Air+Touch: Interweaving Touch & In-Air Gestures

Xiang 'Anthony' Chen, Julia Schwarz, Chris Harrison, Jennifer Mankoff, Scott E. Hudson
Human-Computer Interaction Institute, Carnegie Mellon University
{xiangche, julia.schwarz, chris.harrison, jmankoff, scott.hudson}@cs.cmu.edu

ABSTRACT
We present Air+Touch, a new class of interactions that interweave touch events with in-air gestures, offering a unified input modality with expressiveness greater than each input modality alone. We demonstrate how *air* and *touch* are highly complementary: touch is used to designate targets and segment in-air gestures, while in-air gestures add expressivity to touch events. For example, a user can draw a circle in the air and tap to trigger a context menu, do a finger 'high jump' between two touches to select a region of text, or drag and in-air 'pigtail' to copy text to the clipboard. Through an observational study, we devised a basic taxonomy of Air+Touch interactions, based on whether the in-air component occurs before, between or after touches. To illustrate the potential of our approach, we built four applications that showcase seven exemplar Air+Touch interactions we created.

ACM Classification
H5.2 [Information interfaces and presentation]: User Interfaces - Input devices and strategies, Graphical user interfaces.

Keywords
Touch input; free space gestures; interaction techniques; input sensing; around device interaction.

INTRODUCTION
A generation of mobile devices has relied on touch as the primary input modality. However, poking with a fingertip lacks immediate expressivity. In order to support richer actions, touch must be overloaded in time (e.g., long press), space (e.g., drawing an 's' to silence the phone) or configuration (two-finger tap is 'alt click'). These approaches suffer from one or more of the following issues: scalability of gesture set, time consuming to perform, "Midas" touch, and significant finger occlusion on small screens. Thus, there is an ongoing challenge to expand the envelope of touch interactions by combining it with new input dimensions that increases richness.

Recently, devices such as the Samsung S4 smart phone [22] have emerged with hover sensing capability. In-air (or "free-space") gesturing is an area of intense research (see e.g., [7, 17]). These interactions are attractive as they can utilize a space many times larger than a device's physical boundaries, allowing for more comfortable and potentially richer interactions. However, in-air gestures are typically treated as a separate input modality, rather than integrated with existing touch-based techniques. Further, in-air gestures suffer from the challenge of segmentation: little literature has discussed how to systematically separate intentional gestures from accidental finger movements.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
UIST '14, October 05 - 08 2014, Honolulu, HI, USA
Copyright 2014 ACM 978-1-4503-3069-5/14/10...$15.00.
http://dx.doi.org/10.1145/2642918.2647354

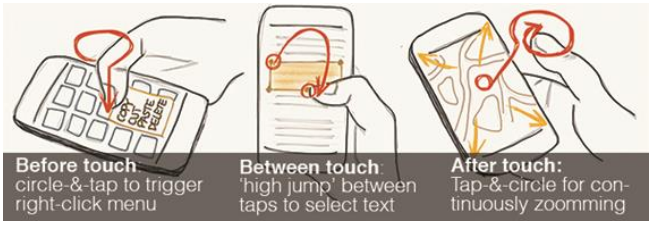

Figure 1. We propose that touch and in-air gestures be interwoven to create fluid and expressive interactions.

In this paper, we reconsider touch and in-air gestures beyond their individual domains. We propose a synthesis of these two input modalities, achieving interaction richness and robustness that neither can provide alone. Indeed, we found in-air and touch inputs to be highly complementary: touch is used to designate targets and segment in-air gestures, while in-air gestures add expressivity and modality to touch events. This Air+Touch modality outlines a class of interactions that enable fluid use of a device's screen and the space above it.

To explore this possibility, we start with a focus on the scenario of single-finger interaction, where a person uses his or her thumb or index finger to gesture in the air and also touch the screen. Through an observational study, we devised a simple taxonomy of Air+Touch interactions. We propose that in-air gestures can augment interactions *before*, *between* and *after* touch events. And in turn, touch events are used to segment in-air gestures and can also specify an on-screen target (e.g., a photo or map location). In-air gestures can be parameterized based on shape, velocity and/or time of a finger's movement. Figure 1 offers three examples, from left to right: 1) circle-in-air and tap an icon to trigger a context menu, 2) do a finger 'high jump' between two taps to select a region of text, or 3) tap and cycle the finger in air to continuously zoom a map.

