Motion and Context Sensing Techniques for Pen Computing

Ken Hinckley[1], Xiang 'Anthony' Chen[1,2], and Hrvoje Benko[1] *
Microsoft Research, Redmond, WA, USA[1] and Carnegie Mellon University Dept. of Computer Science[2]

ABSTRACT
We explore techniques for a slender and untethered stylus prototype enhanced with a full suite of inertial sensors (three-axis accelerometer, gyroscope, and magnetometer). We present a taxonomy of enhanced stylus input techniques and consider a number of novel possibilities that combine motion sensors with pen stroke and touchscreen inputs on a pen + touch slate. These inertial sensors enable motion-gesture inputs, as well sensing the context of how the user is holding or using the stylus, even when the pen is not in contact with the tablet screen. Our initial results suggest that sensor-enhanced stylus input offers a potentially rich modality to augment interaction with slate computers.

Keywords: Stylus, motion sensing, sensors, pen+touch, pen input

Index Terms: H.5.2 Information Interfaces & Presentation: Input

1 INTRODUCTION
Pen input, particularly when used in tandem with multi-touch [4,19,41,43,46], offers a compelling input modality for precise, expressive interaction with direct-input devices such as slates, tabletops, and electronic whiteboards. However, on most current systems, the posture and motions of the stylus can only be sensed when it is brought within hover range (~1 cm) of the display. When the stylus is away from the screen, all the rich nuances of how users hold and move the pen— or even whether they are holding it at all— typically remain unobserved.

In the context of mobile devices, inexpensive low-power sensors have afforded a new generation of motion gestures (such as tilting, flipping, and shaking) [20,26,27,28] as well as context sensing-techniques [14,16,30,40]. These sensing modalities also open up practical new possibilities for enhanced stylus interaction including in-air gestures, contextual sensing, and hybrid input techniques that combine touch and motion [12,18] with pen input.

Previous work has explored a number of auxiliary degrees of freedom for stylus input, including pressure [23,24], tilting [38,39,44], and rolling [3] in various combinations [15,45] as well as sensors including accelerometers [35,42] and capacitive grip-sensing [32,34]. Given the ubiquity of low-power inertial sensors in mobile phones and other devices, we believe including sensors as an active component of battery-powered stylus devices will be practical in the near future. Thus inertial sensing offers a fresh aspect of pen interaction that demands further research and exploration.

In this paper we adopt the perspective that pen input will most likely develop in the context of devices that support both pen and multi-touch input. Furthermore, many slates already include inertial sensors. Bringing together these diverse capabilities in pen + touch interfaces with sensor-enhanced pens could enable a new generation of slates with rich and nuanced input vocabularies that interaction designers can leverage in novel and creative ways.

* {kenh, benko}@microsoft.com; xiangchen@acm.org

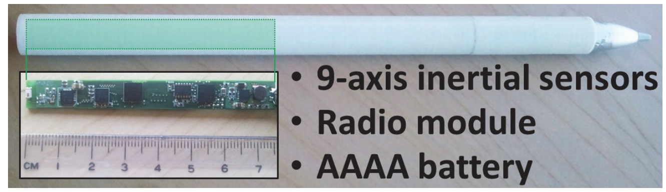

Fig. 1 Our wireless prototype has accelerometer, gyro, and magnetometer sensors in a ~19 cm X 11.5 mm diameter stylus.

Our system employs a custom pen augmented with inertial sensors (accelerometer, gyro, and magnetometer, each a 3-axis sensor, for nine total sensing dimensions) as well as a low-power radio. Our stylus prototype also thus supports fully untethered operation in a slender profile with no protrusions (*Fig. 1*). This allows us to explore numerous interactive possibilities that were cumbersome in previous systems: our prototype supports direct input on tablet displays, allows pen tilting and other motions far from the digitizer, and uses a thin, light, and wireless stylus.

2 RELATED WORK
Our research draws from three main areas of previous work: (1) stylus input enriched with additional degrees of freedom; (2) multi-modal pen, touch, and motion input techniques; and finally (3) related sensing techniques in the context of mobile devices.

2.1 Stylus Input Enhanced with Additional Capabilities
In these sections we consider stylus input enhanced with roll, tilt, and pressure sensing; stylus motion beyond the sensing range of the digitizer itself; and grip-sensing techniques for pens.

2.1.1 Roll, Tilt, & Pressure Sensing Stylus Techniques
Researchers have explored auxiliary stylus input channels including various combinations of tilting, rolling, and pressure sensing. However, most previous efforts only consider such pen movements in close proximity to the digitizer, and often restricted their worldview to a context where the pen had to perform all actions. Our work explores new hybrid techniques that leverage pen motion-sensing in combination with the growing prevalence of displays that support both stylus and direct multi-touch input.

Bi et al. [3] study pen rolling, including studies of the range of incidental rolling during handwriting and sketching tasks, as well as the practical range of intentional pen rolling gestures. They report that rolling can successfully trigger out-of-band interactions including rolling for rotation, multi-parameter input, and mode selection. We also consider rolling gestures, but our users do not have to roll the stylus in close proximity to the display. Our techniques also interleave touch with pen rolling.

A-coord input [15] explores combined tilt and pressure input in an indirect (horizontal desktop) tablet setting for context menus, multi-parameter selection, and manipulation, and finds that users can successfully coordinate two auxiliary input channels. Tilt Menu [39] also explores context menus based on tilting the pen. Xin et al. [44] analyze the human ability to control pen tilt, leading to various pen-tilt based interactions and widgets. A follow-up study [45] explores naturally occurring pen pressure and tilt angles during writing; pressure is prone to false activation, whereas the azimuth angle of pen tilt is more robust. Tilt Cursor [38] provides a correctly oriented pen cursor during hover movements for enhanced stimulus-response compatibility. All these techniques require tilting the device while maintaining contact (or extreme proximity) with the digitizer.

2.1.2 Stylus Motions Sensed away from an Input Surface

Suzuki [35] uses an accelerometer-enhanced pen to sense "movements when the stylus is not touching the display" and note this affords opportunities to "take advantage of all the ways a pen-shaped device can be handled." This resonates with the approach we take here. They propose *shaking the stylus* to cycle through color palettes, and *rolling the stylus* to pick colors or scroll web pages. Our sensor-enhanced stylus also has an accelerometer, but employs a full suite of inertial sensors sampled at 200 Hz. We also consider a wider range of both foreground and background interactions, with an organizing design space.

