The Space Between the Notes: Adding Expressive Pitch Control to the Piano Keyboard

Andrew P. McPherson
Queen Mary University of London
Mile End Road, London E1 4NS, United Kingdom
andrewm@eecs.qmul.ac.uk

Adrian Gierakowski
Queen Mary University of London
Mile End Road, London E1 4NS, United Kingdom
adriang@eecs.qmul.ac.uk

Adam M. Stark
Queen Mary University of London
Mile End Road, London E1 4NS, United Kingdom
adam.stark@eecs.qmul.ac.uk

ABSTRACT
This paper addresses the question of how to extend the capabilities of a well-established interface in a way that respects users' existing expertise. The piano-style keyboard is among the most widely used and versatile of digital musical interfaces. However, it lacks the ability to alter the pitch of a note after it has been played, a limitation which prevents the performer from executing common expressive techniques including vibrato and pitch bending. We present a system for controlling pitch from the keyboard surface using capacitive touch sensors to measure the locations of the player's fingers on the keys. The large community of trained pianists makes the keyboard a compelling target for augmentation, but it also poses a challenge: how can a musical interface be extended while making use of the existing techniques performers have spent thousands of hours learning? In this paper, user studies with conservatory pianists explore the constraints of traditional keyboard technique and evaluate the usability of the continuous pitch control system. The paper also discusses implications for the extension of other established interfaces in musical and non-musical contexts.

Author Keywords
Piano keyboard; musical interfaces; capacitive touch sensing; performance technique; expressivity; digital arts

ACM Classification Keywords
H.5.5 Sound and Music Computing: Systems, Modeling

General Terms
Design; Human Factors; Experimentation

INTRODUCTION
Electronic piano-style keyboards are ubiquitous in many styles of musical performance. Beyond emulating the acoustic piano, the keyboard is frequently used to control sampled string, wind or synthesiser sounds. However, the keyboard lacks an important feature found on nearly all string and wind instruments: once a note is played, the performer has no ability to further control its pitch. This limitation prevents the keyboardist from emulating common expressive techniques including vibrato, pitch bends and slides between notes. Some keyboards feature a "pitch wheel" to the side of the keys for fine pitch adjustment, but this arrangement has several drawbacks: it requires the dedicated use of one hand; it lacks the ability to independently control the pitch of multiple notes; and it requires two unrelated gestures to play, one to select the key, another to fine-tune its pitch.

We seek to integrate expressive pitch control into the keyboard itself. There are millions of trained keyboardists who have each spent thousands of hours to become proficient, and an interface that builds on this wealth of training has significantly greater potential for user uptake than an unfamiliar design. However, existing training is also the chief obstacle for any new keyboard interface. Ideally, an extended keyboard should require minimal re-learning by the performer, and in particular, the addition of new capabilities should not interfere with traditional technique. Broadly framed, then, this paper addresses the question of how a familiar interface with a substantial body of user expertise can be extended while respecting users' existing knowledge.

BACKGROUND

Previous Enhanced Keyboard Work
Designers and musicians have long been seeking ways to add new dimensions of expressive control to the keyboard. Yang and Essl [30] use a 3D depth camera to capture gestures a performer makes above the keyboard; Brent [4] tracks pianist arm motion using an IR camera mounted above the keyboard. Both systems are used to control sonic parameters of the performance. Pianist Sarah Nicolls [18] worked with composers to create performances using light sensors, accelerometers and bio-sensors in addition to data from the keyboard.

Over the past century, several designs have sought to integrate additional control into the keyboard mechanism itself. Aftertouch (key pressure sensing) is commonly available on commercial keyboards, though it is rarely used to control pitch. The Ondioline (invented 1941) and the experimental Yamaha GX-1 (1973) both let the performer add vibrato to notes by shifting the keyboard mechanism from side to side [21, 25]. In the 1980s, Robert Moog and composer John Eaton developed a keyboard measuring the position of the performer's

CHI 2013, April 27–May 2, 2013, Paris, France.

fingers on the key surfaces [17], allowing continuous control of pitch and other sonic parameters of each note. The authors, in previous work [15], also developed a keyboard capable of continuous finger position measurements, and this work forms the hardware basis of this paper. The commercial EVO keyboard (`http://endeavour.de`) senses finger position within a 1x4cm region of each key, with an adjusted geometry to provide a wider playing surface on the black keys.

Why have none of these advances been integrated into mainstream keyboard design? The reason may have as much to do with human factors as the technology itself. In a newspaper interview, composer Eaton said of the Moog keyboard, "It's very difficult to play. But an instrument should be difficult to play. That's the only way to master musical materials, by overcoming these difficulties" [19]. Though using the location of finger-key contact as a control dimension is conceptually straightforward, it presents substantial practical difficulties in performance. Of course, many traditional instruments are also difficult to play, but beyond a certain complexity, the designer will encounter limitations of human motor control and cognitive bandwidth [23]. Jordà defines "musical instrument efficiency" as the ratio between musical output complexity and control complexity [11]. If the constraints of keyboard technique are not accounted for, enhanced keyboards run the risk of greatly increased control complexity in exchange for only modest gains in musical output.

Constraints of Keyboard Technique

Piano technique rests on two mechanical assumptions: that the velocity of the initial press, rather than actions thereafter, determines the sound of a note; and that a key can be pressed anywhere along its surface with similar results. Where the fingers touch the keyboard is thus primarily determined by the physical constraints of playing multiple notes at once, either simultaneously or in sequence.

From the interface designer's perspective, finger location is a "soft" constraint in that the performer could in theory place the fingers nearly anywhere while playing a passage. In practice, though, few performers will spend the time needed to master an instrument that demands drastic changes to their technique. The question thus becomes how to find the "space between the notes": the aspects of performer-keyboard interaction which are *not* constrained by existing technique and which can therefore be repurposed for new musical effects.

Analysis of existing performance technique is important to precisely identify its constraints. Motion analysis of piano playing has a venerable history extending at least as far back as Ortmann (1929) [20]. Recent analyses include video motion capture of piano performance [7, 13], examinations of piano "touch" as profiles of continuous key motion [2, 16] and movement analysis of the arm using accelerometers [8].

Beyond Music: Challenges of Training and Expertise

Extending the keyboard raises a broader issue of expertise, both as an enabler and a constraint in interface design. Players of an extended instrument, even one with a high ceiling of expressivity, may experience a temporary dip in ability due to unfamiliarity. Scarr et al. [26], examining expertise in a broader HCI context, observe that this dip can deter users from adopting a higher-performance interface and propose methods of smoothing the transition from novice to expert.