RELATED WORK
Our work extends the input area from the touch screen to the space immediately above it, which is related to research that situates *interactions beside, behind* & *above* digital surfaces. For example, *SideSight* [3] uses infrared sensors to

track finger movements from the sides of a mobile device. Magnetic sensors have also been used to enable similar interaction styles in *Abracadabra* [6] and *MagiTact* [14]. Wigdor et al. explore the design space of a two-sided interactive tabletop surface [25]. *NanoTouch* [2] and *LucidTouch* [24] demonstrated that the back surface of a device can be used to increase the interactive area.

A number of research projects have focused on the space above interactive tabletops, such as Hilliges *et al.*'s *Interactions in the Air* [7], Marquardt et al.'s *Continuous Interaction Space* [17], and Banerjee et al.'s *Pointerable* [1]. In the realm of mobile devices, *HoverFlow* uses infrared sensors [16] and Niikura et al. use a high frame rate camera [18] to track hand/finger gestures above a mobile device. Marquardt et al. propose blending a digital surface and the space above it into a *continuum* wherein touch, gesture and tangibles can equally take place [17]. However, there is no discussion of mechanisms for segmenting in-air gestures (i.e., rejecting unintended finger movements). Further, the free-space and touch gestures generally co-exist, rather than being interwoven as we propose.

A natural next step is for researchers to explore in-air gestures in the space surrounding a device. Kratz et al. show that gestures above, beside and behind a mobile device yield better performance, compared to a virtual trackball, in manipulating 3D objects [15]. Jones et al. find that around-device free-space interaction can be as good as touch [13]. This work also defines 'comfort zones' around a device, which has strong implications for applying different sensor orientations. Samsung has shipped several basic in-air gestures with their Galaxy S4 [22]: A hand hovering on a lock screen shows time and notifications, and swiping left or right above a screen navigates a photo album.

Air+Touch also builds off of previous work that synthesized multiple inputs to create new interaction possibilities. *Pen+Touch* [11] synthesized pen and touch inputs to create new tools, such as using touch to hold a photo and pen to drag off and create a copy. *Motion+Touch* [10] combined touch with the motion sensing capability of a mobile device to yield touch-enhanced motion gestures and motion-enhanced touch. *Pen+Motion* [9] combined pen input with pen motions, enabling new gestural input abilities. Our work synthesizes touch and in-air gesture in several new ways. First, we provide an input structure to segment in-air gesture using touch, and to augment touch using in-air gesture. Second, air and touch interleave each other, yielding a permutation of input sequences than can richly parameterize interactions.

DESIGN FINDINGS FROM OBSERVATONAL STUDY

To ground and guide our initial exploration of Air+Touch interactions, we conducted a study to observe finger behavior above mobile screens when users are engaged in interactive tasks. We recruited 12 participants (5 female, ages 24-36). One participant was left-handed, one was ambidextrous, and all were regular smartphones users.

We asked each participant to perform a set of common tasks on a smartphone (e.g., compose a text message, navigate on a map). We videotaped the sessions and looked for patterns in how fingers hovered or moved in the space immediately above the screen. From this, we distilled a set of features that could translate into gestural input, while avoiding collisions (i.e., reducing confusion) with natural finger movements. Next, we discuss how these features can contribute to the design of in-air gestures, and further, how touches can be used as natural delimiters to segment these actions.

Air: Properties of Above-Screen Finger Movements

Participants in our study exhibited a wide range of in-air, above-screen finger behaviors. This included hovering over the screen between touches, retracting to the bezels when the screen needed to be read, and wiggling fingers when uncertain about what to do (such as while searching for a button). When discussing the contents of the screen, people also used their fingers to point and wave at content, or to gesture as they spoke, similar to how hand gestures are used in conversation. In particular, we focused on three main categories of finger movement behavior 1) path – the trajectory of fingers' movement, 2) position – fingers' particular positions above the screen, and 3) repetition –how users repeat certain finger movement.

These observations informed Air+Touch design in two ways. Foremost, they illuminated the kinds of above-screen finger movements users can comfortably reproduce, which we then adopted as part of our vocabulary of in-air movements. Secondly, it allowed us to craft a vocabulary of gestures that can be easily disambiguated from natural finger movements. Below are some exemplar findings that later informed our design of Air+Touch gestures:

Elliptical paths: Few of the finger motions we observed followed smooth, elliptical paths. This suggested that a circling action could be distinctive.

*Rectangular path*s: Similar to what Grossman et al. found in [5], few participants exhibited right angles in their finger trajectories. This suggested that paths with corners, such as an L-shaped gesture, could also be robustly recognized.

Leveraging height: Most users' finger movements occurred close to the screen. This suggested that in-air gestures with atypical height components could be disambiguated from typical interactions.

