TriboTouch: Micro-Patterned Surfaces for Low Latency Touchscreens

Craig Shultz
Carnegie Mellon University
Pittsburgh, PA, USA
cshultz@andrew.cmu.edu

Daehwa Kim
Carnegie Mellon University
Pittsburgh, PA, USA
daehwak@andrew.cmu.edu

Karan Ahuja
Carnegie Mellon University
Pittsburgh, PA, USA
kahuja@cs.cmu.edu

Chris Harrison
Carnegie Mellon University
Pittsburgh, PA, USA
chris.harrison@cs.cmu.edu

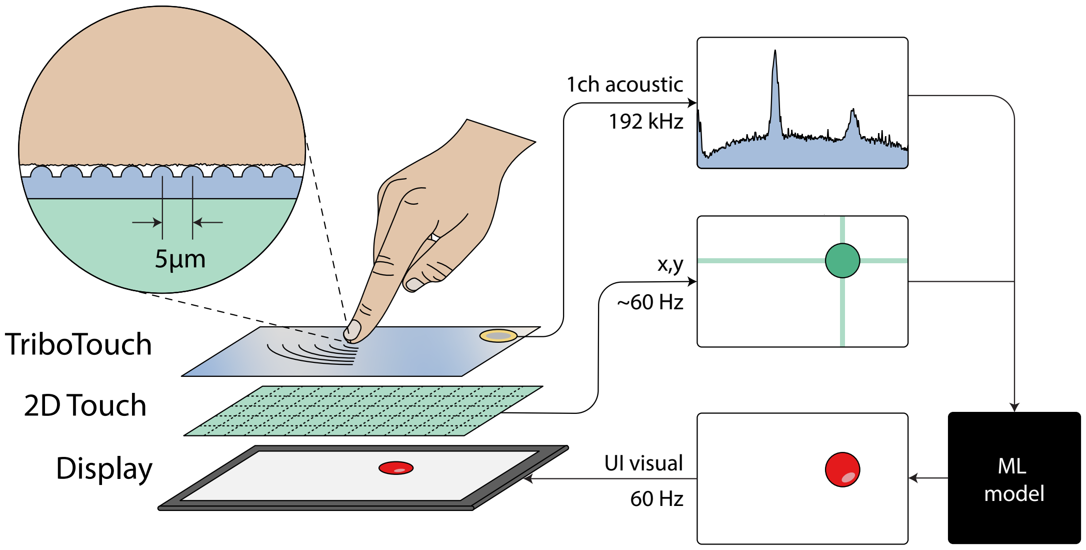


Figure 1: Block diagram of the TriboTouch system. The TriboTouch layer sits on top of a 2D touch sensor and display. Micro-patterns with a pitch of 5μm induce vibrations in the screen when a finger is dragged. This signal is sampled and fused with conventional 2D touch data, then fed into a machine learning model, which makes touch predictions with a lower latency than the touchscreen data alone.

ABSTRACT

Touchscreen tracking latency, often 80ms or more, creates a rubber-banding effect in everyday direct manipulation tasks such as dragging, scrolling, and drawing. This has been shown to decrease system preference, user performance, and overall realism of these interfaces. In this research, we demonstrate how the addition of a thin, 2D micro-patterned surface with 5 micron spaced features can be used to reduce motor-visual touchscreen latency. When a finger, stylus, or tangible is translated across this textured surface frictional forces induce acoustic vibrations which naturally encode sliding velocity. This acoustic signal is sampled at 192kHz using a conventional audio interface pipeline with an average latency of 28ms. When fused with conventional low-speed, but high-spatial-accuracy 2D touch position data, our machine learning model can make accurate predictions of real time touch location.

CCS CONCEPTS

• **Human-centered computing** → **Touch screens**; *Interaction devices.*

KEYWORDS

Input Techniques, Touchscreens, Sensors, Latency, Tribology

ACM Reference Format:
Craig Shultz, Daehwa Kim, Karan Ahuja, and Chris Harrison. 2022. TriboTouch: Micro-Patterned Surfaces for Low Latency Touchscreens. In *CHI Conference on Human Factors in Computing Systems (CHI '22), April 29-May 5, 2022, New Orleans, LA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3491102.3502069

1 INTRODUCTION

Touchscreen technologies have made tremendous progress over the past half century [5]. Metrics such as spatial accuracy, maximum screen size, scratch resistance, water rejection, optical clarity, and number of touch points tracked have steadily improved, while costs have continued to fall. For this reason, touchscreens can be found in domains as diverse as kitchen appliances to airplane cockpits, and of course the ubiquitous smartphone. However, there is one metric that has been notoriously stubborn to improve: touch latency. Specifically, the motion-to-photon delay between what a finger does and how fast coordinated graphics are drawn. Most touchscreen systems have improved latency below the 100ms threshold for keyboards and mice that has been the industry standard for decades [39], yet even modern generation smartphones have dragging latencies on the order of 80 ms.