XWand [42] employs a baton-like pointing device with fusion of multiple inertial sensors to enable three-dimensional pointing in a "smart room" environment. The user can gesture to objects in the room and speak voice commands. Systems with 3D spatial input, such as 3-Draw [29], or the above-tabletop multi-layer interaction techniques proposed by Subramanian et al. [33], employ stylus-like devices in free space, but require *absolute* tracking technologies that are impractical for mobile pen-and-tablet interaction at present. Techniques that employ the hover state of traditional pen digitizers (e.g. [1,10,13]) also make some use of above-surface interaction, but rely purely on sensing the *(x,y)* pen tip location rather than inertially-sensed motions of the stylus, and hence offer different capabilities.

2.1.3 Grip-Sensing

Several recent efforts have explored grip-sensing. Stylus barrels with multi-touch enable sensing finger gestures, potentially eliminating the need for pen barrel buttons for mode switching [32]. Specific grips can also be associated with particular pens or brushes [34]. Several systems employ inertial sensors in tandem with grip sensing to boost grip pattern recognition [21,34,37], whereas our work focuses on pen motions and gestures.

Researchers have also proposed palettes that fade in or fade out based on which devices the user touches [17], or devices that turn themselves on when the user picks up a mobile device [16]. We explore a related context-sensing technique based on pen motion.

2.2 Multi-Modal Pen, Touch, and Motion Input

Researchers have explored new approaches afforded by displays that support simultaneous pen and multi-touch input (e.g. [4,19,43,46]). However, much of the existing work on enhanced stylus input focuses entirely on pen motions or grips, while ignoring for the most part the possibility of direct-touch input that might complement the techniques. In this paper we strive to always consider pen stroke and motion inputs in the context of touch-sensitive displays, and we therefore explore hybrid techniques that incorporate simultaneous (as well as interleaved) pen, motion, and touch inputs on the display.

Conte [41] is a good example of a system that combines pen tilt with direct-touch input. A Conte-crayon-like stylus senses which corners, edges, or sides of the crayon come into contact with a tabletop display. Thus, by tilting or rolling the Conte crayon *while it remains in contact with the display*, the user can fluidly switch between a number of tools, modes, and other input controls. Conte also combines direct multi-touch input with stylus orientation, allowing users to tap a finger on a control while holding or "tucking" the stylus in the palm. Conte cannot sense tilt or other motions once the stylus breaks contact with the display, however.

Combinations of touch and motion have also been considered for mobile devices. "Sensor synaesthesia" explores techniques where touch cues the system to recognize shaking and other device motions [18]. GripSense [12] demonstrates how combined accelerometer, gyro, and vibrotactile damping enable devices to sense nuanced touch gestures and motions. PhoneTouch [31] uses accelerometer bumps to determine when a mobile phone contacts a multi-touch tabletop display. WalkType [11] and 1Line Keyboard [8] both employ finger taps in combination with a tablet's accelerometer to enhance touch-screen typing.

Our system is the first, however, to explore the combination of pen input, direct-touch input, and motion-sensing inputs that are not restricted to the near-proximity sensing range of the digitizer.

2.3 Sensing Techniques for Mobile Devices

Many motion-based gestures have been demonstrated for mobile devices and smartphones. For example, several related techniques for mobile devices employ tilting, often in combination with button presses to avoid false-positives [14,16,18,26].

Some motion gestures are unique enough to leave "active" for recognition even without any explicit trigger. For example, DoubleFlip [28] demonstrates that a flipping gesture (analogous to the rolling degree-of-freedom for pens) offers a robust delimiter for motion gestures. Whack Gestures [20] use hard contact forces as an out-of-band signaling channel for mobile devices. And TimeTilt [27] demonstrates jerking movements as a way to trigger state transitions in mobile interfaces. In our work, we thus explicitly considered sensor-enhanced stylus input techniques that explore each of these types of motion.

Inertial sensors are oft-used for mobile context sensing [14,16,30,40]. Analogous techniques for sensor-enhanced stylus input may exist, as hinted at by previous work [34,35].

3 DESIGN SPACE OF MOTION-BASED PEN INPUT

Our research was motivated by the observation that people hold, wiggle, tilt, pick up and put down, and otherwise manipulate their pens [19,34,41], and that sensing these varied postures and motions might increase the bandwidth, naturalness, or expressiveness of user's interactions with pens and slates. On current tablets, the system has no idea if the user is even holding the pen any more once it leaves the immediate proximity of the screen, never mind more nuanced aspects of posture, movement, and context. Current user experiences with these devices are still very far from leveraging the full physicality and expressiveness of implements found in the real world such as fountain pens, Conte crayons [41], brushes [9,34], and other mechanical intermediaries.

Having decided to pursue this line of investigation, we employed the design dimensions of the taxonomy below (*Fig. 2*) to guide a systematic exploration of the space. We devised this taxonomy of motion-based pen input techniques to frame our work, differentiate it from previous efforts, and suggest new possibilities for future techniques, much in the tradition of past input research [6,7,18,43]. However, the primary motivation for our techniques arose from human-centered limitations of expressiveness, naturalness, or efficiency that we observed in both application-specific interactions and more generic cross-application scenarios for pen devices.

Our design space, adapting aspects of previous taxonomies [5,6,18], arranges pen-motion techniques predominantly by the ***property sensed*** (columns) vs. ***distance above the digitizer*** (rows).

For *property sensed*, we group motions into techniques that employ *device orientation*—whether absolute or relative—versus other *motion types,* which broadly categorize various styles of motion input, including *hard contact* forces, sensing particular *patterns or gestures* of movement, as well as techniques that use *stability* (lack of motion) to trigger actions or specific contexts.

For the *distance above the digitizer* of a stylus (or other mechanical intermediary, e.g. [2,31]), we consider three categories: *contact*, within *hover* range of the digitizer, or *far* away from the digitizer, as suggested by Buxton's 3-state model of input [5]. Hover range may vary by technology, but is typically 10-15mm; we treat everything beyond hover as the *far* category.

The ***activation mechanism*** further subdivides the rows according to what action (if any) the user must take to enable motion input. The activation mechanism can be *none* (the motion input is always active), a finger *touch* on the screen to specify the object a motion input acts on, or pen tip *pressure* (applicable only when the mechanical intermediary contacts the digitizer). Note that in principle, one could further distinguish touches on an object vs. other touches anywhere on the screen; however, as discussed later, at least in the context of triggering explicit motion gestures, we discovered that users found non-targeted touches to be somewhat cumbersome and counter-intuitive, so we decided not to distinguish these types of touch in our taxonomy.