Using framing gestures: Whack Gestures [12] demonstrated that simple gestures (e.g., a *whack*) can yield expressive input when used as a framing feature (i.e., <*frame_gesture*> *primary_gesture* </*frame_gesture*>). Matched framing gestures can dramatically decrease the probability of false positives, even when the underlying gesture has a high error rate. In our study, we observed that users seldom touched the same location twice, except when scrolling, and in this case, rarely performed any finger movement of interest in the intervening time. This suggested that in-air gestures

could be performed between framing touches. Another possibility is to include framing within the air gesture, such as using the first in-air circle as the signal that triggers recognition for subsequent finger circling (similar to using consective "whacks" in [12]).

Touch: Delimiting In-Air Gestures

Even with a carefully designed in-air gesture set, the uncertain nature of free space gestures demands a more explicit way to signal when an interaction is actually taking place. Our observations suggested that touch events could serve as a powerful and intuitive delimiter.

In a typical interactive task, touch interleaves 'air' by bringing an in-air finger movement to a *closure* when the finger touches the screen or by *introducing* a new chunk of in-air movement as the finger disengages itself from the screen. Thus touch naturally segments in-air gestures into three possible categories: *before*, *between* or *after* touches. This allows the in-air gesture recognition engine to search only a small window of time for applicable in-air finger movements (i.e., instead of constant monitoring). In the remainder of the paper, these temporal categories serve as the organizing principle for the example Air+Touch interactions we created.

Air+Touch: A Gesture Vocabulary

Our observations also helped us to craft an initial vocabulary of in-air gestures, which can be delimited by touch events in three ways as described in the previous section. To further explore this space, we looked at existing applications and considered whether any of the Air+Touch gestures could be adopted to enhance the present interaction. This helped us come up with four applications covering a set of seven Air+Touch gestures (Figure 2, in red) that are representative (but not inclusive) of the entire design space.

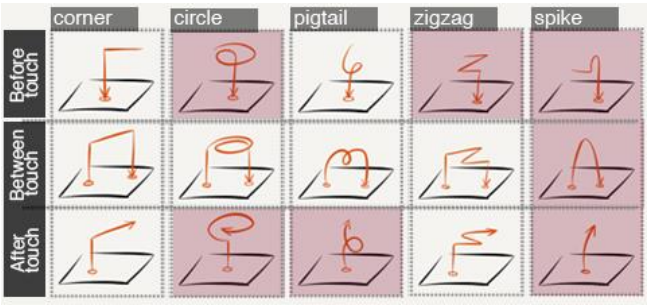

Figure 2. A proof-of-concept design space of Air+Touch gestures. We implemented seven of these techniques (red shading).

- *Corner*: the finger contains a 90-degree angle in air (on a plane perpendicular to the screen);
- *Circle*: the finger draws smooth, cyclical paths in the air.
- *Pigtail*: the finger draws a small loop along its in-air trajectory.
- *Zigzag*: the finger makes sharp 'turns' in air (on a plane parallel to the screen), e.g., drawing an 'L' or 'Z';
- *Spike*: the finger reaches a 'special' air position during its movement, e.g., reaching a position higher than the usual hover range, or a position outside the screen boundary.

AIR+TOUCH PROTOTYPE

There are an increasing number of devices featuring capacitive touchscreens able to track fingers in the air (i.e., hover sensing). At CES 2014, Synaptics demonstrated a prototype touchpad able to track fingers at up to 4cm away [23]. All indications suggest this technology will continue to improve and become more pervasive. Unfortunately, the sensing range on today's consumer devices is limited. For example, the Samsung Galaxy S4 has a tracking range of approximately 1.5cm.

Thus, in order to explore the full range of Air+Touch interactions that might be possible in a few years, it was necessary to build our own prototype. Although bulky today, our prototype served as a useful vehicle for exploration and investigation. We also used this platform to build seven demonstrations of Air+Touch interactions (Figures 2 and 6-12), which span our outlined design space and demonstrate the viability of our approach.

Hardware

Our prototype finger tracking system consists of a commercial smart phone and a PMD Camboard Nano [19] depth camera obliquely mounted to a common chassis (Figure 3). The Camboard Nano has a 90º×68º field of view and senses a 160×120px depth and infrared image from 5 to 50 cm at up to 90 fps. Finger tracking is performed on an external PC, and finger positions are sent to a mobile client via a wireless network. This setup allowed us to rapidly prototype ideas without having to instrument any customized hardware into the smart phone.