Analogous situations be found in efforts to extend other familiar interfaces, including on-screen QWERTY keyboards [6] and pen-based input [10]. In the latter case, understanding typical pen motion profiles used in writing can guide the creation of more natural interfaces [29]. Touchscreen input is a frequent target for augmentation with new sensor modalities, including pressure [27], finger angle [28], device tilt [24] and shear (sideways) force [9]. Cognitive bandwidth constraints also appear in other domains, including interfaces used while driving [5]. In all cases, new input methods are added to a familiar activity, and non-interference with the original task becomes important.

When new sensors are added to an existing interface, questions may arise of when a user *intends* to engage with a new modality, versus when the sensor data merely reflects a byproduct of familiar actions. For example, when eye-tracking is used as an input device, separating intention from involuntary eye movement is a challenge [22]. Similarly, most screen touches will exert a measurable amount of pressure, but pressure will not always reflect a deliberate decision by the user. Requiring consistent regulation of finger pressure at all times would make a touchscreen substantially harder to use, and consequently a pressure-enhanced screen might make use of the new data only in selected situations. For other interactive systems, including those employing gesture recognition in free space [3], every motion by the user is necessarily an input, making it critical to separate intentional actions from non-meaningful movement [12].

Paper Overview

The remainder of this paper presents our sensor system extending the keyboard, followed by an initial investigation of the constraints of existing keyboard technique. Techniques are presented for adding expressive pitch control to the keyboard which are evaluated in user studies with expert pianists. The conclusion examines implications for both keyboard performance and interfaces outside the musical domain.

We recently developed a capacitive sensing system for measuring the location of the player's fingers on the key surfaces [15]. Thin printed circuit board overlays adhere to the surface of an existing keyboard. Figure 1 shows the sensors attached to a weighted-key electronic piano. The shapes of the sensors reflect measurements of several acoustic and electronic instruments; experimentally, we have found very little variation in key dimensions among instruments.

Figure 2 shows the principle of operation. On each black key, the capacitance values of 17 discrete pads are measured (25 on each white key). The presence of the player's finger increases the capacitance, and by interpolating between pads, a spatial resolution of 1024 or more points in the lengthwise axis is achieved. Both black and white keys measure in the lengthwise (Y) axis; the front of the white keys also measure horizontal (X) finger position with 256-point spatial resolution. Further detail on sensor operation can be found in [14].

Figure 1. Capacitive sensors installed on a Yamaha Clavinova CLP-150.

SENSOR SYSTEM

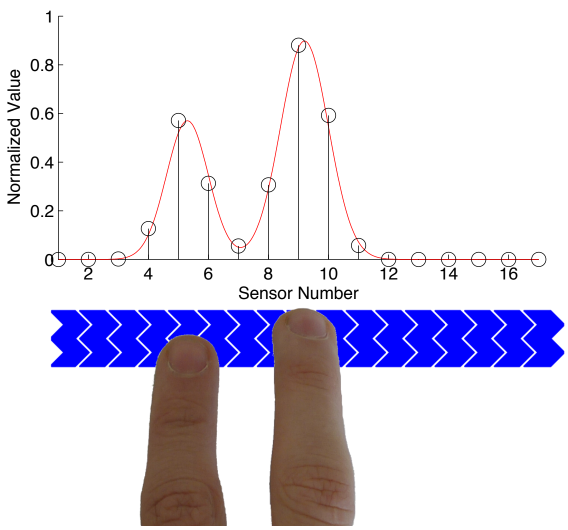

Figure 2. Simulation of touch location measurement by interpolating between discrete sensor pads.

Data Processing

Finger-key contact locations are scanned at 200 frames per second and transmitted to a computer by USB. Each incoming frame is compared to the previous frame for the same key to identify touches that have been added, removed, or changed in location. This data is combined with MIDI[1] from the underlying keyboard to determine finger location for each played note. In our analyses, we are primarily interested in the motion of the finger while a key is pressed, though the sensors can also determine finger contacts on unpressed keys.

Hardware Improvements

The sensor system used in this paper includes several improvements to the design in [15]. User feedback highlighted the square corners of the black keys as a source of discomfort; they are rounded in the current version (Figure 1). The white key sensors have been made 0.5mm narrower to reduce the likelihood of performers catching their fingers on the sides. The white and black colour of the current system preserves the familiar look of the keyboard. After considerable testing of various overlay materials, we found that the raw soldermask coating of the printed circuit boards received the most positive response from performers.

The current sensor system also incorporates optical reflectance sensors into the back of each key (Figure 1 bottom). When a fixed reflective object (e.g. a piece of white plastic cut to the vertical contours of the keys) is placed above the back of the keyboard, these sensors provide a measurement of continuous key angle. Each optical sensor is sampled at 1000Hz. To handle this increased sensor bandwidth, the original design's 8-bit microcontroller is replaced with a design based on a 72MHz ARM Cortex-M3.[2] The current paper does not use the optical sensors, however, instead restricting its focus to MIDI and capacitive touch sensor data.

PRE-STUDY: FINGER MOTION IN PERFORMANCE

We conducted informal testing with several performers examining finger motion in traditional piano performance, including recording one author's [AG's] performance of 8 short pieces from J.S. Bach's *Notebook for Anna Magdalena Bach* (1722-25). These investigations identified two constraints:

Constraint 1: Raw finger location cannot be used as a dimension of expressive control. Finger-key contact location is highly dependent on the fingering used, which is often determined by the sequence of notes in a passage. As Figure 3 demonstrates, the different lengths of the fingers (particularly the shorter thumb and pinky) manifest in different touch locations. Furthermore, to reach the black keys, the hand must be positioned higher on the keyboard.

Clearly, these hand positions do not represent free expressive decisions, though a certain amount of variation is possible. Though raw finger position may be usable for parameters whose setting has only a secondary impact on the sound (e.g. pluck location on a string [15]), raw position cannot be used for expressive pitch control.

Constraint 2: Finger motion on the keys is common, and the system must separate intentional from unintentional actions. Though for most notes the finger remains in place on the key while it is pressed, the touch location can drift for several reasons, including rolling the finger from the pad to the tip and shifting the hand to prepare for the following notes. These factors are further investigated in a study with 8 professional-level pianists, described later in this paper ("Examining Traditional Technique"; Figure 11).