This delay causes a rubber-banding effect that impacts precision in tasks such as drawing, and, in general, breaks the realism and responsiveness of direct manipulation interfaces [10, 23, 38]. Thus, techniques to reduce touchscreen latency are an active area of research [19, 31, 32, 36]. Many hardware and software approaches have been proposed with varying degrees of success. Software extrapolation is inherently limited by old and ambiguous positional data, and closing the gap to zero perceptible latency with acceptable jitter may be impossible [31]. Hardware approaches appear more promising, but it is not trivial to simply increase system scan rate due to factors such as sensor noise, timing constraints, data overhead/computation, and power consumption.

In this paper, we present a new hardware+software approach to reducing touchscreen latency. Specifically, we apply a film embossed with a 2D pattern of small bumps with a pitch of 5 microns (Figure 1). When a finger, stylus, or tangible is translated across this patterned surface, interaction with the micro-features induces an acoustic vibration with a fundamental frequency that directly encodes x and y velocities (Figure 3). This vibration is low amplitude and, at many velocities, ultrasonic, meaning it is imperceptible to the user. We sample this signal using a conventional audio pipeline at 192kHz with a mean latency of 28 ms – less than half the time of most touch pipelines. We fuse this high-bandwidth, low-latency, 1D vibroacoustic signal with low-framerate but high-spatial-accuracy 2D touch data (Figure 1) reported by a conventional projected capacitance touchscreen (latency of 69ms). We note for the reader that our technique could also work on resistive, self capacitive, optical, acoustic, and a plethora of other touchscreen technologies, but we focus on projected capacitance because of its dominance in the touchscreen market. By fusing this multimodal data, our machine learning model can make a more accurate prediction of touch location than using touch alone, reducing latency from 96ms to 16ms with mean distance error of 5.13mm.

2 RELATED WORK

Reducing visual touchscreen latency can be complicated, as there are many interrelated aspects of hardware and software which determine overall latency. Because of this, many approaches in both hardware and software have been explored. Additionally, touchscreen latency affects users differently depending on the task (such as tapping or dragging), and interface feedback type (direct or indirect). We review aspects of latency perception, and approaches to reducing overall system latency, highlighting key methods and seminal work. We also include an overview of work in the HCI literature that uses physical patterning for touch-related input.

2.1 Latency Impact on User Experience

For decades, the general rule of thumb for graphical user interface latency was to keep it under 100ms [39]. Traditional keyboard and mouse GUIs, however, are indirect interfaces, and latency issues became more pronounced with the advent of the direct manipulation touchscreen. Kaaresoja and Brewster were among the first to systematically measure touchscreen visual latencies, finding latencies on smartphones ranging from 60-210ms [24]. To answer the question of if this level was acceptable to users, Anderson et al. studied the effect of latency in tablet computing and found that most users would accept latencies under 580ms, however the authors noted that their users were quite inexperienced with the technology, and were possibly overly positive with their judgements [2]. Ritter et al. repeated this type of study with tapping and dragging tasks, and found "acceptable" latencies of 300ms and 150ms respectively [46]. Regardless, users continued to show preference for lower latencies in both studies, even down to the limits which were used (80ms and 100ms respectively).

Ng et al. also measured common touchscreen latencies between 50-200ms, and examined latency perception psychophysically with a system capable of 1ms latency. They found that not only could users perceive under 10ms of latency (average latency JND of 2-10ms), but that experiencing such low latency "broke" their perception of latency, leaving them unsatisfied with higher latency devices [38]. Later work showed that users' performance is also affected by latency, though performance gains do not continue much beyond 25ms net latency for dragging tasks, or 50ms for tapping tasks. The authors recommended 20ms as a good tradeoff between performance, perception, and typical tasks [23]. This task dependency was investigated further by Deber et al., who found that latency was dramatically more noticeable in direct dragging tasks (JND of 11ms) as opposed to the indirect dragging or direct and indirect tapping (JNDs of 55ms, 69ms, and 96ms respectively) [10]. For context, this implies that latency improvements of a single display rendering frame (16.7ms on a 60Hz display) are noticeable down to under 1 frame of delay. Stylus interactions show similar trends to touch dragging [23, 37], though they are not entirely the same, reinforcing that motor-visual latency perception is a complex, task dependent problem.