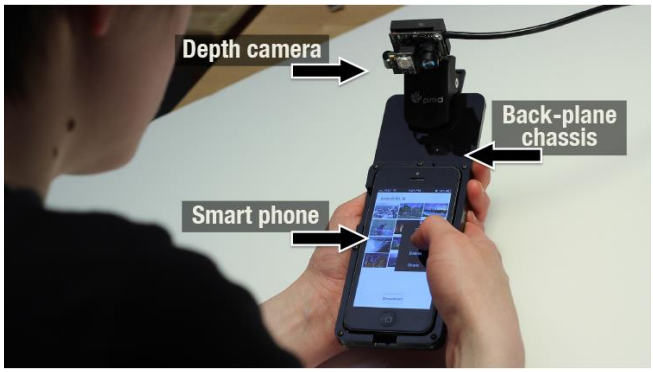

Figure 3. Our prototype smart phone uses a depth camera to simulate future, more advanced hover-capable devices. We used this setup as a vehicle for exploration and also as a platform to develop several Air+Touch augmented applications.

Finger Tracking

Our finger-tracking software is written in C++ and uses the OpenCV library. Since the geometry of the phone is known, we can perform simple volume-based background subtraction (Figure 4b). We also remove noise due to infrared reflection from the phone's screen (Figure 4c). Using this

image, we identify the largest blob in the scene and perform contour analysis. We assume the fingertip to be the farthest contour point from the blob centroid (Figure 4d). To help reject false positives, we only look at contours situated along the fingers' major orientation.

In cases when the finger is pointing towards the depth camera, the fingertip will not lie along a contour, but will rather lie inside the finger boundary. We detect this case by using our camera's infrared image; due to skin's high infrared reflectance (and the infrared emitter our depth camera employs), the fingertip will appear as a bright, roughly Gaussian spot. In this instance, we use the brightest spot as the fingertip position.

This process yields a camera-space, fingertip X/Y/Z position representing the point of interest during an Air+Touch gesture. We then transform this raw 3D coordinate to X/Y screen coordinates (in pixels), along with a Z value (distance perpendicular from the screen). This transformation matrix is computed using three known points on the phone's screen, selected in 3D camera space during a one-time calibration procedure (Figure 4, hollow dots). Finally, fingertip position is lightly smoothed with an exponentially weighted moving average.

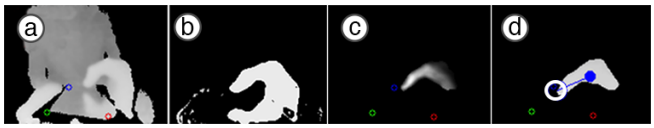

Figure 4. Finger tracking pipeline: a) raw image, b) background removal, c) noise removal & blob tracking, and d) fingertip localization.

In-Air Gesture Classification

Our system records 3D finger position at 20 frames per second and maintains a positional history of approximately one second. When a touch down event occurs, we run the $1 gesture recognizer [26] on the X and Y coordinates (as projected onto screen space) of the buffered finger positions. If a good shape match is found with sufficient size, a corresponding interactive event is fired. For in-air gestures after touch, we run the recognizer on the buffer after approximately one second following the touch up event. In the touch-down case, we also check to see if a reciprocal touch event happened within the last one second, and if so, interpret this as an in-air gesture being performed *between* two touch events.

To support in-air gestures that utilize Z distance (and not shape), we use a virtual plane situated 4cm above the screen as a threshold, providing something akin to a 3D crossing gesture. Each time this plane is crossed, a timestamp is recorded. If a touch event occurred within ±500ms, an interactive event is fired.

EXAMPLE AIR+TOUCH INTERACTIONS TECHNIQUES

Based on our design findings from the observational study, we developed a set of example Air+Touch interactions (Figure 5). To provide a use context for these interaction techniques, we created four host applications: photo viewer, drawing app, document reader, and map. Please also see our Video Figure.

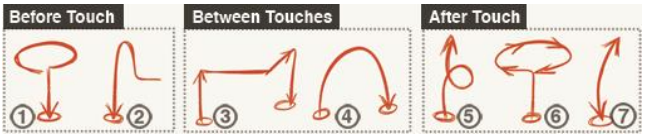

Figure 5. Air+Touch interactions can be characterized as air before touch (e.g., 1 - circle-in-air and tap, 2 - high-up and tap) air between touches: (e.g., 3 - draw an 'L', 4 - finger 'high jump') and air after touch (e.g., 5 - draw a 'pig tail', 6 - cycling in air, 7 - hovering.).

Air Before Touch

Unlike a mouse, touch (generally) only has one 'button'. This has led to a persistent need for additional modal mechanisms, such as touch-and-hold to invoke e.g., a context menu. Toolbars are also popular, but consume valuable screen real estate. To mitigate this problem, Air+Touch allows users to perform in-air gestures before or en route to touching the screen, as a way to parameterize the touch event. We offer two example interactions for this technique.