One straightforward approach to adding expressive control to the keyboard surface is to consider the *relative* position of a touch as it differs from the point of onset. This investigation highlights the need for, at minimum, a threshold which separates smaller movements that result from standard technique

[1] Musical Instrument Digital Interface; `http://www.midi.org`

[2] STMicroelectronics STM32F1; `http://www.st.com`

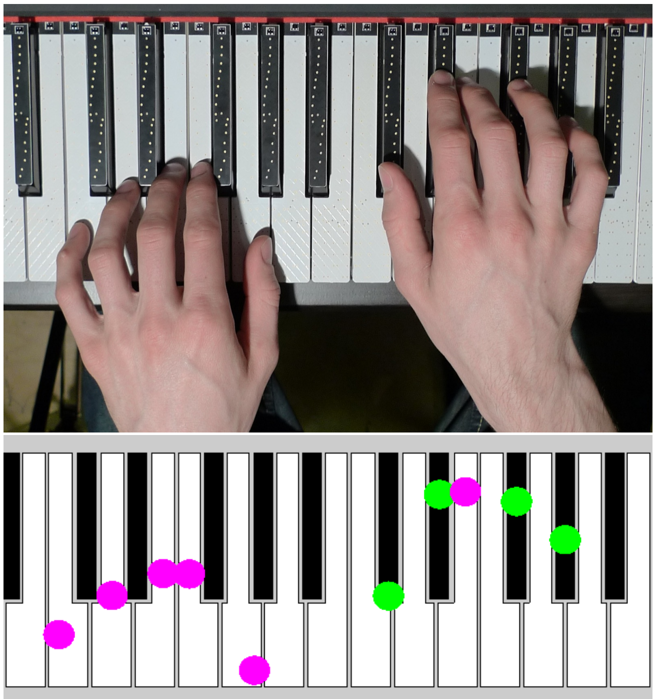

Figure 3. Relationship of hand position to touch data. Finger length and use of black keys strongly influence touch location.

from larger deliberate motions on the key surfaces [15]. In the next section we present techniques for continuous pitch control which further improve on the ability to separate deliberate expressive actions from byproducts of standard performance.

EXPRESSIVE PITCH CONTROL

We developed techniques for performing vibrato, pitch bends and slides between notes from the keyboard surface. Based on earlier feedback from performers, we chose gestures inspired by string instrument playing to control these techniques. While on violin family instruments, vibrato and pitch bends are performed on the same physical axis (lengthwise on the fingerboard), we chose to separate vibratos and pitch bends into orthogonal dimensions: vibrato on the horizontal (X) axis and pitch bends on the vertical (Y) axis (Figure 2). This was done to match the geometry of the keys (long and narrow) with the requirements of the gesture (large motion for pitch bends, small periodic motion for vibrato).

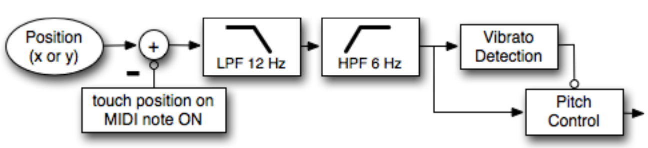

Figure 4. Data processing system for detecting and controlling vibrato based on finger motion. Arrows signify the flow of position data from the keyboard; connections with circles signify other data and triggers (e.g. changing parameters of a section).

Vibrato

To control vibrato (periodic oscillation in frequency), we had two primary goals:

1. Robust detection: vibrato should be easily activated, but with a minimum of inadvertent triggers
2. Accuracy of the centre pitch while performing the vibrato

By analogy to the violin, we chose a side-to-side rocking gesture to control vibrato, as we found sideways motion easier to control than front-to-back motion (confirmed in user studies below). However, since X-axis sensing is only available on the front of the white keys, rocking along the lengthwise (Y) axis was used for the remaining parts of the keyboard.

A simple approach might map relative finger position to pitch bend value, perhaps with a minimum threshold to activate. However, this fails to adequately address either objective: thresholds small enough to make the vibrato usable result in unacceptable numbers of false triggers, and the centre pitch is easily detuned as the finger rocks back and forth. Instead, we designed and fine-tuned a system (Figure 4) which filters out unidirectional movements and slow finger drift from its original position. Only when an oscillatory motion in a specific frequency range is detected does the pitch bending engage. A high-pass filter on the input position data ensures that the pitch always remains centred.

Position Data Filtering

The vibrato system consists of two stages: filtering and oscillatory motion detection. In the first stage, the finger position relative to note onset is low-pass filtered at 12Hz (1st-order) to smooth out irregularities in the sensor signal. A 1st-order high-pass filter at 6Hz eliminates drift from the finger moving from its original location. The filter frequencies were determined empirically, and their slow roll-off allows oscillatory motion below 6Hz to be detected while blocking near-constant input.

Oscillatory Motion Detection

Figure 5 shows the features of the filtered signal that are used to detect oscillatory motion of the finger. This section of the system has three parameters:

- **Threshold:** after the filtered finger position crosses zero, it must reach this value (in either direction) to initiate vibrato detection. When the threshold is reached, the algorithm stores the maximum deviation of the filtered finger position. This parameter is expressed as a fraction of the key width and can be set independently for X and Y axes. Default values: $x = 0.05$, $y = 0.05$.
- **Ratio:** after the filtered position crosses the threshold and the maximum value is stored, the finger position must then cross zero in the *other* direction (i.e. move right, then left or vice-versa). If X represents the maximum deviation in the first direction, the position must exceed $X * ratio$ in the other direction for vibrato to begin. Default values: $x = 0.3$, $y = 0.5$.
- **Minimum Detectable Frequency:** over a long enough time, the finger is likely to move in both directions even when vibrato is not desired. This parameter calculates a timeout (orange in Figure 5) between the first threshold cross and the second (ratio) cross in the opposite direction. Default value: 1.25Hz.

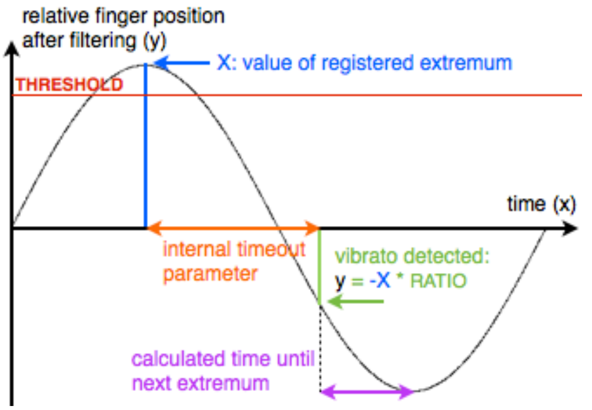

Figure 5. Parameters of the oscillatory motion detection algorithm. The waveform represents filtered finger position relative to point of onset.