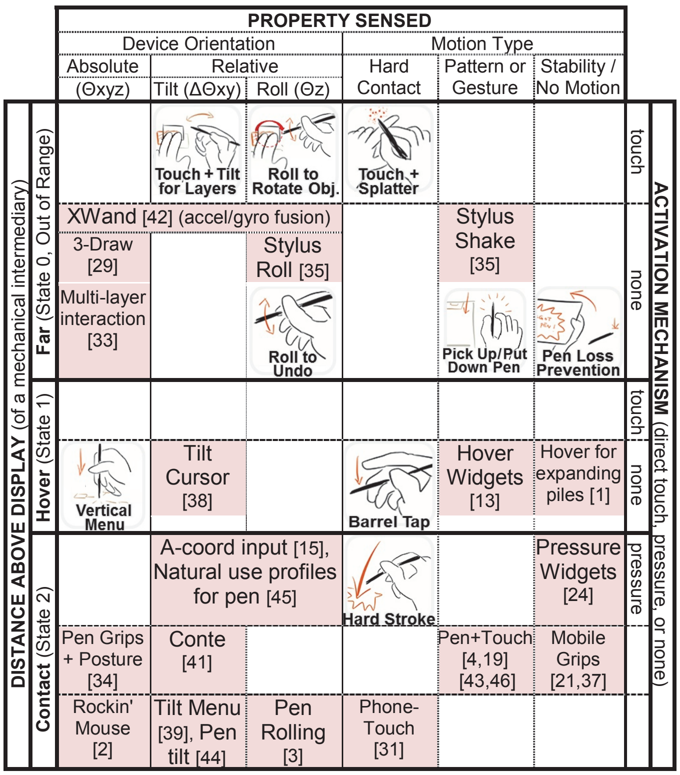

Fig. 2 Design space of motion-based pen input, organized primarily by property sensed (columns) versus distance above display (rows) where the corresponding motion gestures may be triggered. Cells in pink highlight previous motion-sensing work.

Cells shaded in pink highlight previous work, whereas the sketches denote techniques from this paper. This immediately makes clear that most previous techniques have emphasized device orientation in contact with the digitizer (i.e. the bottom left cells in the table), whereas we focus our efforts on gestures that take place above our display tablet's digitizer, as well as more diverse types of stylus motion-sensing gestures. For example, of the previous work exploring pen-motion gestures far from the input surface, both [42] and [29] consider stylus-like devices as pointers in 3D space, whereas only [35] considers motion gestures produced by a stylus that is also used for direct input on a display. Likewise, [33] and [1] consider above-tabletop interaction with layers, but [33] relies on absolution 3D sensing, and [1] employs traditional tablet hover. Neither use inertial motion sensing.

To focus on stylus motion-sensing techniques, this taxonomy does not currently include capacitive multi-touch *grip sensing* as a property sensed, although it clearly could via the addition of a column for grip-sensing. We note, however, that grip sensing could alternatively be treated as an *activation mechanism*, e.g. as a cue to trigger the software to look for particular motion gestures in any of the three states (contact, hover, or far/out-of-range). These extensions would be interesting to explore in the future, but such techniques are beyond the capabilities of our current stylus hardware and the scope of interactions considered by this paper.

Furthermore, note that this design space treats pen-tip pressure as an activation mechanism, i.e. as a sub-row of *Contact (State 2)*, and not as another *property sensed*, because tip-pressure only occurs when the pen tip contacts the display. If future pens were to incorporate a pressure-sensitive grip-sensing substrate around the pen barrel, however, then as discussed above for capacitive grip-sensing, this could be treated as a *property sensed* as well.

Finally, the design space emphasizes that few previous techniques [38] have considered pen tilt or motion in the *hover* state, likely because most tablets only support a narrow hover range of 10-15mm. Nonetheless, there may exist techniques akin to Hover Widgets [13] or Tracking Menus [10] that could leverage this space, particularly if future tablets can sense the *(x,y)* location of the pen tip at greater heights. Our *Vertical Menu* and *Barrel Tap* techniques provide examples that make use of device posture and motion signals even within a limited hover range.

4 HARDWARE & SOFTWARE SYSTEM COMPONENTS

do technique that interleaves pen motion and touch inputs, in this red by a single AAAA battery. With our current design (not optimized for power consumption at all) the battery lasts for about two hours of continuous use. An Atmel XMEGA 32A4U microcontroller collects the sensor data and runs our firmware. For inertial sensing we use the STMicroelectronics L3G4200D three-axis MEMS gyroscope, as well as their LSM303DLHC three-axis accelerometer and magnetometer module. For wireless connectivity we use a 2.4GHz Nordic Semiconductor nRF24L01 transceiver operating at 2Mbps.

Our firmware in the pen samples the sensors at 200 Hz and wirelessly transmits to a dongle on a host computer. The dongle requires a desktop PC, so we relay the sensor data to the tablet.

4.1 Stylus Components and Mechanical Design

All of the pen components— sensor board, AAAA battery, and the pen tip itself— fit in a 3D-printed casing. We actually constructed two separate pen prototypes, one for Wacom pen digitizers, and another for Ntrig integrated pen-and-touch digitizers, so that we can test our sensor-enhanced pen on a variety of commercially available Tablet PC and slate computers.

Fig. 3 The tapered Wacom **(top)** and cylindrical Ntrig **(bottom)** hardware prototypes of our sensor-enhanced stylus device.

The **Ntrig version** of our pen (Fig. 3) has an 11.5mm diameter, and is 18.9cm long. It has a smooth cylindrical shape that works well for motion gestures such as rolling the stylus. The internal pen components make the **Wacom version** of our pen (*Fig. 4*) slightly larger, at 12mm diameter x 21.2 cm long. Also, the casing for this stylus is tapered (from 9mm wide near the tip, reaching the 12mm diameter 6.2cm up the barrel), rather than cylindrical.

We implemented our techniques on an Asus EP121 Windows 7 slate with a 31 cm diagonal display, two-point capacitive multitouch, as well as a Wacom electromagnetic pen digitizer. By default Windows 7 disables all standard touch events when the pen comes in range, to prevent false touch inputs (e.g. incidental palm contact). Thus to support simultaneous pen and touch input, we use raw HID input events to handle the touch inputs outside of the normal Windows event mechanisms. We then get pen events using the Windows RTS (real-time stylus) interface.

We also attach a Phidgets Spatial 3/3/3 motion sensor to the tablet. At present we only use this for our *Pen Loss Prevention* context-sensing technique; our other techniques currently assume that the tablet is lying flat on the table. However, it would be straightforward to generalize our techniques to correct for the tablet orientation in future work (e.g. as demonstrated by [34]).