Circle-in-Air and Tap

In our photo viewer application (Figure 6), a user can trigger an image's context menu by performing an in-air circling motion (Figure 6a-c) immediately before tapping on a desired image (Figure 6d-e). The in-air gesture specifies the command (in this case, trigger context menu), while the touch specifies the item of interest (e.g., a photo). These two motions are combined into a single, fluid finger motion: circle-and-tap.

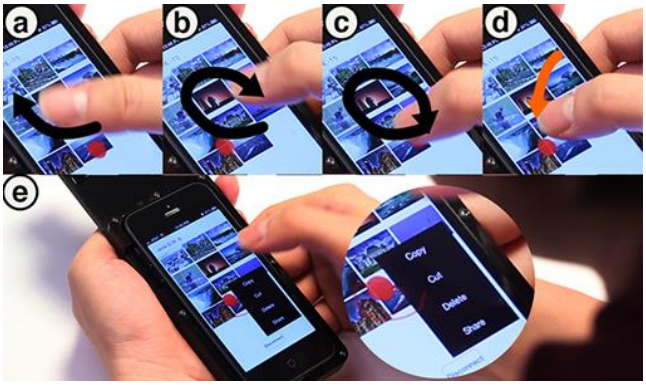

Figure 6. In our photo viewer, a circle-in-air (a,b,c) and tap (d) brings up a context menu (e).

High-up and Tap for Mode Switching

One-handed map navigation is difficult on a mobile handheld device when only the thumb is available for interaction. Our map application demonstrates how Air+Touch allows users to switch between panning and zooming modes simply by raising the thumb 'high-up' before a tap (Figure 7). The person can then scroll on the screen to pan the map (Figure 7ab), or to zoom in/out of it as if using a virtual slider (Figure 7cd).

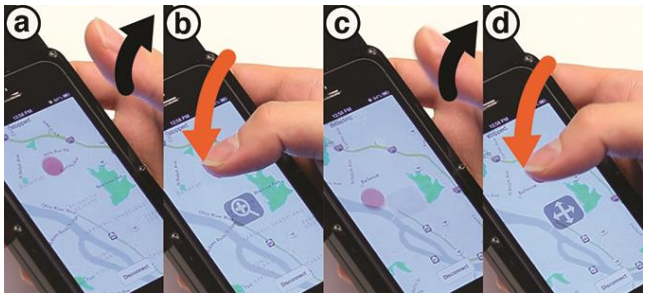

Figure 7. In our map app, raising the finger 'high-up' (a, c) before touch down switches between pan/zoom modes (b, d).

Air Between Touch

Performing an in-air gesture in between consecutive touch events offers the opportunity to parameterize two-point or even multi-point actions.

Finger 'High Jump' Between Touches to Select Text

Because there is no immediate way to disambiguate between scrolling and selection in touch interfaces, routine actions such as copy and paste are unwieldy. Air+Touch can streamline this process with a solution that takes two taps (Figure 8). A user can select a region of text by 1) tapping the beginning of the desired selected region, 2) raising the finger up high, and then 3) touching the end of the selected region. In sequence, these three steps can be executed in a single finger movement. Further touches can provide fine-grained adjustment if needed (d). This creates a gestural shortcut that chunks [4] the specification of text area and the intention to select it into a single finger 'high-jump'.

Drawing an 'L' Between Touches for Marquee-Selection

Similarly, cropping or selecting a sub-region of an image typically requires first interrupting the current interaction and then specifying a special application mode (e.g., through toolbar buttons). However, with Air+Touch, this can be achieved in a more fluid manner, by performing an 'L' gesture in-between two touches. The first and second touches specify the opposite corners of a rectangular marquee. In piloting, we found that drawing an 'L' was a succinct and natural way of expressing the intention of selecting a rectangular area.

Air After Touch

In this category, a person performs in-air gesture as the fingers leave the surface. Air augments touch by mapping touch to a specific function (similar to air before touch) or by allowing touch to continue the interaction unconstrained

Figure 8. In our reader app, a finger 'high jump' (b) between two touches (a,c) defines and selects a region of text. The user then can also use touch to adjust the selection (d).

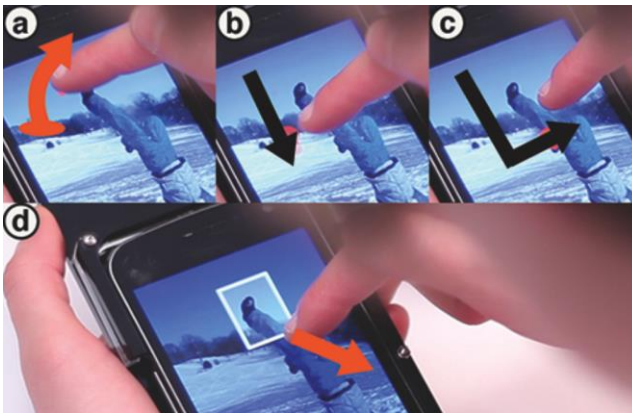

Figure 9. In the drawing app, a rectangular selection can be made by performing a tap (a), followed by drawing an in-air 'L' gesture (b.c), and finally closed by another tap (d).

by screen size, e.g., clutch-free scrolling and zooming.