Despite the fact that users can perceive very low latencies, and that these perceptions can translate to preferences and performance, touch latencies on commercial devices have seemingly not improved in the last decade. We ran our own small study using recent smartphone hardware. We captured 960 FPS footage of icons being dragged across the homescreen (i.e., a native app optimized by manufacturers) and calculated the motion-to-photon latency. For an Apple iPhone 12 (released October 2020) we measured a mean latency of 78ms. For a Google Pixel 4A (released August 2020), we measured a latency of 90 ms. For a Samsung S10e (released March 2019) we measured a mean latency of 77ms. While users may accept this level of latency in their devices, the debate still continues on if this is really low enough, and there are calls for updated HCI latency guidelines [3]. We take the approach that perceivable latency is too much latency, and believe that moving to imperceptible latency will improve the user experience in much the same way as the transition to visual pixel sizes below the perceptible limit (e.g. "retina" displays) led to a step change in display fidelity (i.e. once users experience it, it's hard to go back). To this end, we set a benchmark target of reaching approximately 16ms of round trip drag latency, or one frame update of a 60Hz screen, which is right on the edge of perceptible latency for everyday dragging tasks.

2.2 Hardware Centric Approaches

There is an extremely broad assortment of touch tracking hardware technologies, by one tally 13 categories with 38 variations [48]. Some notable methods include: passive acoustic time-of-flight [22, 41, 56], electric field sensing [57, 58], LIDAR [30, 40], and cameras of many varieties, including visible light [25, 29, 47], infrared [15, 51, 52], and depth [16, 54, 55]. However, capacitive [12], and especially projected capacitive (pcap) [32, 45, 49] touchscreens have come to dominate the market due to their robust performance, thin construction, and multitouch capability [48].

Specific hardware attempts to reduce latency either redesign the sensing and display platform [23, 32, 38] or add additional hardware to improve performance [14, 31, 53]. Redesigned systems can achieve motion-to-photon latencies of 1ms [10, 23, 38] or, in special cases of a FDM pcap device, 150μs [32]. However, these hardware prototypes use custom hardware, firmware, and software stacks, and are primarily intended to test the limits of latency perception. Add-on hardware systems have included finger-worn IMUs, [14, 31], as well as external high speed cameras viewing the ballistic trajectory of finger inputs [53]. While there is work using triboelectrification (electric charge generation from friction) for touch sensing [27], there have been no reports in the literature of using friction induced vibration for touch tracking.

2.3 Software Centric Approaches

Software based approaches to reducing latency are more limited in their scope. Since they must deal with inherently slow hardware, they rely on tracking finger trajectories and making forward predictions (sometimes nearly 100ms in the future). Cattan et al. studied software corrected latencies with a simple linear predictor model, making predictions between 25ms and 75ms into the future [7, 8]. They found that errors dramatically increased with forward prediction time, especially overshoot, which negatively impacted user's preference and performance. Importantly, they highlight that software prediction is unable to account for trajectories of high acceleration, such as when the finger rapidly stops, as there is no way of accurately predicting from an absence of data.

Henze et al. applied neural networks to the prediction problem, trained with actual touch strokes [19]. They found that polynomial fit predictions (1st, 2nd, or 3rd order) generally performed worse than no prediction except for small latencies, but a neural network approach was able to reduce error (by roughly 50%) and increase user performance. This came, however, with a negative consequence of increased prediction jitter, which was notable and led to unpleasantness. Additional machine learning algorithms were subsequently evaluated [20], and it was found a LSTM approach outperformed neural networks, or multilayer perceptrons in terms of raw error. Work by Nancel et al. recently showcased a finite-time approach which was found to be effective at predicting 32-48ms ahead [36], 2-3 times longer than many published industry and academic techniques.

We think these software approaches are promising, especially for short term prediction (under 30ms), and it is something that smartphone manufacturers appear to routinely apply [36]. However, software only implementations always show their limitations at high velocities and large latency prediction values, and do not seem capable of closing the gap on many systems which have total latencies of 60-100ms. The only way around this is to either increase the capabilities of existing hardware, or to add additional hardware. As mentioned previously, speeding up the hardware is not always a possibility, especially true as display size increases. Adding new hardware, such as high speed sampling of IMUs, seems promising at reducing error in long term prediction [31], but requiring non-integrated hardware is a large burden on the user. It is with this in mind that we developed the TriboTouch technique, a method which reduces touch tracking latency with minimal hardware and software impact to the system or user.

2.4 Uses of Patterning for Touch Interaction

Before describing TriboTouch in detail, there are a handful of projects to mention that have utilized coarse geometry of surfaces to enable touch input. For example, Scratch Input [17] used a contact microphone in concert with unstructured textured surfaces (e.g., wooden tables, painted walls) to enable touch gesture classification, such as swipes and circles. Rubbinput [26] is similar, but uses the squeaking sounds produced by fingers on wet surfaces. By decomposing and analyzing acoustic time-difference-of-arrival signals, Pham et al. [43, 44] was able to demonstrate continuous touch tracking.