Pitch Control and Vibrato Exit

Once vibrato is initiated, the pitch bend value is scaled to the filtered relative finger position. The total pitch bend range is a user-adjustable parameter, defaulting to 3 semitones. When the interval between zero-crossings of the filtered signal exceeds 1 / (minimum detectable frequency), vibrato is discontinued and the pitch bend returns gradually to 0.

Pitch Bends and Slides

We implemented a system for the performer to deliberately bend the pitch of a note, which can be used to make *portamento* (continuous-pitch) transitions between notes. The design goals were:

1. Robust detection of bend motions
2. Controllable intonation: the ability to play in tune easily while retaining control over nuances of pitch

Pitch bends can potentially span wide intervals, so it is appropriate to use the longer Y dimension of the key to control them. Since the physical motions are larger, the threshold of motion required for detection can also be larger, helping separate deliberate gestures from unintentional position variations. The total pitch range of the key is a user-adjustable parameter, with typical values ranging from 3 to 12 semitones.

Of the two goals, the second (intonation) proved more challenging: unlike string players, keyboardists are not accustomed to continuous pitch control, nor to the feel of a fretless surface on which the right pitch must be located. However we aim to allow a player unfamiliar with the system the ability to play in tune with only minutes of training. Therefore we developed a *pitch snapping* algorithm which causes the pitch to gravitate toward the steps of the chromatic scale.

Threshold Detection

Figure 6 shows the operation of the pitch bend system. Initially, the finger position is calculated relative to MIDI note onset. When the relative position crosses a pre-defined threshold, the bend algorithm engages. Our initial approach was to define the threshold as a fraction of a semitone, however for large total bend ranges (e.g. 12 semitones for the length of the key), this produces an unacceptably small threshold and interferes with later parts of the algorithm. Thus we use two thresholds, one based on total pitch bend distances (fixed at 0.4 semitones in either direction) and another based on a fraction of the key length (user-adjustable; see Figure 7).

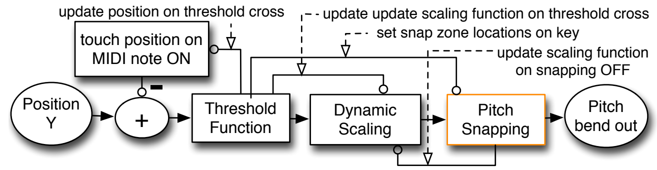

Figure 6. System for detecting and controlling pitch bends from finger motion in Y axis.

Dynamic Scaling

While the finger is within the threshold zone, no pitch bend occurs. Once it exits this zone, the pitch bend should engage gradually without an abrupt jump. Therefore the centre-point (zero bend) is recalculated to the edge of the zone. But to maintain consistency on the total pitch bend range of the key, the resulting map between finger location and pitch needs to be warped (Figure 7). This ensures that larger finger motions, which are less likely than fine adjustments to be executed by ear, have predictable points of arrival near the key edges.

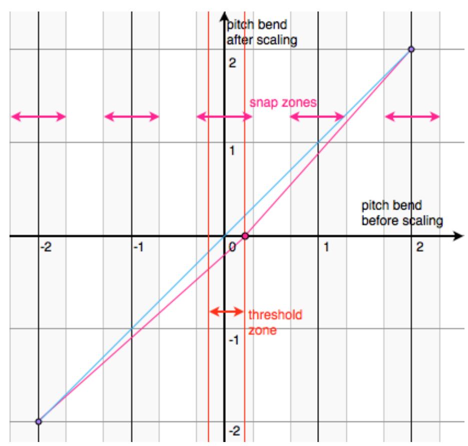

Figure 7. Threshold for pitch bend detection and resulting dynamic scaling of pitch.

Pitch Snapping

Figure 8 shows the algorithm for guiding pitch bends to the nearest semitone (right) and its conditions for activation (left). When a key is pressed, the algorithm calculates the positions along the Y axis which correspond to the steps of the chromatic scale. The *snap zones* are defined around these points, with a parameter (*snap zone size*) defining their width in semitones (Figure 9). For example, with a snap zone size of 0.3, any touch within 0.3 semitones of a chromatic pitch would engage the algorithm. The zone locations generally remain static throughout a note, even though the *dynamic scaling* function might slightly adjust the actual points corresponding to each chromatic pitch.

Snapping is engaged when a finger enters a snap zone and its speed of motion falls below a certain value. In this case, the note's pitch gets "pulled" toward the exact chromatic pitch. If

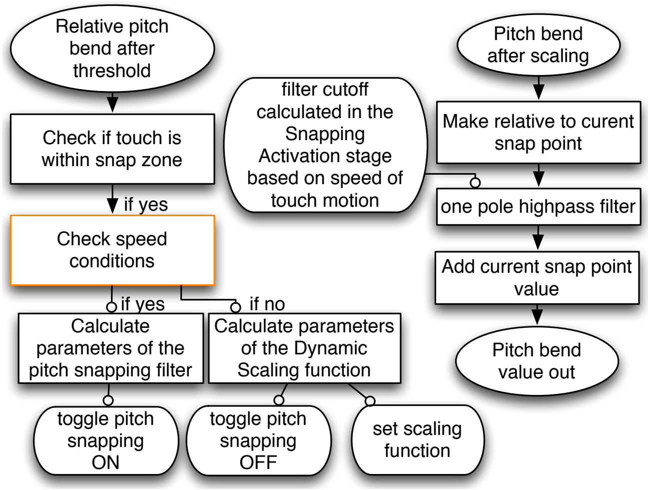

Figure 8. Left: conditions for activation and de-activation of pitch snapping, based on touch location and speed. Right: data flow in the pitch snapping algorithm, which pulls the pitch to the nearest semitone.

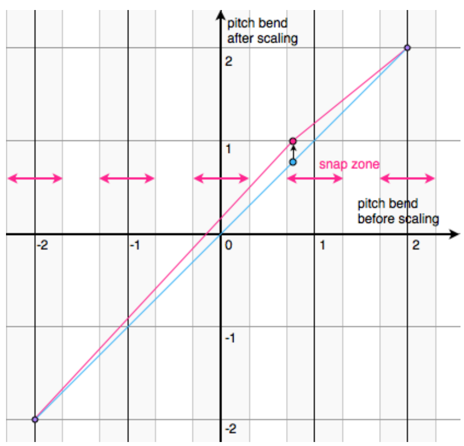

Figure 9. Illustration of snap zones (horizontal arrows) and dynamic scaling (comparison of scaled pink line to unscaled blue line).

the snap zone size is set to 0.5 semitones, then every point on the key surface is part of a snap zone, and it becomes nearly impossible to play out of tune: wherever the finger stops, the pitch is always pulled toward a chromatic step. Snap zone size is a user-adjustable parameter, and its effect is examined in the user studies below.