Drawing a 'Pigtail' After Touch for Free-form Selection

In our drawing application, dragging a finger on the screen is used to draw. However, this path can be parameterized with a post-touch, in-air gesture. For example, by lifting the finger and performing a pigtail motion in the air (Figure 10), the last drawn path is converted into a clipped region that can be e.g., moved, scaled or copied to the clipboard.

Cycling In-Air After Touch to Zoom on a Map

We previously described an air before touch technique that enables quick mode switching between pan and zoom. Another solution is to 'divide the labor' – touch can be used to pan, while in-air cycling zooms.

More specifically, a person starts by tapping on e.g., a map to specify the zoom center (a). As she releases her finger from the screen, a 'zoom mode' may be triggered by drawing a circle high in the air (Figure 11b). Once in 'zoom mode', continuously cycling the finger in the air zooms in or out (depending on cyclical direction) at the tapped location (Figure 11cde). Tapping on the screen, or a short period of non-cyclical finger motion exits the zoom mode. This

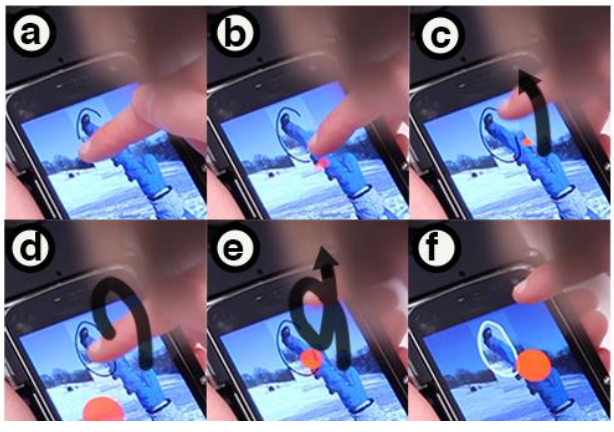

Figure 10. In our drawing app, a user can specify a clipping region by using touch to draw an arbitrary path (a,b), lifting her finger (c), and drawing a pigtail in the air (d,e,f).

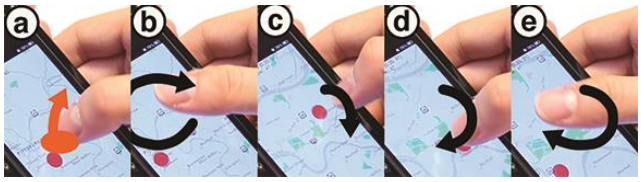

Figure 11. In our map app, a tap (a) followed by a 'circle' high above the screen (b) allows one to continuously zoom at the map by cycling the finger (c,d,e).

technique leverages the concept of a repeated gesture; even if the finger accidentally draws a circle in-air after touch, it will at worst turn on the zooming mode but not cause any actual zooming.

Hovering After Touch to Change Scroll Speed

On a touch screen, clutching is inevitable as touch is constrained by the screen's physical surface. For example, scrolling through a long page requires repetitive finger flicking [20, 21]. Our reader application enables fine control of page scrolling for long lists. When a user triggers inertial scrolling via a flick (a), he can use the hover height of the finger to control the scrolling speed – higher finger position maps to faster scrolling (Figure 12b-d). This is similar to Zliding and Zoofing techniques [20, 21], but uses Z-distance instead of pressure. Touching the screen stops scrolling. Two height thresholds are used to differentiate this 'hover scroll' from a normal scrolling, which is unaffected.

DISCUSSION

The example techniques we have presented above are only a small subset of the possible interactions, yet we believe demonstrate the expressiveness and promise of Air+Touch. Importantly, Air+Touch actions can work in concert with conventional touch gestures, such as one finger pan and click, pinch to zoom, and various chorded swipes. As highlighted by our observational study and implemented in our example applications, Air+Touch techniques can weave in-air gestures before, between, and after touch events. Through extensive use and piloting, it become apparent that these categorization have different strengths and can support a variety of interactive tasks:

- Both *air before* and *after touch* enable quick mode switching connected to a touch down/up (e.g., Figure 6). They can also specify an action specific to a set of touch points (e.g., Figure 10);
- *Air after touch* further allows a user to continue a touch-initiated operation with in-air, continuous motions (e.g., Figure 11);
- *Air between touches* is good for tasks that by nature require specifying multiple screen positions. An air-gesture command can be embedded in between the touch events, saving the overhead of tool or mode switching (e.g., Figure 9).