5 INTERACTION TECHNIQUES

We employed our design space to help guide the techniques that we chose to implement. In particular, we include a breadth of stylus properties sensed in terms of the *motion type* (denoted by the columns of Fig. 2). Furthermore, we concentrate our effort on techniques in the *Far* state, with a number of examples that integrate *touch* inputs. We also experiment with representative examples from the *hover* and *contact* rows of the design space.

We implemented our techniques in a sketching application, as this represents a natural test-bed for pen computing. We therefore designed semantically appropriate mappings for each gesture within the context of inking and sketching tasks. We also sought to enable more expressive forms of pen-and-touch input. Nonetheless, in a different application context such as active reading [36] or mathematical sketching [46], different gestures or mappings might prove desirable. With this in mind, we strove to explore a variety of both *application-specific interactions* (e.g. our Touch + Spatter technique) as well as *generic cross-application commands* (e.g. *Roll to Undo*, *Vertical Menu*), with a moderate bias towards generic interactions that are applicable across a wide range of pen computing tasks. Having said this, however, much of the potential appeal of pen-based interfaces may derive from the more physical and idiosyncratic, highly task-specific ways in which a stylus can be brought to bear in support of creative work.

In the following sections we describe our techniques, including context sensing, pen motion gestures away from the display, motion gestures combined with touch input, and close-range motion gestures (i.e. in hover range or in contact with the display).

5.1 Context Sensing Techniques

Sensing the motion and resting states of both the stylus and the tablet itself offers a number of opportunities to tailor the user experience to the context of the user's naturally occurring activity.

5.1.1 Tool Palette Appears & Disappears with Pen

On tablet computers, there is often a tension between having tool palettes and other UI controls on the screen at all times, versus employing the entire screen for showing the user's content.

The tools of interest while handwriting or sketching may be different than those of used to browse content or reflect on work-in-progress. Thus our stylus tool palette fades in quickly when the user picks up the pen. This palette includes tools such as color chips to change the pen color, as well as controls that change the mode of the stylus (eraser, highlighter, lasso selection, inking, etc.). When the user puts down the pen— or holds the pen extremely still, such as when thinking intensely about the current drawing— the palette slowly fades out to minimize distraction.

We sense lifting the pen as a transition from a state where the pen is not moving at all to a state of motion above a fixed threshold. Our implementation uses the gyroscope because it is sensitive to subtle motions. If the three-axis sum-of-squares gyro signal exceeds 36 deg/s we consider the pen to be moving, and the palette fades in over the course of one full second. When the pen motion falls below this threshold, the palette fades back out over the course of five seconds. If motion resumes before this fade-out finishes, the palette quickly fades back in to full opacity.

For interaction with the palette itself, we also found that it was important for the palette to respond to either pen taps *or* finger taps, as users often interleaved these modalities to select the current pen tool and color. This echoes earlier recommendations that system controls such as palettes and menus should treat pen and touch interchangeably [19]. For this reason, it is also possible to call up the palette using touch alone, by tapping on it with a finger (necessary if the pen is currently not in use, for example).

5.1.2 Pen Loss Prevention

One problem with pens in general, and dedicated stylus devices with active electronics in particular, is that the user can lose the pen– in which case the tablet might as well be a touch-only device. The user can easily leave the stylus on their desk, or forget to put it back in the tablet's pen holster after a meeting.

To address this problem, we designed a sensing technique to remind the user if the system observes the tablet moving away without the pen. Since both our pen and tablet have motion sensors, and they are in communication with one another, it is straightforward to infer whether or not they are moving together.

If the tablet starts moving, and continues moving while the pen remains stationary, then a large message ("**Forgot the pen?**") appears on the tablet's display (*Fig. 4, left*). The message then fades away (it is non-modal and does not have to be explicitly dismissed by the user). This serves to remind the user, without getting in the user's way if the reminder is unheeded. The technique also illustrates how sensing the motion states of both the tablet and pen can help provide a more complete picture of the system's state. In future work, we plan to explore augmentation of this purely visual feedback with auditory and tactile feedback on the tablet, as the user may not be looking at the screen.

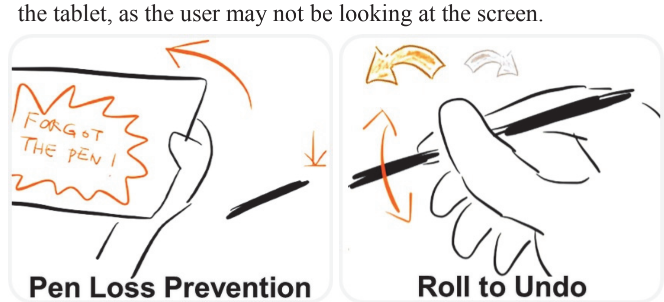

Fig. 4 Moving the tablet away without the pen brings up a reminder (left). Twisting the pen barrel triggers the Roll to Undo gesture (right). The user can then tap the feedback icons that appear to repeatedly invoke either *Undo* or *Redo* actions via touch.

5.2 Always-Active Pen Motion Away from the Display

We implemented only one explicit gesture that is always-active: our *Roll to Undo* gesture, which uses a quick rolling motion of the pen (i.e. twisting the pen sharply around the long axis of barrel).

However, several of our techniques— including the context sensing techniques above, as well as the pen motion gestures combined with direct touch input that follow— rely on sensing the motion of the pen while it is away from the display. Therefore, the ability to sense pen activity at a distance from the display enables many of the pen-motion techniques that we explore in this paper.

5.2.1 Roll to Undo / Finger Tap to Redo

Since previous papers have demonstrated that rapid pen rolling can offer a distinct gesture for pen input, we experimented with this gesture in our prototype as well for a *Roll to Undo* gesture (*Fig. 4, right*), but with the twist that the pen does not have to be on or near the screen while rolling, and furthermore incorporating direct-touch input as part of the interaction sequence.

We observed that rolling feels asymmetric: it is easier to roll the stylus in one direction than the other. For right-handers, rolling the pen forward (clockwise) is easier than rolling it back; for left-handed users, the preferred directions are reversed. Therefore, in our present prototype, we elected to support only one rolling command, *Undo*, which we trigger regardless of which direction the user rolls the pen. The gesture is recognized via the gyroscope as a quick motion (exceeding 300 deg/s over a 25ms interval) around the long axis of the pen, with minimal excitation of the other axes of the gyro during the motion, followed by rotation back to the original pen orientation within 250ms.

When we recognize this rolling gesture, feedback appears on the screen that shows the user they have activated the *Undo* command. However, our technique also allows the user to then *tap on the feedback icon again* to repeatedly invoke the *Undo* command if desired. This is faster than rolling the pen repeatedly, yet the pen rolling gesture saves the user the distraction of finding the *Undo* command in a menu and the wasted movement of going to the edge of the screen to invoke it.