Parameters

In summary, the primary user-adjustable parameters are the *threshold* (expressed as a fraction of key length), the *snap zone size* (expressed as a fraction of a semitone) and the total *pitch bend range* for the whole key (expressed in semitones). Default values are 0.1, 0.4 and 7, respectively.

STUDY 1: EXAMINING TRADITIONAL TECHNIQUE

To better understand the constraints of traditional technique and to evaluate our expressive pitch control system, we conducted a study with 8 pianists (5 female, mean age 23.9, range 18-28). 7 were currently enrolled in or recently graduated from conservatory; 1 had recently graduated from (non-conservatory) university. The pianists had been playing piano for an average of 17.5 years (range 12-23).

Each session began with performances of three Inventions by J.S. Bach (BWV 772-786) and the Preludes in G major and B minor by Frédéric Chopin (op. 28 nos. 3 and 6). Each pianist was given a different selection of Inventions (out of 15 total); all played the same Preludes. The purpose was to capture the physical motions of the fingers during traditional piano performance; sensor data was logged, but no new techniques were added. In addition to musicological implications to be explored in future work, the study sought to verify or refute earlier informal findings about the aspects of finger motion that could be safely repurposed for expressive pitch control.

Mechanical and Tactile Considerations

Participants played on the instrument in Figure 1. After playing the Bach Inventions, each participant was asked to comment on how the feel of the instrument differed from a traditional piano. 4 of 8 players focused not on the sensors but the mechanical action of the electronic piano, which was felt to be less sensitive than an acoustic grand piano. 2 participants indicated that the action was realistic for an electronic keyboard, but none felt it was identical to a grand piano.

The most commonly identified drawback to the sensor system was the edges of the black keys (mentioned by 5 participants initially and 1 more on follow-up questioning), particularly the tendency for the fingers to catch on the sides when dragging the hand across the keyboard. The texture of the sensor surface was generally well-received. 5 participants found nothing at all unusual about it. 3 more commented on the texture, of which 1 found it objectionable (“too metallic”). None felt that the keys were significantly too sticky or too slippery.

Finger Motion

Detailed analysis of each performance is beyond the scope of this paper; however, the aggregate data revealed some interesting results. In addition to keys pressed, pianists frequently rested the hands on keys that were not played: in fact, fewer than 50% key touches were associated with a MIDI note. The location of touches that do not correspond to a MIDI note may be useful for locating the position of the hands or anticipating the player’s next move, factors which will be explored in future work. This result also suggests that using touches on non-pressed keys as an expressive control dimension would create problems for traditional technique.

Both vibrato and pitch bend algorithms rely on finger motion relative to its starting location. Figure 10 shows histograms of the amount of finger motion per note on the X and Y axes, expressed as a fraction of the key length/width. Table 1 presents details of finger motion broken down by participants and pieces. The greater amount of motion in the X axis is to be expected given the key is much longer than it is wide. Overall, we found that motion of the fingers toward the player’s body was more common (“pulling” motion; 63%) than motion away (“pushing”). Despite the different musical styles, we found little difference in the patterns of motion between Bach and Chopin, but more variation among players.

After our pre-studies, we settled on 0.1 as a default threshold for the pitch bend algorithm: finger motion within this range will not trigger a change in pitch. In our recordings of Bach

	X Mean	$X > 0.05$	$X > 0.1$	$X > 0.05$ (both dir.)	Y Mean	$Y > 0.05$	$Y > 0.1$	$Y > 0.05$ (both dir.)
All	0.153	75%	54%	8.9%	0.044	26%	11%	0.30%
Participant 1	0.124	66%	43%	8.4%	0.035	20%	7.9%	0.18%
Participant 2	0.171	80%	60%	8.9%	0.039	23%	8.7%	0.20%
Participant 3	0.154	81%	57%	8.3%	0.059	33%	17%	0.83%
Participant 4	0.126	66%	44%	6.2%	0.039	23%	9.0%	0.14%
Participant 5	0.171	78%	61%	13.6%	0.058	36%	18%	0.15%
Participant 6	0.185	82%	64%	9.2%	0.044	25%	11%	0.33%
Participant 7	0.149	76%	54%	10.4%	0.045	29%	10.3%	0.53%
Participant 8	0.144	73%	52%	8.2%	0.035	22%	7.1%	0.11%
All Bach	0.141	75%	52%	9.4%	0.044	27%	11%	0.16%
All Chopin	0.164	76%	57%	8.4%	0.045	25%	11%	0.47%

Table 1. Amount of finger motion on key surface over the course of a note, measured as the maximum deviation of the finger from its starting location. Table indicates the percent of notes deviating more than a specified value from onset; "both dir." indicates notes that deviate more than 0.05 in both directions from onset, likely to trigger a vibrato gesture.

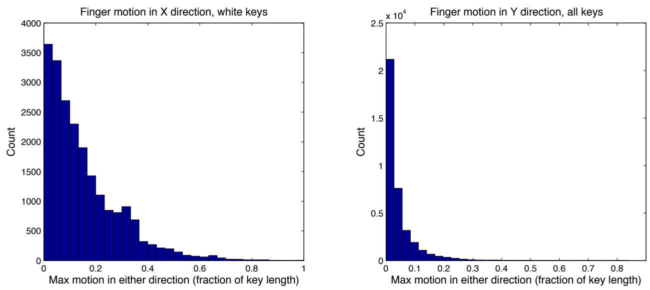

Figure 10. Histogram of maximum touch motion on a pressed note, X and Y axes.

and Chopin, 89% of notes fall below this motion threshold. 97% fall below a higher (but still playable) pitch-bend threshold of 0.2. This implies that with no change whatsoever to traditional technique, notes within the threshold would sound normally without inadvertent pitch bends. It is not necessarily a ceiling on performer accuracy: with an awareness of pitch bends and acoustic feedback from the instrument, it is likely that finger motion would be more precisely controlled.

Horizontal motion on the white keys is quite common, and accordingly, the vibrato algorithm requires evidence of a periodic motion to engage. We use motion in both directions of greater than 0.05 as a proxy for vibrato, though bidirectional motion will only trigger the vibrato if it happens in a short period of time. By this metric, approximately 9% of X-axis touches and virtually no Y-axis touches have the possibility of activating the vibrato function.