Engineered textures are also possible, as exemplified by the Stane [35] - a palm-sized, 3D-printed device with an internal microphone covered in a multitude of engineered textures; the device classifies touch input using sound produced by rubbing different regions. Acoustic Barcodes [18] is another example of engineered surface geometry. Much like their optical counterparts, a linear sequence of irregularly-spaced lines are etched into a material, which produces an acoustic pattern when swiped with a finger that can be resolved to a binary data payload. Kim et al. proposed a millimeter scale textured surface and wrist mounted piezo as a gestural input device [28]. To our knowledge, no work to date has investigated using micron-scale patterning for touch input broadly, nor latency improvements specifically.

3 TRIBOTOUCH PRINCIPLE

TriboTouch works on a principle of texture induced vibrations from frictional interaction. When two surfaces rub together vibrations are created which are a function of each surface's spatial roughness distribution and the velocity at which they are sliding. This idea is intuitively understood by considering a zipper: The faster the zipper is opened and closed, the higher the frequency of sound

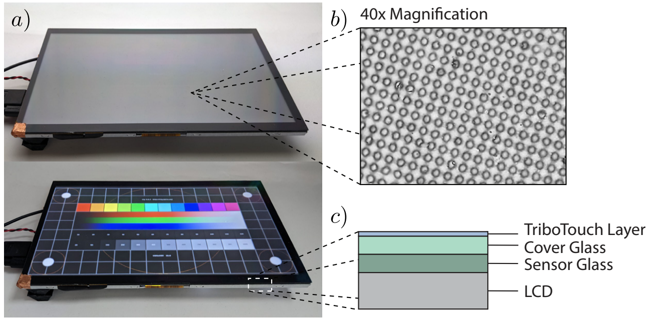


Figure 2: a) The patterned layer and piezo is fitted to an LCD touchscreen; screen off (top) and with a test pattern shown (bottom). b) Magnified image of the diffraction pattern (reproduced from [9] with permission from the author). c) a schematic device stackup showing the ordering of functional layers.

that is generated. Instead of a zipper, however, our system uses a finger or other object sliding against a micro-patterned surface. Friction is a well known cause of a number of acoustic phenomena [1], and in recent years the systematic variation of friction-induced vibration of fingers against textured surfaces has been investigated as a basis for human perception of finely textured surfaces [34, 50]. These vibrations can travel along the skin and into the body [11], however, as we have found in our work, they readily propagate into the surface as well. Additionally, it has become clear that texture-elicited vibrations can carry rich information about finger sliding speed [13]. It was with this insight in mind that we developed TriboTouch (Tribo being short for tribology, the study of friction).

3.1 Using the Right Pattern

In order to feel a texture, the spatial features of the surface need to be in the range of 100s of microns. This is because a 100um spaced texture scanned at a nominal velocity of 10mm/s will produce vibrations of 100Hz, which is in the range of tactile vibration perception. For the purpose of sensing, however, we do not want the users to be able to feel the textured pattern. Fortunately, we discovered that the relationship between wavelength and frequency is not limited only to widely spaced textures. By selecting surfaces with much finer features, vibrations are produced at much higher frequencies. This has the benefit of reducing the tactile perception of the texture while sliding, and moving the frequency into a region of the spectra that is virtually free of acoustic noise. We have found that the general relationship $f = v/\lambda$, where f is the vibration fundamental frequency, v is the sliding velocity, and λ is the dominant spatial wavelength, holds true even for wavelengths of down to $5\mu m$ and frequencies in excess of 80 kHz. To study this effect, we obtained holographic diffraction grating films, inexpensive films made of UV cured epoxy on PET substrate, and laminated them to a touchscreen. These films come in single or double axis variations, and a microscope image of the double axis variety can be seen in Figure 2b. We selected these films because they are pre-made and inexpensive, but nothing precludes the use of patterns of different spatial wavelengths or orientations.

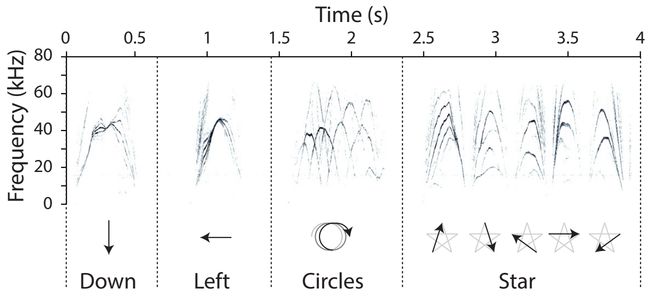


Figure 3: Example spectrogram plots of acoustic data coming from a finger performing various gestures (down, left, circles, and star) on our TriboTouch-enabled prototype system. Spectral peaks encode velocity information.