We also added an icon for *Redo* to this feedback, so that the user may tap to perform *Undo-Undo-Redo* command patterns, for example. In this way we also expose a *Redo* capability without complicating the motion gesture design. Since *Redo* is a less common command than *Undo*, we felt this offered a productive design trade-off. Furthermore, note that we designed tapping on the feedback icons with *one-handed operation* in mind (i.e., the user can roll the pen, tuck it between their fingers [19,41], and then tap on the icons with the same hand). In this way our *Roll to Undo / Finger Tap to Redo* technique interleaves stylus motion and touch input in a hybrid design that takes full advantage of the properties of each interaction modality, and keeps the pen close to the user's working space and, thus, the locus of attention.

5.3 Pen Motion Gestures Combined with Direct Touch

Unlike the *Roll to Undo* technique that **interleaves** pen motion and touch inputs, in this section we describe three gestures that employ **simultaneous** touch and pen motion inputs: *Touch + Spatter*, *Touch + Tilt to Reveal Layers*, and *Touch + Roll to Rotate*. We also offer some general observations on using a finger touch to trigger explicit stylus-motion pen gestures in this context.

5.3.1 Touch + Spatter

Artists working in water media often employ a technique of rapping a loaded brush on the finger to produce spatters of paint on the paper. Such effects can produce natural-looking textures for foliage and landscapes, for example [9].

We implement a corresponding touch + pen-motion gesture that mimics this physical gesture (*Fig. 5*). The user touches the screen with a finger, and then strikes the pen against that finger to produce spatters. Note that, given the limited hover-sensing range of our tablet, the pen remains out-of-range (more than ~1cm away) when the user performs this gesture. Therefore, the tablet does not know the actual *(x,y)* location of the pen tip. Our system therefore produces spatters (in the currently selected pen color) centered on the finger contact point. The result feels quite natural.

Our implementation detects the acceleration peak corresponding to the pen strike and uses the amplitude of the peak to determine the number and transparency level of the spatters, how large the individual spatters are, and how far they scatter from the contact point. The semi-transparent spatters allow the colors to mix with one another in a natural-looking manner. Furthermore, to prevent possible incidental activation, at present our implementation does not respond to isolated strikes. The user must strike the pen to finger several times to begin the spattering effect. This results in a short delay before the paint spatters begin.

The resulting technique offers a good example of a unique effect made possible by our novel combination of pen-motion gestures with touch input. This illustrates how new sensing modalities can build on the existing skills and habits of users who may be passionate about a particular application or type of content (e.g. watercolor painting).

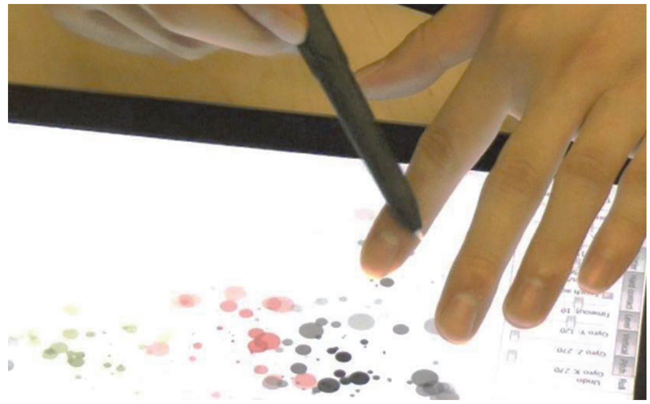

Fig. 5 Color splashes produced by our **Touch + Spatter** technique mimic the spattering of a loaded brush in water media.

5.3.2 Touch + Tilt to Reveal Layers

A common problem in graphical layout applications (e.g. PowerPoint, Photoshop, etc.) is working with multiple layered objects that occlude one another [25]. To address this problem, we explored the gesture of touching an object and then pitching (tilting) the pen back to reveal a list of the layered objects in z-order (*Fig. 6, left*). The user may then tap on the objects in the list to reorder them.

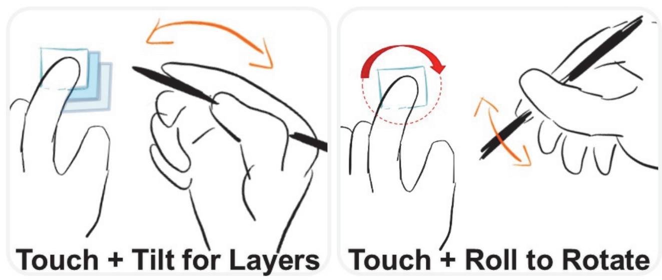

Fig. 6 The user holds an object while tilting the pen to trigger Touch + Tilt to Reveal Layers **(left)**. Holding an object and rolling (twisting) the pen enters the Touch + Roll to Rotate mode **(right)**.

In this example, the touch component of the gesture serves a double purpose. First, touching the screen activates the pen pitching motion for recognition; in our preliminary pilot testing we found that if pitching the pen were always active as a motion gesture, it was far too prone to inadvertent activation. Limiting the motion gesture to contexts where the *holds a finger on a stack of objects* avoids this limitation. Second, touching the screen here also identifies *which objects* the motion gesture applies to.

We implemented this gesture as a fairly specific motion; the user must tilt the pen tip away from the screen, and then back towards the screen, within a limited time-window. This makes the gesture more distinct, while still keeping it easy to perform.

5.3.3 Touch + Roll to Rotate

To gain further insight into combined touch and pen-motion gestures, we specifically chose to implement an additional technique that would contrast a pen-motion gesture that requires *touch activation* to an identical pen-motion gesture *without* touch (i.e., a gesture with an activation mechanism of *none*, as in Fig. 2).

Since previous work has demonstrated that rolling offers a distinct motion distinguishable from other incidental pen motions [3,35], we devised the *Touch + Roll to Rotate* technique as a touch-activated complement to our *Roll to Undo* technique, which is an "always-on" motion gesture.

If the user touches an object and rolls the pen, this enables a rotation mode (*Fig. 6, right*). The user can then dial his finger to precisely rotate the object. For this technique we implemented a relaxed version of rolling that accepts more incidental motion in the other axes of the gyroscope, as compared to the always-active *Roll to Undo* version of the gesture. A more permissive approach was acceptable for this gesture because touch activation reduces the chances of false positives while the user handles the pen.