Example

Figure 11 shows an example case which involved large touch motions on the key surface. This example comes from the left hand of the B minor Prelude. The Y-axis motion reflects a combination of a deliberate technique of moving the hands toward and away from the keyboard and the mechanical constraints of shifting the hand across octaves. Future work will analyse these performances in more detail to identify scenarios in which the fingers are expected to move significantly.

STUDY 2: NEW TECHNIQUE EVALUATION

After the study of traditional technique, we evaluated our vibrato and pitch snapping systems with the same 8 participants. These tests were conducted on the previous generation interface [15] as the algorithms had been optimised in pre-studies for this sensor configuration.

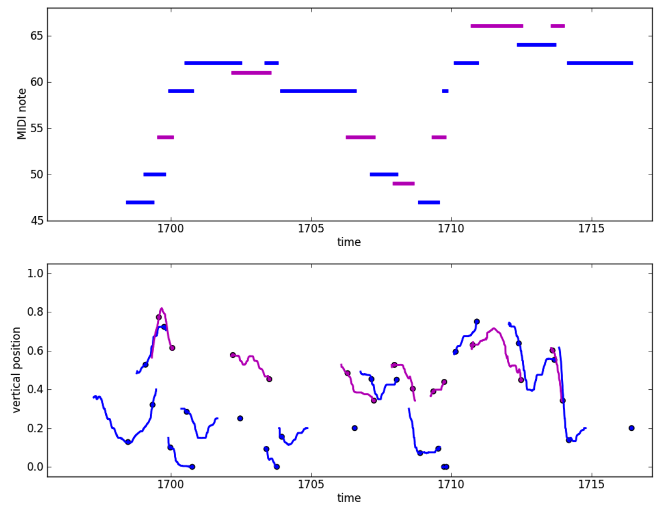

Figure 11. Example finger motion during a performance of the Chopin B minor Prelude, op. 28 no. 6. Top: MIDI notes over time; bottom: Y location of finger. Colours indicate black and white keys.

Vibrato

Each participant was initially asked to play vibratos on the white keys using side-to-side hand motion (X axis). After they became familiar with this technique, they were asked to play vibrato on the black keys with a front-to-back gesture (Y axis). Participants unanimously felt that the white key motion was natural and intuitive, but that the black key motion was not. One participant observed that the rolling wrist motion used on the white key vibrato is similar to wrist technique when playing rapidly alternating octaves. By contrast, several participants observed that the larger arm muscles were required to move the hand forwards and backwards; two participants attempted to play vibrato on the black keys by turning the forearm at a right angle.

Once the basic gestures were familiar, participants were asked to explore a range of parameter values for *sensitivity* and *pitch bend range*. Data was lost on one participant's choices, but on average, the remaining 7 chose a threshold of 0.027 and a pitch bend range of 2.2 semitones (edge-to-edge of key), a more sensitive but narrower range than default. Most participants felt that narrower vibrato was more musically appropriate, and that higher sensitivity allowed the vibrato to engage more easily while still offering reasonable robustness to inadvertent triggers.

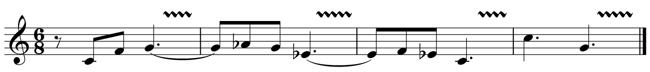

Figure 12. Test melody for selectively applying vibrato to certain notes.

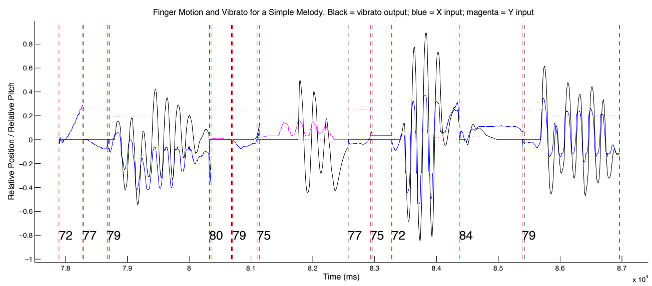

Figure 13. Example performance of a short melody with vibrato. Black line indicates output pitch; blue and magenta lines indicate X and Y relative position input. Numbers are MIDI notes.

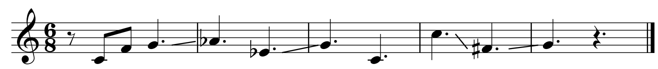

Figure 14. Test melody including pitch bends.

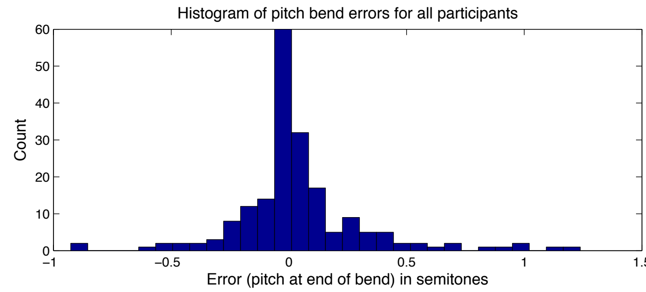

Figure 15. Pitch bend tuning errors aggregated across participants (absolute deviation between target and actual arrival pitch).

Results in Melodic Playing

After setting their preferred parameters, each participant was asked to practice and perform a simple melody (Figure 12). Notes marked with a wavy line were to be played with vibrato, and the other notes without. Of the notes marked with vibrato, 100% had vibrato added (32/32), and 9% (5/56) of notes not marked vibrato nonetheless caused the algorithm to engage. 4 out of 5 incorrect vibrato triggers took place on the high C (m. 4), where they were generally triggered by hand motion preparing for the following G.

An example performance is shown in Figure 13. Note that the pitch stays centred even as the relative touch location drifts, fulfilling one of our primary goals.

Pitch Bends

To evaluate the accuracy and usability of the pitch bending interface, each pianist was asked to perform 24 single-note bends. For the first set of 12 bends, the total range of the key (i.e. the largest possible bend) was 7 semitones (*s.t.*); for second set of 12 bends, it was 5 s.t. Each set of 12 consisted of three intervals (2 s.t. up, 3 s.t. down, 5 s.t. up; 1 s.t. down; 3 s.t. up; 4 s.t. down) with 4 different settings for the *snap zone* parameter (0, 0.15, 0.3 and 0.4 s.t.). Presentation order within each set was randomised.