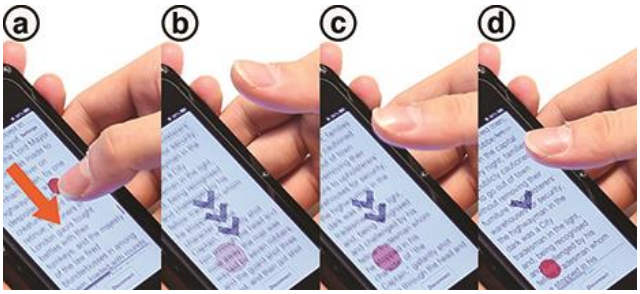

Figure 12. In our reader app, one can use the finger's hover height to control the auto-scrolling speed.

Chunking Air and Touch into Fluid Interactions

Table 1 provides a comparison of how Air+Touch techniques approach the design for six interactive tasks in comparison with existing touch-only interaction. While the elements of these tasks remain the same (e.g., text selection consists of specifying the selection mode and the region to select), a touch-only design presents them as discrete steps. Air+Touch, however, chunks these elements into fluid interactions [4]. For expert users, Air+Touch could become integrated into their interactions as *a single flow* of movement, whereas touch-only actions are inherently sequential.

Choosing Air Gestures based on Accompanying Touch

Our initial concept to trigger a context menu (Figure 6e) was by drawing a 'pigtail' and tapping on an target (similar to [8]'s design). However, we found it difficult to perform, because as the finger drew the 'pigtail', it strayed from its original target, requiring the user to retarget at the end of the gesture. In contrast, a full 'circle' gesture was easier, as the finger could complete a full loop, naturally returning to its starting point, from which the user could simply tap down onto the target. Conversely, when designing after-

	Touch Only	Air + Touch
Open context menu	**Tap** to open image → **tap** menu button (or **tap** and hold)	Circle in air → **tap** on image
Zoom on map	**Pan** to center zoom area → **tap** buttons to zoom in or out	**Tap** on zoom center → cycle finger in air to zoom
Marquee select	**Tap** to bring up toolbar → **tap** 'marquee selection' → **tap** and drag to specify selection	**Tap** start point → draw 'L' in air → **tap** end point
Text selection	**Tap** to specify cursor location → **tap** and hold → **tap** and drag to specify selection	**Tap** starting point → finger high jump → **tap** end point
Free form selection	**Tap** to bring up toolbar → **tap** 'free-form selection' button → **tap** and drag to specify selection	**Tap** and drag to specify selection region → draw pig tail in air
Scrolling	**Tap** and scroll (repeat as needed)	**Tap** to scroll → hover continues to scroll

Table 1. Air+Touch techniques for six interactive tasks (right) in comparison with existing touch-only approach (left). Air+Touch is able chunk steps of interaction into fluid movement of the finger on and above the device's screen.

touch in-air gestures, we found that 'pigtails' became easy to perform, as there was no ending targeting constraint. This suggests that the choice of in-air gesture should consider whether it affects the touch that precedes or follows it.

Segmenting Air Gestures Before and After Touch

For air before and after touch, touches only segment air's start or end points, leaving the developer to decide when to start/stop processing the finger's remaining movement. This translates to the implementation level question of setting the size of the buffer that keeps a history of the finger's 3D positions. In prototyping, we visualized the finger's trajectory as a projection onto the screen. We chose buffer sizes that neither gave an incomplete gesture (too few points), nor overshot it (too many points). An alternate approach would be to analyze different buffer sizes, choosing gestures that yield highest recognition confidences.

CONCLUSION

The prevalence of hover technologies at CES 2014 and the continued inclusion of hover in flagship devices (such as the soon-to-be-released Galaxy S5) suggests in-air technologies will continue to mature and could play an increased role in touch devices. Today, a scant few 'air' gestures are supported and are fundamentally compartmentalized from touch interactions. Our work helps point the way to more powerful interactions, by synergistically interweaving touch and air modalities, where air augments touch, adding expressivity, and touch segments in-air gestures to resolve segmentation ambiguity. With good design, these actions can blend into single, fluid movements, offering a level of expressivity rarely achieved by each modality in isolation. Nonetheless, there is much future work to consider, including expanding the gesture vocabulary, capturing not just 3DOF position, but also 3DOF rotation of the fingers, as well as utilizing several fingers at once.