5.3.4 Observations on Combined Touch + Pen Motion

In addition to the techniques above, we also experimented with touch as a general "mode switching" mechanism to activate motion gestures. For example, we tried a variant of *Roll to Undo* that required the user to hold a finger anywhere on the screen for the gesture to be recognized. While this eliminates many opportunities for incidental pen motions to be recognized as false-positives, we also found the need to hold a finger "somewhere on the screen" cumbersome in this context. The reason is that, unlike pen motion gestures that act on an object and therefore take that object (or finger location) as a parameter, when we tried this particular gesture we found that touching the screen with the nonpreferred hand purely for the sake of enabling pen motion gestures felt unnatural—an extra step that undermined the benefit of offering a pen-motion gesture in the first place.

Therefore, in our subsequent work, we limited our exploration of touch + pen motion gestures to techniques that *acted on the specific object or (x,y) location* touched by the user. In retrospect, most examples of multi-modal touch gestures advocated by previous work (such as pen + touch [4,19,43,46] and touch + motion for mobiles [12,18]) also appear to exhibit this property.

5.4 Close-Range Motion Gestures (in Hover / Contact)

We explored two techniques in the hover range, the *Vertical Menu* and *Barrel Tap*, both of which use the sensed *(x,y)* location of the pen tip to determine where to bring up a menu, and one technique that requires direct contact with the display, *Hard Stroke*.

5.4.1 Vertical Menu

The *Vertical Menu* is one of only two techniques we implemented in the *hover* distance category of our design space; it is also our only technique that uses a fixed orientation of the stylus relative to the display. An advantage of this posture is its salience: users can easily orient a pen vertically via the kinesthetic sense.

We noticed that people typically angle a pen relative to the screen while writing or drawing, and that therefore a stylus brought perpendicular to the screen might offer a distinct cue to trigger a mode that brings up a localized menu, thereby enabling efficient interleaving of stroke input with menu invocation.

Menus that appear at or near the locus of interaction, typified by Tracking Menus [10] and Marking Menus [22], can save round-trips to tool palettes at the edge of the screen. Such localized menus are particularly valuable for frequent commands, as well as contextual commands such as *Copy* and *Paste* that integrate object selection and direct manipulation with commands [22].

The *Vertical Menu*, then, is a marking menu triggered when the pen is held vertical and stationary for a short time within the hover range of the display (Fig. 7*, left*). The user may then bring the stylus into contact with the display and stroke in any of the eight cardinal compass directions to invoke a command. If the user continues to hold the pen stationary for 1/3 second, the marking menu pops up to reveal the mapping of stroke direction to command. The Vertical Menu thus combines stylus motion and pen-tip stroke input in the same technique.

The user can stroke the pen to invoke familiar commands in expert mode without waiting for the visual menu, but our present implementation does require a distinct stationary pause of about 100ms before it will accept marking commands, to avoid the potential for false positives. If the user places the pen vertically over an object, the marking menu includes object-specific commands (e.g. Copy, Paste) that take the current object and screen location as operands. This nicely integrates object selection and command invocation into a single fluid pen gesture.

Fig. 7 A vertical posture triggers the **Vertical Menu** from the hover-state (left). **Barrel Tap** is a hard contact force perpendicular to the pen shaft (middle). **Hard Stroke** is a pen stroke that begins with a hard contact force against the screen (right).

We detect the vertical posture of the pen using a combination of the accelerometer and gyroscope. We use the accelerometer to estimate the orientation of the pen, but even if the accelerometer indicates that the pen is near-vertical, we ignore it when the gyroscope indicates that the pen is still moving. In this way, we avoid false positive activation of the marking menu if the user briefly passes through the vertical pose while handling the pen.

In principal the Vertical Menu technique could be attempted with existing tilt-sensing pens (by looking for a vertical posture when the pen comes in range). In practice the very limited proximity sensing range of current tablets leaves very little time for the system to react to the gesture before pen-down, and furthermore the motion of the pen prior to this moment is unknown, making it extremely difficult to infer if the vertical posture is a transient motion or not. We therefore believe the technique would not likely work without our sensors and ability to sense the pen's posture as it approaches proximity sensing range.

5.4.2 Barrel Tap

Our sensor-enhanced stylus prototypes do not include any barrel buttons. We experimented with sensing *barrel taps*—that is, hard-contact finger taps on the barrel of the pen—as a way to "replace" mechanical button input. Much like Song et al. [32], this approach is motivated by the desire to support button-like inputs without resorting to the fixed-grip— and acquisition time— required to properly place a finger over a mechanical button.

Our implementation looks for an acceleration spike perpendicular to the long axis of the pen while the pen is stationary in the hover state (*Fig. 7, center*). This brings up a menu with commands specific to the object that the user hovers over, and centered under the pen tip. However, in early pilot testing we found that a straightforward implementation of this technique is too prone to false positive activation. We therefore decided not to evaluate it further with our test users, but a refined implementation that combines grip-sensing with motion-sensing would be interesting to try in future work.

5.4.3 Hard Stroke

We also tried a technique analogous to "hard tap" and "hard drag" techniques [18] proposed for touch input. In our implementation, bringing the stylus down hard triggers lasso mode, whereas normal pen strokes produce ink. We look for an accelerometer spike that corresponds to the pen-down (*Fig. 7, right*).

While this gesture felt promising, we found that it was difficult to choose a good accelerometer threshold to distinguish the "hard" contact from normal strokes, in part due to the varying angles at which a user might hold the pen when it comes into contact with the display. We therefore elected to leave this gesture out of our user tests, but again a refined implementation that corrects for the angle of the stylus relative to the tablet screen at the time of contact might produce better results.

6 PRELIMINARY USER FEEDBACK

We conducted an informal evaluation to gather feedback of people using our techniques. The intent here was not to produce a detailed quantitative study, but rather to gather some initial user reactions and feedback on our proposed techniques.

We recruited 8 participants, age 25-53 (median 37.5). Two were female, one was left-handed, and all were employed by the same institution but were not members of our research group. Each participant used a slate flat on a small table in front of them. The experimenter explained and briefly demonstrated each of the interaction techniques. We then asked participants to try out each technique. Test users responded to 7-point Likert scale questions, and discussed the best and worst aspects of each technique.

Please note that **all** motion gestures were active during the entire usability test, so that we could observe any false-positives or undesired interactions that might occur as the user progressed through each interaction technique.

6.1 Results

We noted the following comments, concerns, and issues for each of the techniques that we included in the evaluation.

6.1.1 Tool Palette Appears & Disappears with Pen

This technique was well-accepted. Users commented that “it made sense and was what I would have expected,” that it had an “instantaneous response when I picked up pen,” and that the technique was “simple and intuitive.” In 7-point Likert scale responses, participants strongly indicated that the technique felt natural (median response of 6 out of 7) with only one user rating it as low as a 4 (“neither agree not disagree”). However, several users did comment that they wanted the palette to fade out more quickly when the pen stopped moving, suggesting that our 5s fade-out (intended to avoid distraction) may be a bit too slow.