Participants were given 15 seconds to practice each bend, at which point they were asked to execute it a final time for evaluation. Each gesture was performed on a single key, and participants were free to choose any key on which to play it. The existence of the pitch-snapping algorithm (and its changing parameters) was not revealed to participants during the test.

Following the individual note test, the algorithm was explained and participants were invited to explore the *threshold* and *snap zone* parameters, choosing values that felt most natural. Each participant was then given a melody (Figure 14) to play which incorporated pitch bends. Questions that this evaluation sought to answer included:

- To what extent can pianists accurately control the tuning of bent notes?
- Does the snapping algorithm improve tuning accuracy?
- Can pitch bends be triggered selectively only when desired, while other notes are left unaffected?

Analysis of the data began with manual annotation of the beginning of the bend (region of changing pitch), the end of the bend, and the end of the note. Since some participants exhibited slight pitch variations even upon arrival at the final note, the pitch was taken as a mean value between the end of the bend and just before the end of the note. The last few samples prior to note release were discarded to eliminate bias from the finger moving as it left the key. Error was calculated as the difference between the target pitch and the actual pitch, measured in semitones.

Results: Tuning Test

Figure 15 shows a histogram of tuning errors in the individual note test. The mean absolute error across all tests was 0.17 semitones, or 1.0% deviation in frequency. Accuracy varied widely by participant, from 0.067 semitone mean error (0.39% frequency error) to 0.28 semitones (1.6% frequency). Based on observation of the tests, it appeared that falling short of the target bend was more common than overshooting it; however, the final analysis did not show a significant effect one way or the other (46% of errors greater than 0.05 semitones overshot the target pitch).

Table 2 shows a breakdown of the tuning errors according to *snap zone size* along with pairwise t-test comparisons of each setting. The two larger snap zones (0.3 and 0.4 semitones) show significantly better pitch accuracy than the two smaller ones ($p < 0.007$). From this sample we do not observe a significant effect between no snapping at all (size 0) and the smallest snap (0.15), nor do we observe a significant difference between the two larger zones. We conclude that the pitch snapping algorithm achieves its goal of improving tuning, though we leave the fine-tuning of the zone size for future work (or perhaps to the performer).

Examining the results by target bend size, we observed the highest accuracy on the 1-semitone bend (Table 3) on the lowest accuracy on the 4- and 5-semitone bends, suggesting (but

Snap zone (s.t.)	0	0.15	0.3	0.4
Error (s.t.)	0.223	0.207	0.129	0.122
p-values for t-tests of zone size; bold = significant				
Snap zone (s.t.)	0	0.15	0.3	0.4
versus 0	-	0.74	**0.045**	**0.041**
versus 0.15	-	-	0.078	0.071
versus 0.3	-	-	-	0.88

Table 2. Pitch bend performance, tuning error according to snap zone parameter value.

Key range (s.t)	7	7	7	5	5	5
Interval (s.t)	2	-3	5	-1	3	-4
Error (s.t.)	0.17	0.14	0.24	0.067	0.17	0.24

Table 3. Pitch bend tuning error according to target interval size.

not proving) that small bends may be easier to execute. We did not observe a statistically significant difference in accuracy between upward bends (0.19 s.t. error) and downward bends (0.15 s.t. error; $p = 0.17$). We expected that we might observe an improvement in accuracy from beginning to end of the task as participants became accustomed to the gesture; however, this was not supported by the data. Comparing bends 2-6 against bends 20-24 (excluding the widely varying first attempt), the difference in accuracy (0.175 s.t. versus 0.123 s.t.) was not significant ($p = 0.23$). A larger sample size and longer training period would be needed to establish the extent to which pianists improve in accuracy over time.

Results: Melody

Prior to playing the melody, each participant set the *snap zone size* and *threshold* parameters to comfortable values. The mean choices were 0.24 semitones and 0.15 semitones, respectively (a larger threshold with smaller snap zone than default). However, it was unclear whether participants were able to fully internalise the effect of the *snap zone size* parameter within the limited period of exploration.

The results for the pitch bend melody were less consistent than for the melody with vibrato. One of the stronger performances is shown in Figure 16. While all notes marked with a bend had a bend applied, participants were often unhappy with their performance and would pause to replay a missed note, making a straightforward analysis of all performances difficult. In particular, participants found the leap to the high C followed by the large bend to F# (m. 4) awkward, as the finger had to be placed near the top of the upper C to allow a bend of 6 semitones down.

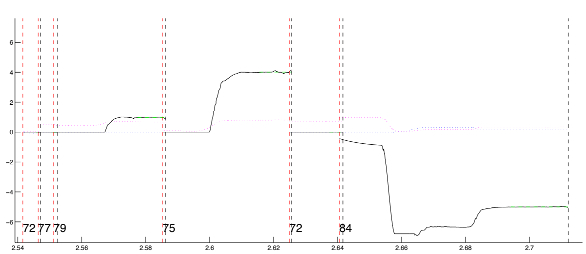

Figure 16. Example performance of melody with pitch bend, indicating relative pitch in semitones (black), raw X and Y input (dotted lines) and MIDI note numbers.

Discussion

The results of the vibrato and pitch bend tests indicate that these techniques are accurately playable by trained pianists. The melodic tests show that vibrato can be reliably applied to only selected notes, making it easily deployable in larger pieces of music.

Participants found the pitch bending in particular to be challenging, such that it would mainly find use in less technically complex passages. Since continuous pitch control is not a feature of traditional keyboard playing, several participants indicated that aural skills (hearing the right interval) was as significant a challenge as the physical motion. Nonetheless, the techniques were seen to be playable and useful. One participant, professionally active in theatre ensembles, indicated the techniques would be useful for playing other instrument sounds from the keyboard as theatre players are often asked to do. Two others mentioned the possibility of new music being composed for extended keyboard.

CONCLUSION: ENGAGING WITH EXPERTISE

Pianists spend thousands of hours mastering their instrument, in the process developing a highly specialised technique. It is not reasonable for a new musical interface to receive a similar amount of training, so existing technique should be considered a constraint just as much as basic mechanical principles. We have demonstrated a system which adds expressive pitch control to the keyboard which aims to be minimally intrusive to existing technique. Some adaptation will always be required for any new instrument (as any pianist-harpsichordist knows), but our results indicate that the new techniques are usable by trained pianists.