3.2 Spectral Complexity

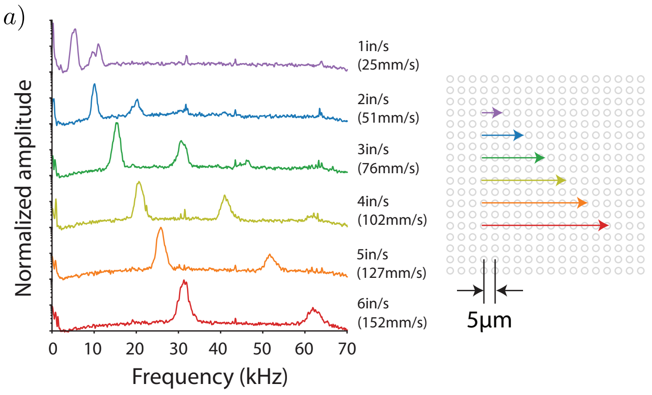
When surface vibrations are recorded with a piezoelectric disc placed in the corner of the surface, complex and high frequency vibrations are observed, see Figure 3. If the finger travels predominantly along the x or y axis of the diffraction grating, vibrations are limited to a fundamental frequency and series of harmonic vibrations. This can be seen in the vertical and horizontal finger motions (Figure 3, down and left). As a swipe begins, the frequency rises sharply, since the finger is accelerating. The frequency then reaches a peak value, and begins to fall as the finger decelerates and finally stops. If a finger follows a path with curvature, fundamental frequencies associated with both x and y axes interchange with one another (Figure 3, circles), and intermodulation frequencies (sum and difference frequencies) can be observed. Swipes at off axis angles also show complex patterns of intermodulation, as seen by the star gesture (Figure 3, star).

3.3 Systematic Variation

We performed two experiments using a CNC platform to explore and demonstrate the effects of velocity magnitude and angle on the spectra of observed vibrations. A teflon coated stylus was attached to the head of the CNC platform, and held against the surface by a compressed spring. A picture of this setup can be seen in Figure 4b.

3.3.1 Magnitude Variation. For our magnitude experiment, we ran the stylus inline with the horizontal x axis at 6 different velocities, ranging from 1in/s (25mm/s) to 6in/s (152mm/s). Constant velocity segments of 0.5s length were broken into non-overlapping windows of 2048 samples. These windows were averaged to produce spectra seen in Figure 4a. As can be seen in the data, changing the velocity magnitude has the main effect of linearly translating spectra along the frequency axis. Both the fundamental and the second harmonic appear in the data, and scale linearly with sliding velocity (e.g., a fundamental peak of 5kHz at 1in/s equates to a 20kHz peak at 4in/s).

3.3.2 Angle Variation. For our angle experiment, we adjusted the path of the stylus to coincide with 5 different angles, from 0° (inline with the x axis) to 45°. Data was collected at constant velocity of 5in/s, and 0.75s of constant velocity data was extracted, windowed, and averaged to produce the spectra in Figure 5a. Here the impact



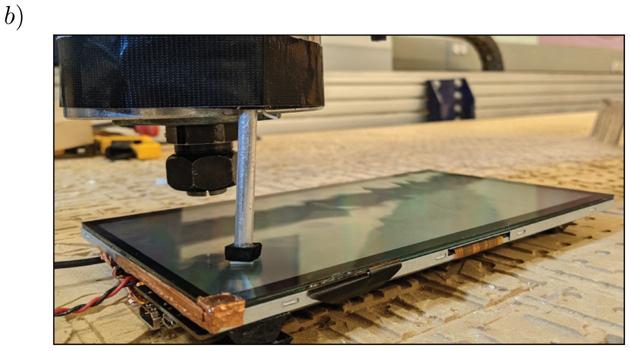


Figure 4: Spectra amplitude and frequency resulting from a stylus dragging across the surface at different controlled velocities. The inset figure shows example movement alignment with the micro-pattern (not to scale). b) Setup image for magnitude and angle experiments, showing stylus and display orientation to CNC head.