6.1.2 Roll to Undo

This was our least successful gesture. Most users had trouble articulating it, and many users also experienced false-positives for this gesture while performing other actions. In retrospect, we realized that in large part this was because we originally tuned our implementation of this gesture using our cylindrical Ntrig sensor pen, but due to the tapered barrel of our Wacom sensor pen, users found it difficult to perform a pure rolling motion without also tumbling some by accident. We expect this tumbling could have been avoided with a cylindrical pen casing. On the positive side several users found it “handy when it worked” and liked that they “didn't have to open up an additional menu” for *Undo*.

Users really liked the ability to tap on the feedback icons with a finger to repeatedly invoke additional *Undo* and *Redo* actions. Given these comments and the success demonstrated with rolling gestures in previous research [3,35], we are reluctant to dismiss this gesture altogether based on our current results. Revisiting this gesture with a refined implementation—including a pen with a cylindrical shape— may produce better results in future efforts, but our experience with it so far also flags the possibility that false positive activations may be a problem with this technique if it is implemented as an always-active motion gesture.

6.1.3 Touch + Spatter

Users enjoyed this gesture, commenting that it was a “cool new drawing ability” that “mimicked a gesture you might actually do.” When asked if “this technique felt like it required extra steps” all eight users disagreed (median 2/7). Because the corresponding physical gesture in the real world requires both a finger and an implement (a brush), users found the combination quite natural.

Some users found the delay before the ink starts spattering annoying, and a few users commented that they had to strike the pen too forcefully for the gesture to be recognized. Some users also felt that even though the effect was cool, spattering did not represent a common function that they needed on a pen-operated device (none of our test users were artists or designers).

6.1.4 Touch + Tilt to Reveal Layers

Most users found this to be a simple, distinct gesture that effectively supported “scoped search of objects at a point.” However, half of the test users expressed some desire to have a one-handed version of the technique. For this technique we again asked if it “felt like it required extra steps,” most users disagreed (median 2.5/7), indicating that overall users felt that touching the layered objects was a natural and integral step of the interaction. However, in this case, the median obscures a somewhat bimodal response to the technique, with three of our eight users strongly agreeing that holding a finger on the object felt like an extra step, while the other five users strongly disagreed.

Thus, while overall most users appreciated the ability to easily reveal layers, and found holding a finger to scope the gesture was something that they could easily learn to do, the combination of touch and tilting for this action didn’t feel quite as natural to users as our *Tilt + Spatter* gesture.

6.1.5 Touch + Roll to Rotate

Users had fewer problems activating the rolling motion for this gesture because of the more relaxed recognition, but three users *still* experienced difficulty rolling without unintended tumbling due to the tapered shape of our Wacom pen. Several users appreciated the “speed to do something that would take lots of menu clicks” but some users felt that they would have preferred a more direct-manipulation approach to object rotation. Thus, although the *Touch + Roll* gesture was more successful than the pure *Roll to Undo* gesture, opinion remained divided on the utility of our current semantic mapping for the gesture.

6.1.6 Vertical Menu

The *Vertical Menu* was well-accepted. Most users found it “responsive” and liked that they “didn’t have to navigate somewhere else” to bring up the menu. Users also particularly liked the context-sensitive Copy and Paste marks. However, some users disliked the vertical posture, finding it somewhat awkward to hold. We did observe a few instances where users accidentally triggered this technique, but it was not a common problem.

When asked if the *Vertical Menu* was “a fast way to call up a context menu” test users strongly agreed (median response 6/7) with only one user disagreeing (2/7). The latter user felt that bringing the pen to a stable vertical pose in the hover state took too long. However, this user also commented that the *Vertical Menu* “saves space on the screen by reducing UI clutter.”

6.1.7 Overall preference

Overall, the three functions that users listed as their most desired stylus motion gestures for general tablet use were having the tool palette appear & disappear with the pen, the *Vertical Menu*, and the *Touch + Tilt to Reveal Layers* gesture.

Somewhat surprisingly the least desired gesture was *Touch+Spatter*; although users liked the gesture, since our test users were not artists or designers, this did not represent a function important to their typical tablet use. The rolling motions were ranked poorly because of the difficulty many users had with rolling the tapered Wacom version of our sensor pen.

7 CONCLUSION AND FUTURE WORK

Taken together, the enhanced stylus input techniques that we explored in this paper illustrate a number of ways that we can:

- Adopt the perspective of context-sensing from mobile computing and apply it to motions of the stylus itself.
- Leverage the ability to sense tilting or other explicit motion gestures that occur close to the tablet as well as beyond the hover-sensing range of the digitizer.
- Integrate both pen motions and tablet motions for richer understanding of the interaction context, as demonstrated by our *Pen Loss Prevention* technique, for example.
- Explore expressive application-specific gestures such as our *Touch+Spatter* technique, as well as more generic interactions such as having the interface respond to picking up and putting down the pen, or cross-application gestures such as our *Vertical Menu* and support for *Undo-Redo* sequences.
- Combine pen stroke, finger touch, and motion-sensing together into hybrid, multi-modal techniques that enable new types of gestures for pen-operated tablet computers.

In future work we would like to deploy our sensors to collect sensor data that illustrates the variety of naturally-occurring pen motions, gestures, and postures that users exhibit "in the wild." By also observing the co-occurring multi-touch inputs and pen strokes, such an endeavor could yield interesting insights on what people do with the stylus during typical use, or suggest additional contextual signals with which to attack problems such as incidental palm contact while writing ("palm rejection"). Issues such the tendency of some users to wiggle or play with the pen when it is not in use (causing potential false activations), or fatigue from motion gestures, also need to be carefully examined.

We also anticipate that augmenting the barrel of the stylus with full capacitive multi-touch sensing (e.g. [32,34]) may enable sensing more robust gestures or additional contextual cues. Our current stylus hardware prototype already includes the requisite circuitry, but as of this writing we have not yet mechanically integrated a flexible multi-touch substrate with our stylus design.

Our exploration of stylus motion sensing builds on the existing literature of enhanced stylus input that, taken as a whole, suggests users can benefit from these new ways of interacting on or near displays. Many creative and compelling techniques that combine pen strokes, motion-sensing, and direct-touch input in unexpected ways likely yet remain to be discovered and refined.

8 References

[1] Aliakseyeu, D., Subramanian, S., Lucero, A., Gutwin, C. Interacting with piles of artifacts on digital tables. *Proc. AVI '06*. 2006: ACM.

