | | | | |
| | *"jitter"* | *"end of movement"* | -.137 | * | -.187 | * | | | | |
| | *"jitter"* | *"direction change"* | -.15 | ** | | | | | | |
| | *"jitter"* | *"angles"* | -.111 | * | | | | | | |
| | *"jitter"* | *"slow"* | .168 | ** | .197 | * | | | | |
| | *"jumps"* | *"slow"* | .279 | **** | .246 | ** | .312 | ** | | |
| | *"spring effect"* | *"end of movement"* | .23 | **** | .198 | * | .297 | ** | .213 | * |
| | *"spring effect"* | *"direction change"* | .107 | * | | | | | | |
| | *"spring effect"* | *"angles"* | .169 | ** | | | | | .322 | *** |
| | *"spring effect"* | *"curves"* | .145 | ** | .339 | **** | | | | |
| | *"spring effect"* | *"speed-dependent"* | | | | | .234 | * | | |
| | *"spring effect"* | *"fast"* | .122 | * | | | .309 | ** | | |
| | *"spring effect"* | *"slow"* | -.109 | * | | | | | | |
| | *"random"* | *"end of movement"* | | | | | | | .273 | ** |
| | *"random"* | *"direction change"* | .131 | * | | | | | .29 | ** |
| | *"random"* | *"angles"* | | | | | .236 | * | | |
| | *"stick"* | *"direction change"* | .153 | ** | | | | | | |
| | *"stick"* | *"curves"* | .208 | **** | | | | | .229 | * |

**Table 6: Spearman correlation for all data. Positive values mean those two codes were frequently observed together; negative values mean those two codes were frequently observed separately.**

## APPENDIX II: SIMULATIONS OF DIFFERENT LATENCIES

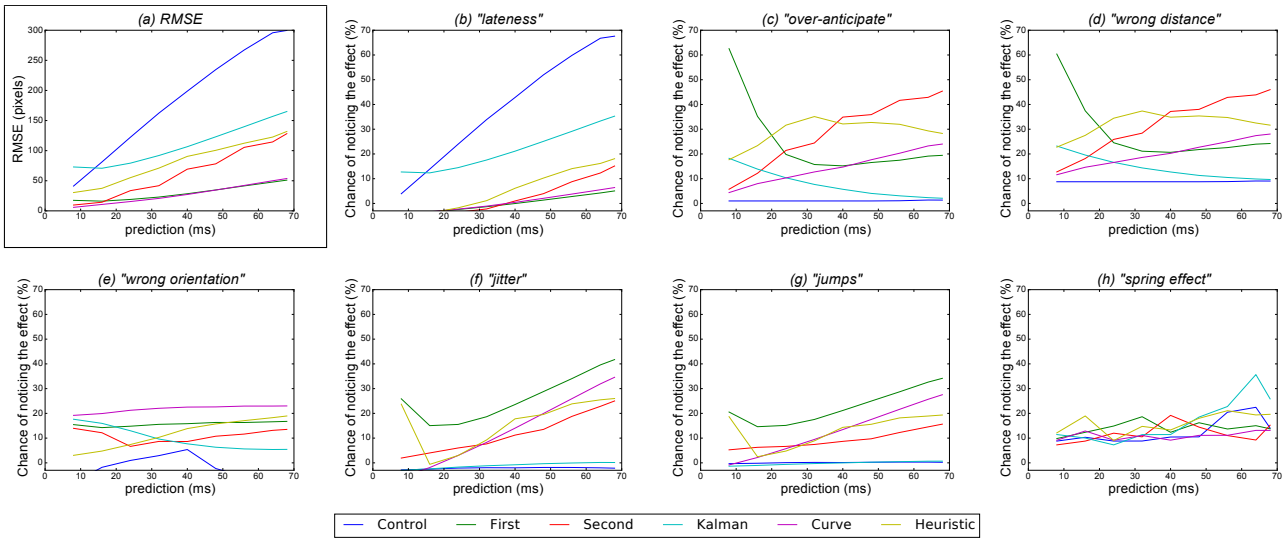


**Figure 5: Simulation of each predictor for different amount of prediction from 8 to 68 ms and the corresponding results for each side-effect and the correspond metric. RMSE is added for comparison. Each simulation was run as if the system had the corresponding end-to-end latency (*e.g.* 16 ms end-to-end latency for 16 ms prediction time).**

The chances of *"lateness"* being perceived increase with the amount of prediction, but at different rates for each predictor. Considering that RMSE best predicts *"lateness"* (Table 5), it is interesting that the RMSE curves are so similar to the curves obtained for our lateness metric.

It is interesting to see that CONTROL never over-anticipates.

For *"over-anticipate"* and *"wrong distance"*, it can appear counter-intuitive that FIRST has a decreasing curve, considering the way the predictor works; one would expect to see a straight line increasing as the prediction duration increases. This is explained by the fact that we filter out any point predicted outside of the screen. Such points are caused by near-singularity in speed calculations, and their distance to the finger increase with the predicted duration. Therefore, for lower predictions, these predicted points are far but more likely to remain on screen, and thus highly contribute to the metric. Note that this effect is partially mirrored in the simulations for *"jitter"* and *"jumps"*: if more predicted points are likely to appear suddenly away from the finger, both side-effects will be observed. This is an interesting illustration of the effects of practical implementation considerations, and of the efficiency of our metrics to simulate events that were not perceived during the experiment.

As discussed in the article, at 68 ms KALMAN tends to behave like CONTROL which we hypothesize is because it detects the inaccuracies of its model and corrects accordingly by relying more on its measurements. This is supported by the above simulations: for *"over-anticipate"* and *"wrong distance"*, KALMAN's responses get closer to CONTROL's as predicted durations increase.

For *"wrong orientation"* and *"spring effect"*, the amount of prediction seems to have little effect.

For *"jitter"* and *"jump"*, CONTROL and KALMAN are very close to each other and the other predictors show raising chance of noticing noise, which is to be expected.