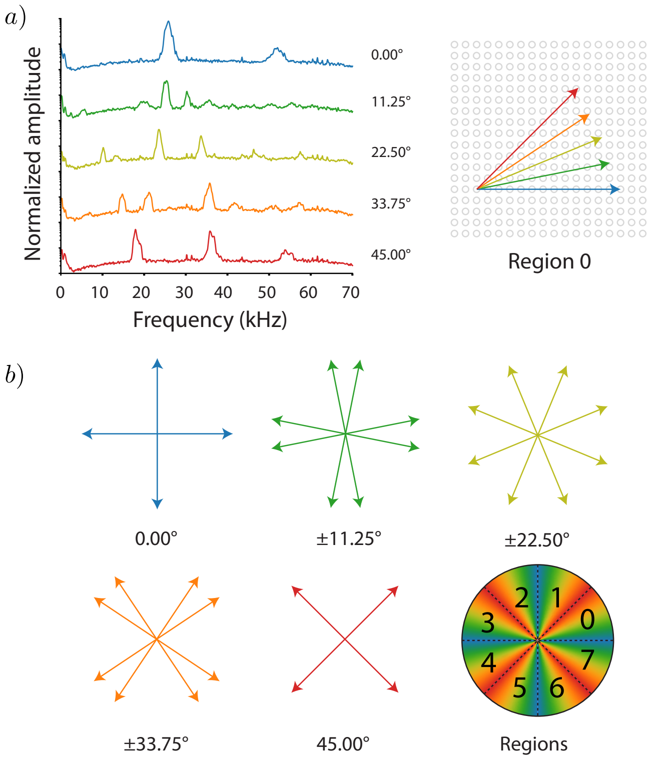


Figure 5: a) Spectra amplitude and frequency resulting from a stylus dragging across the surface at different controlled angles (5in/s velocity). The inset figure shows example movement alignment with the micro-pattern (not to scale). b) Color coded angular "homophones", i.e., different angles that will produce identical spectra, and a region plot showing the symmetry due to the micro-pattern.

of angle on spectra peak is less obvious, but we present a theory on what is happening. Essentially, there are two fundamental frequencies that are produced by the stylus (and harmonics associated with each). Each fundamental peak is associated with the x or y axis, and how in-line the motion is with these axes. Therefore, as the angle increases from 0° to 45°, the x axis fundamental decreases, while the y axis fundamental increases. This can be seen in the 11.25° and 22.5° spectra, as the 26kHz peak decreases slightly, while another peak begins to rise from 0 Hz. This trend continues until the angle reaches 45°, at which point the x axis fundamental and y axis fundamental are equal (18kHz). This is $1/\sqrt{2}$ times the original fundamental at 0°, which is the reduced linear velocity of each axis. Beyond 45°, there is a symmetry in the spectra, as the y axis fundamental continues to increase and the x axis fundamental continues to decrease. This description explains the rising and falling frequencies seen in the circular motion of Figure 3. The remaining peaks in the spectra can correspond to harmonics of the fundamentals, or intermodulation frequencies (essentially sum and difference frequencies of the fundamentals).

This angular behavior is directly due to the symmetry of the diffraction pattern texture across the x, y, and the 45° offset axes. This leads to an interesting phenomenon where there are angular homophones, that is, angles that produce vibration spectra identical to other angles. This concept is illustrated in Figure 5b, where each set of lines will, in theory, produce identical spectra. These angular homophones can be separated into 8 separate regions, where the symmetry of each region is reflected across each border, as illustrated by the regions graph in the lower right of Figure 5b. While the separation of velocity magnitude and angle appear straightforwardly discernible here, we stress that this was with CNC controlled velocities and perfectly linear trajectories, conditions not seen with real user input. For this reason, we took a machine learning approach to identify the most useful features for mapping sound spectra onto x and y predicted touch positions.

4 PROTOTYPE IMPLEMENTATION

The ethos behind our system design was to add new hardware without adding a large amount of engineering complexity. We chose to use materials that are commonly available, and processing pipelines that are well established, robust, and widely implemented. Importantly, software pipelines for sound are commonly designed for low latency and high bandwidth, which we use to our advantage. The hardware, while novel, is compatible with common touchscreen constructions and integrations, and achieves a high level of refinement, even in this first prototype demonstration. We detail the hardware implementation below.

4.1 Hardware

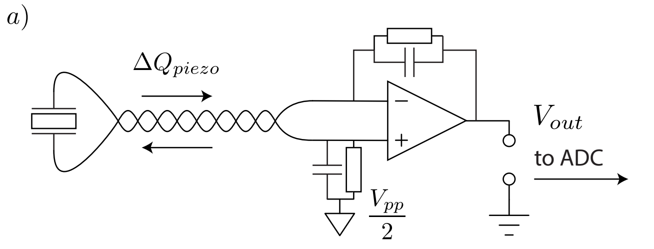
The hardware is centered around an open frame 10.1" touchscreen, with a touch controller ASIC by GOODIX. The display is an IPS LCD, with a native resolution of 1024x600 and viewing area of 223x125mm (1px ≈0.21mmx0.21mm). Video is transferred via HDMI, while touch and power are over USB. Photos of this display can be seen in Figures 2, 4 and 8. The visual screen has a refresh rate of 60 Hz, and the touch has a report rate of 100 Hz. The touch panel is a traditional "out-cell" pcap design, with TX and RX lines on either side of a sensor glass, which is then optically bonded to the LCD on one side and a front cover glass on the other (see Figure 2c for a complete stackup).