[2] Balakrishnan, R., et al. The Rockin'Mouse: Integral 3D manipulation on a plane. *CHI 1997*.

[3] Bi, X., Moscovich, T., Ramos, G., Balakrishnan, R., Hinckley, K. An Exploration of Pen Rolling for Pen-Based Interaction. *UIST 2008*.

[4] Brandl, P., et al. Combining and measuring the benefits of bimanual pen and direct-touch interaction on horizontal interfaces. *AVI 2008*.

[5] Buxton, W. Three-state model of graphical input. *Human-computer interaction - INTERACT 1990*. Amsterdam: Elsevier Science.

[6] Buxton, W., Touch, Gesture, and Marking, in *Readings in Human-Computer Interaction: Toward the Year 2000*: Morgan Kaufmann.

[7] Card, S., et al. The Design Space of Input Devices. *CHI 1990*.

[8] Chun Yat Li, F., Guy, R., Yatani, K., Truong, K. The 1Line Keyboard: A QWERTY Layout in a Single Line. *UIST 2011*.

[9] Engle, N., *How to Make a Watercolor Paint Itself: Experimental Techniques for Achieving Realistic Effects*. 1999: Watson-Guptill.

[10] Fitzmaurice, G., et al. Tracking menus. *UIST 2003*.

[11] Goel, M., et al. WalkType: using accelerometer data to accomodate situational impairments in mobile touch screen text entry. *CHI 2012*.

[12] Goel, M., et al. GripSense: Using Built-In Sensors to Detect Hand Posture and Pressure on Commodity Mobile Phones. *UIST 2012*.

[13] Grossman, T., et al. Hover widgets: using the tracking state to extend the capabilities of pen-operated devices. *CHI 2006*.

[14] Harrison, B., et al. Squeeze me, hold me, tilt me! An exploration of manipulative user interfaces. *CHI 1998*.

[15] Hasan, K., et al. P. A-coord input: coordinating auxiliary input streams for augmenting contextual pen-based interactions. *CHI 2012*.

[16] Hinckley, K., Pierce, J., Sinclair, M., Horvitz, E. Sensing techniques for mobile interaction. *UIST 2000*.

[17] Hinckley, K., Sinclair, M. Touch-Sensing Input Devices. *CHI 1999*.

[18] Hinckley, K., Song, H. Sensor Synaesthesia: Touch in Motion, and Motion in Touch. *CHI 2011*.

[19] Hinckley, K., et al. Pen + Touch = New Tools. *UIST 2010*.

[20] Hudson, S., et al. Whack Gestures: Inexact and Inattentive Interaction with Mobile Devices. *Tangible & Embedded Interaction (TEI 2010)*.

[21] Kim, K.-E., et al. Hand Grip Pattern Recognition for Mobile User Interfaces. *AAAI/IAAI-2006: Innovative Applications of AI*.

[22] Kurtenbach, G., Buxton, W. Issues in combining marking and direct manipulation techniques. *UIST 1991*.

[23] Li, Y., Hinckley, K., Guan, Z., Landay, J.A. Experimental analysis of mode switching techniques in pen-based user interfaces. *CHI 2005*.

[24] Ramos, G., Boulos, M., Balakrishnan, R. Pressure Widgets. *CHI'04*.

[25] Ramos, G., et al. Tumble! Splat! Helping Users Access and Manipulate Occluded Content in 2D Drawings. *AVI 2006*.

[26] Rekimoto, J. Tilting operations for small screen interfaces. *UIST'96*.

[27] Roudaut, A., et al. TimeTilt: Using Sensor-Based Gestures to Travel through Multiple Applications on a Mobile Device. *Interact 2009*.

[28] Ruiz, J., Li, Y. DoubleFlip: A Motion Gesture Delimiter for Mobile Interaction. *CHI 2011*.

[29] Sachs, E., et al. 3-draw: A tool for designing 3D shapes. IEEE Computer Graphics and Applications, 1991. **11**(6): p. 18-26.

[30] Schmidt, A., et al. Advanced Interaction in Context. *Handheld and Ubiquitous Computing (HUC'99)*. 1999: Springer-Verlag.

[31] Schmidt, D., Chehimi, F., Rukzio, E., Gellersen, H. PhoneTouch: A Technique for Direct Phone Interaction on Surfaces. *UIST 2010*.

[32] Song, H., et al. Grips and Gestures on a Multi-Touch Pen. *CHI 2011*.

[33] Subramanian, S., Aliakseyeu, D., Lucero, A. Multi-layer interaction for digital tables. *UIST 2006*.

[34] Sun, M., et al. Enhancing Naturalness of Pen-and-Tablet Drawing through Context Sensing. *ITS 2011*.

[35] Suzuki, Y., Misue, K., Tanaka, J. Stylus Enhancement to Enrich Interaction with Computers. *HCII 2007, Human Computer Interaction, Part II*. 2007: Springer-Verlag Berlin Heidelberg.

[36] Tashman, C., Edwards, W.K. LiquidText: A Flexible, Multitouch Environment to Support Active Reading. *CHI 2011*.

[37] Taylor, B., Bove Jr., V. Graspables: Grasp-Recognition as a User Interface. *CHI 2009*.

[38] Tian, F., et al. The Tilt Cursor: Enhancing Stimulus-Response Compatibility by Providing 3D Orientation Cue of Pen. *CHI 2007*.

[39] Tian, F., et al. Tilt Menu: Using the 3D Orientation Information of Pen Devices to Extend the Selection Capability. *CHI 2008*.

[40] Verplaetse, C., Inertial proprioceptive devices: Self-motion-sensing toys and tools. IBM Systems Journal, 1996. **35**(3&4): p. 639-650.

[41] Vogel, D., Casiez, G. Conte: Multimodal Input Inspired by an Artist's Crayon. *UIST 2011*.

[42] Wilson, A., Shafer, S. XWand: UI for intelligent spaces. *CHI 2003*.

[43] Wu, M., et al. Gesture Registration, Relaxation, and Reuse for Multi-Point Direct-Touch Surfaces. *IEEE Intl. Workshop on Horizontal Interactive Human-Computer Systems*. 2006.

[44] Xin, Y., Bi, X., Ren, X. Acquiring and Pointing: An Empirical Study of Pen-Tilt-Based Interaction. *CHI 2011*.

[45] Xin, Y., Bi, X., Ren, X. Natural Use Profiles for the Pen: An Empirical Exploration of Pressure, Tilt, and Azimuth. *CHI 2012*.

[46] Zeleznik, R., Bragdon, A., Adeputra, F., Ko, H.-S. Hands-on math: a page-based multi-touch and pen desktop for technical work. *UIST'10*.