We offer two suggestions for designers (within or beyond the musical domain) seeking to add new sensor modalities to interfaces with pre-existing user expertise. First, before any behaviours are attached to new sensors, passive logging of the sensor data is useful to establish traditional patterns of interaction, with a focus on finding patterns that are *not* part of traditional use. In Benford's taxonomy of *expected*, *sensed* and *desired* interaction [1], passive observation can find patterns that are sensed and possibly desired, but not previously expected. Our vibrato example shows that useful patterns are often found from the temporal profile of sensor data rather than single values: every key press has a touch location, and these locations often change with time, but the particular case of an oscillating motion was not part of traditional technique.

Second, we suggest that a major problem with extending an existing interface is the potential for new techniques to be engaged unintentionally. Users may eventually adapt their technique to minimise unintentional engagement, but it could take several months of practice for this adaptation to fully develop. Sensor data from traditional use can provide a rapid and useful proxy for how much a new technique interferes with familiar use. We found that (algorithm setting dependent) fewer than 10% of notes in traditional performance would have triggered vibrato or pitch bends. This does not indicate a user's eventual false-positive rate, but shows a worst-case scenario where the user is unaware of the technique and receives no feedback from it. We hypothesise that, in general, lower trigger rates in this passive case will translate to a better eventual separation between traditional and extended techniques.

Future Directions

The sensor hardware in this paper represented a refinement over earlier versions, but user testing indicates that further changes would be useful. In particular, the ability to sense horizontal motion on the black keys would substantially improve the user experience of playing vibrato on these notes. Future investigations will also examine existing piano technique more closely, particularly the relationship between expressive intentions and physical motion on the keyboard.

ACKNOWLEDGMENTS

This work was supported by the UK Arts and Humanities Research Council through a "Digital Transformations in Arts and Humanities" grant (AH/J013145/1).

REFERENCES

1. Benford et al., S. Expected, sensed, and desired: A framework for designing sensing-based interaction. *ACM Transactions on Computer-Human Interaction 12* (2005), 3–30.
2. Bernays, M., and Traube, C. Piano touch analysis: A MATLAB toolbox for extracting performance descriptors from high-resolution keyboard and pedalling data. In *Proc. JIM* (2012).
3. Bevilacqua, F., Zamborlin, B., Sypniewski, A., Schnell, N., Guédy, F., and Rasamimanana, N. Continuous realtime gesture following and recognition. In *Gesture in Embodied Communication and Human-Computer Interaction*, vol. 5934 of *Lecture Notes in Computer Science*. Springer, 2010, 73–84.
4. Brent, W. The gesturally-extended piano. In *Proc. NIME* (2012).
5. Burnett, G. E. On-the-move and in your car: An overview of HCI issues for in-car computing. *Intl. J. Mobile HCI 1*, 1 (2009), 60–78.
6. Findlater, L., Lee, B. Q., and Wobbrock, J. O. Beyond QWERTY: Augmenting touch-screen keyboards with multi-touch gestures for non-alphanumeric input. In *Proc. CHI* (2012).
7. Goebl, W., and Palmer, C. Tactile feedback and timing accuracy in piano performance. *Experimental Brain Research 186*, 3 (April 2008), 471–479.
8. Hadjakos, A., Aitenbichler, E., and Mühlhäuser, M. Potential use of inertial measurement sensors for piano teaching systems: Motion analysis of piano playing patterns. In *Proc. i-Maestro Workshop on Technology-Enhanced Music Education* (2008).
9. Harrison, C., and Hudson, S. E. Using shear as a supplemental two-dimensional input channel for rich touchscreen interaction. In *Proc. CHI* (2012).
10. Hasan, K., Yang, X. D., Bunt, A., and Irani, P. A-Coord input: Coordinating auxiliary input streams for augmenting contextual pen-based interactions. In *Proc. CHI* (2012).
11. Jordà, S. Instruments and players: Some thoughts on digital lutherie. *J. New Music Research 33* (2004).
12. Lee, H., and Kim, J. An HMM-based threshold model approach for gesture recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence 21* (1999), 961–973.
13. MacRitchie, J., Buck, B., and Bailey, N. J. Visualising musical structure through performance gesture. In *Proc. ISMIR* (2009).
14. McPherson, A. TouchKeys: Capacitive multi-touch sensing on a physical keyboard. In *Proc. NIME* (2012).
15. McPherson, A., and Kim, Y. Design and applications of a multi-touch musical keyboard. In *Proc. SMC* (2011).
16. McPherson, A., and Kim, Y. Multidimensional gesture sensing at the piano keyboard. In *Proceedings of the 29th ACM Conference on Human Factors in Computing Systems (CHI)* (Vancouver, Canada, 2011).
17. Moog, R. A., and Rhea, T. L. Evolution of the keyboard interface: The Bösendorfer 290 SE recording piano and the Moog multiply-touch-sensitive keyboards. *Computer Music Journal 14*, 2 (Summer 1990), 52–60.
18. Nicolls, S. Twenty-first century piano. In *Proc. NIME* (2009).
19. NJ Star-Ledger. Waiting to be heard, Wanted: Someone to bring the successor to the Moog synthesizer to life.
20. Ortmann, O. *The Physiological Mechanics of Piano Technique*. Kegan Paul, Trenc, Trubner & Co., 1929.
21. Paradiso, J. A. Electronic music: new ways to play. *IEEE Spectrum 34*, 12 (1997).
22. Poole, A., and Ball, L. Eye tracking in HCI and usability research. In *Encyclopedia of Human-Computer Interaction*, C. Ghaoui, Ed. 2006.
23. Pressing, J. Cybernetic issues in interactive performance systems. *Computer Music J. 14*, 1 (1990).
24. Rahman, M., Gustafson, S., Irani, P., and Subramanian, S. Tilt techniques: investigating the dexterity of wrist-based input. In *Proc. CHI* (2009).
25. Rothstein, J., and Metlay, M. P. Products of interest: The musician-machine interface to MIDI. *Computer Music Journal 14*, 2 (1990), 73–83.
26. Scarr, J., Cockburn, A., Gutwin, C., and Quinn, P. Dips and ceilings: Understanding and supporting transitions to expertise in user interfaces. In *Proc. CHI* (2011).
27. Stewart, C., Rohs, M., Kratz, S., and Essl, G. Characteristics of pressure-based input for mobile devices. In *Proc. CHI* (2010).
28. Wang, F., and Ren, X. Empirical evaluation for finger input properties in multi-touch interaction. In *Proc. CHI* (2009).
29. Xin, Y., Bi, X., and Ren, X. Natural use profiles for the pen: An empirical exploration of pressure, tilt, and azimuth. In *Proc. CHI* (2012).
30. Yang, Q., and Essl, G. Augmented piano performance using a depth camera. In *Proc. NIME* (2012).