A 6"x5' roll of the diffraction grating was purchased online and cut down to rough size before being laminated to the cover glass and trimmed. Liquid optically clear adhesive (LOCA) was used for bonding. It was leveled with a roller by hand and cured in place with a UV light. This is a common technique used in bonding touch sensors to phones and tablets. The pattern spacing of $5\mu m$ was measured by shining a 532nm laser through the diffraction grating, and measuring a separation distance of 210mm between diffraction nodes at a distance of 2m.

A corner of the film was removed for the application of a 10mm diameter, 0.13mm thick, brass backed piezo disc (p/n AB1070B). Before bonding the piezo to the screen, layers of copper and polyimide tape were wrapped around it for shielding. This was found to substantially reduce noise from the touch sensor scanning. The piezo was secured by cyanoacrylate glue ("superglue") and the wires from the piezo were twisted and shielded until they reached the piezo preamplifier, seen in Figure 6.

A custom preamplifier was built for the piezo sensor (Figure 6b) as the vibrations induced by friction are quite low amplitude, and the piezo is in a high noise environment with interference coming from both the touch sensor and LCD module. Nonetheless, with basic noise mitigation techniques, low noise differential charge signals could be read from the piezo with minimal interference. The function of the preamplifier is to 1) buffer the high impedance of the piezo, 2) read the piezo differentially, and convey it to the audio ADC with a modest amount of gain (~100-500x), and 3) integrate the piezo charge and convert it into a voltage representing the force applied to the piezo.

A schematic representation of a single channel of the preamplifier is shown in Figure 6a. It operates from a single 4.5V supply (locally regulated from a 5V USB power source), and uses a single package of two high bandwidth (GBP 22 MHz), low noise (2.9nV/rtHz) op amps (p/n LMP7732) per channel, one for buffering/integrating and another for additional gain and high pass filtering. Overall gain bandwidth was found to extend beyond 100kHz, and signals are conveyed single-endedly to the audio interface via a shielded audio cable. There are two channels per board for additional experiments in the future. The audio interface used is the ZOOM UAC-2 USB 3.0 device. It has a 24-bit ΔΣ ADC, and samples stereo data at 192kHz.



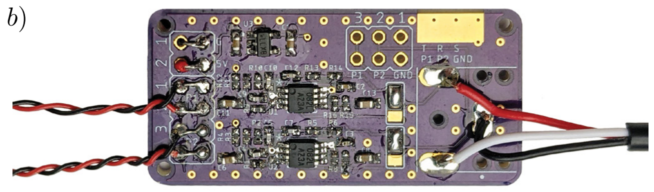


Figure 6: a) Dual channel preamplifier PCBA containing four op amps, a voltage regulator, and various passives. b) Schematic of the main piezo front end charge amplifier. It converts a differential charge signal into a single ended voltage.

As a test computer, we use an Intel NUC (2.6 GHz Core i7 6700HQ, 16GB RAM) running Windows 10.

Note, the 192kHz sampling rate of the ADC sets a theoretical maximum velocity that this system can capture. Using the fact that $v_{max} = \lambda f_{max}$, $\lambda = 5\mu m$, and $f_{max} = 96$kHz (per Nyquist), the highest velocity than can be sensed is approximately 480 mm/s. In practice, it's closer to 400 mm/s, as seen later in Section 7.

4.2 Software

Data collection for the user study and experiments was centralized in Python. Data was captured from the audio interface using the PyAudio library, Python bindings for the cross platform C library PortAudio. The PortAudio library provides an audio callback function which gets called every 256 samples, or approximately 1.33ms. Data is logged at this rate, and the system cursor location is polled to check for touchscreen updates. Touch is reported to the system via the native USB HID driver, and it updates at 100 Hz. Audio is delivered from USB to the PortAudio library via the ASIO4ALL 3rd party driver, which was adjusted to have a minimal buffer size and low latency. Various other Python libraries are used for UI and data analysis tasks, and it was ensured that they did not interfere with the audio callback timing. Our machine-learning-powered touch prediction implementation is described after a discussion on latency characterization.