REFERENCES

1. Banerjee, A., Burstyn, J., Girouard, A., and Vertegaal, R. Pointable: an in-air pointing technique to manipulate out-of-reach targets on tabletops. In *Proc. ITS '11*, 11-20.
2. Baudisch, P. and Chu, G. Back-of-device interaction allows creating very small touch devices. In *Proc. CHI '09*, 1923-1932
3. Butler, A., Izadi, S., and Hodges, S. SideSight: multi-"touch" interaction around small devices. In *Proc. UIST* '08, 201-204.
4. Buxton W. Chunking and phrasing and the design of human-computer dialogues. In *Human-Computer Interaction*. Morgan Kaufmann, San Francisco, CA, USA 494-499.
5. Grossman, T., Hinckley, K., Baudisch, P., Agrawala, M., and Balakrishnan, R. Hover widgets: using the tracking state to extend the capabilities of pen-operated devices. In *Proc. CHI '06*, 861-870.
6. Harrison, C. and Hudson, S.E. Abracadabra: wireless, high-precision, and unpowered finger input for very small mobile devices. In *Proc. UIST '09*, 121-124.
7. Hilliges, O., Izadi, S., Wilson, A.D., Hodges, S., Garcia-Mendoza, A. and Butz, A. Interactions in the air: adding further depth to interactive tabletops. In *Proc. UIST '09*, 139-148.
8. Hinckley, K., Baudisch, P., Ramos, G. and Guimbretiere, F. Design and analysis of delimiters for selection-action pen gesture phrases in Scriboli. In *Proc. CHI '05*, 451-460.
9. Hinckley, K., Chen, X. and Benko, H. Motion and context sensing techniques for pen Computing. In *Proc. GI* '13.
10. Hinckley, K. and Song, H. Sensor synaesthesia: touch in motion, and motion in touch. In *Proc. CHI '11*, 801-810.
11. Hinckley, K., Yatani, K., Pahud, M, Coddington, N., Rodenhouse, J., Wilson, A., Benko, H. and Buxton, B. Pen + touch = new tools. In *Proc. UIST '10*, 27-36.
12. Hudson, S.E., Harrison, C., Harrison, B.L., and LaMarca, A. Whack gestures: inexact and inattentive interaction with mobile devices. In *Proc. TEI '10*, 109-113.
13. Jones, B., Sodhi, R., Forsyth, D., Bailey, B., and Maciocci, G. Around device interaction for multiscale navigation. In *Proc. MobileHCI '12*, 83-92.
14. Ketabdar, H., Yüksel, K.A., and Roshandel, M. MagiTact: Interaction with Mobile Devices Based on Compass (Magnetic) Sensor. In *Proc. IUI '10*, 413-414.
15. Kratz, S., Rohs, M., Guse, D., Müller, J., Bailly, G. and Nischt, M. PalmSpace: continuous around-device gestures vs. multitouch for 3D rotation tasks on mobile devices. In *Proc. AVI '12*, 181-188.
16. Kratz, S. and Rohs, M. HoverFlow: expanding the design space of around-device interaction. In *Proc. MobileHCI '09*, 1-8.
17. Marquardt, N., Jota, R., Greenberg, S. and Jorge, J.A. The continuous interaction space. In *Proc. INTERACT '11*, 461–476.
18. Niikura, T., Hirobe, Y., Cassinelli, A., Watanabe, Y., Komuro, T. and Ishikawa, M. In-air typing interface for mobile devices with vibration feedback. In *Proc. SIGGRAPH* '10, 1–1.
19. PMD Technologies. Camboard Nano camera. http://pmdtec.com/products_services/reference_design.php
20. Quinn, P. and Cockburn, A. Zoofing!: faster list selections with pressure-zoom-flick-scrolling. In *Proc. OZCHI '09*. 185-192.
21. Ramos, G. and Balakrishnan, R. Zliding: fluid zooming and sliding for high precision parameter manipulation. In *Proc. UIST '05*. 143-152.
22. Samsung. Galaxy S4. http://www.samsung.com/galaxys4
23. Synaptics. 3D Touch. http://blog.synaptics.com/?p=401
24. Wigdor, D., Forlines, C., Baudisch, P., Barnwell, J. and Shen, C. Lucid touch: a see-through mobile device. In Proc. UIST '07, 269-278.
25. Wigdor, D., Leigh, D., Forlines, C., Shipman, S., Barnwell, J., Balakrishnan, R. and Shen, C. Under the table interaction. In *Proc. UIST '06*, 259-268.
26. Wobbrock, J.O., Wilson, A.D. and Li, Y. Gestures without libraries, toolkits or training. In *Proc. UIST '07*, 159-168.