4.3 Latency Characterization

We characterized the various aspects of latency in our test system through carefully constructed experiments, and report the results here (see also Figure 7). There are three critical latency numbers we investigated: motion-to-reported audio, motion-to-reported touch, and commanded-to-completed display rendering. All report latencies are for dragging, not tapping (tapping latencies are generally

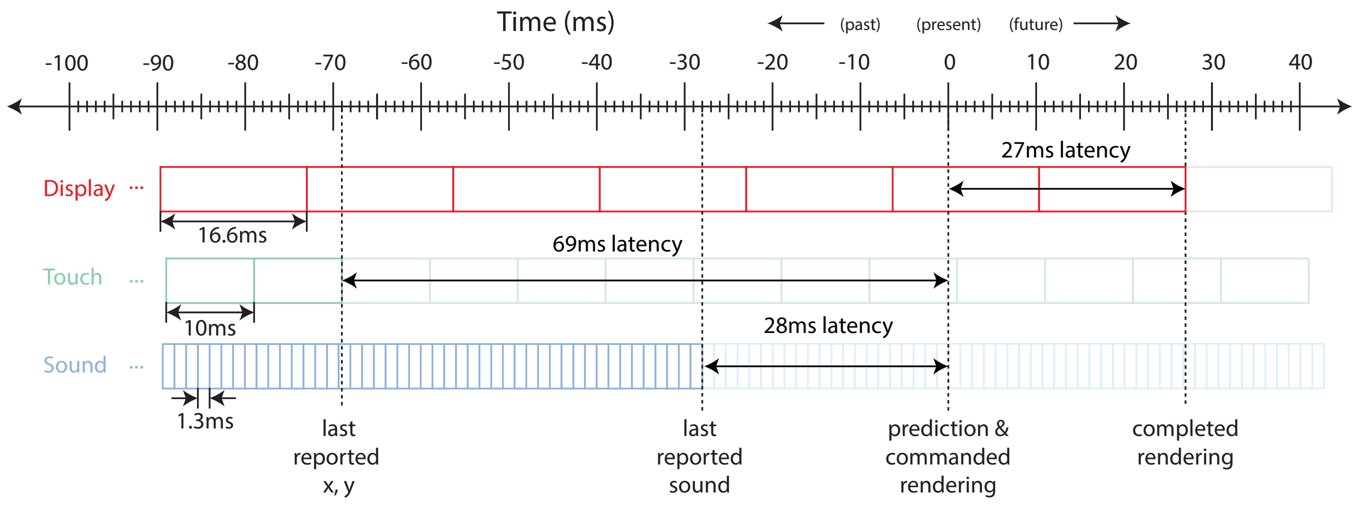


Figure 7: Latency characterization of our system, with results from a latency experiment. Display, touch, and audio update rates are shown as the time between new sampled data. Predictions are run at time 0, i.e. present time, and utilize delayed data from the past to make predictions of the future.

higher). A finger was placed on the screen, held motionless for a brief moment, and then quickly swiped. The piezo signal was used to capture the start of finger motion, as acoustic vibration is immediately produced even at very low velocities. A phototransistor was used to record screen brightness as a proxy for rendering output. We also used the sound input and output pipelines to record critical events and line them up with the piezo and phototransistor outputs. Data was recorded simultaneously on a 100 MHz digital oscilloscope.

There is negligible latency of the vibration traversing the screen or from the piezo preamplifier, so the audio input latency essentially is the motion-to-reported audio latency. This was found by looping the audio input in Python back to the audio output, and then sampling both with the oscilloscope. This gives the round trip latency, which we then divided by two to get the input audio latency. Mean motion-to-audio latency was 28.2ms (SD=1.36ms, N=10). Motion-to-touch latency is measured by pulsing the audio output as soon as a new touch location is reported. Because we characterized the audio output latency (equal to the audio input latency), we can simply subtract this from the touch pulse time and get the motion-to-touch latency. Motion-to-touch latency was 69.1ms (SD=4.30ms, N=10). Finally, we measured the commanded-to-completed rendering latency by looking at the piezo and phototransistor outputs. The phototransistor shows the start and end of a single frame render, and we define the completed display rendering time as the time when the commanded output frame is fully rendered. The display was commanded to render a single black background frame when a new touch is detected. The commanded-to-completed display render time is then the total piezo to phototransistor latency minus the motion to touch latency, a mean of 27.3ms (SD=5.25ms, N=10). These values are summarized in Figure 7.

4.4 Machine Learning

As previously noted, our system receives a new buffer of 256 sound samples every 1.3ms (i.e., 750 Hz) via an audio callback handler. While it is possible to run predictions at this speed, it was too taxing for our older-generation Intel NUC and Python software stack (we discuss possible performance improvements later). Instead, we make predictions every other audio callback, or roughly every 2.7ms (i.e., 375 Hz, vs. the touchscreen's 100 Hz touch reporting rate).

Every time a prediction is triggered, we compute the magnitude spectra of the DFT (using the FFT algorithm) for the most recent sound buffer. This represents frequencies from 0 to 96 kHz in 750 Hz-wide bins. We discard phase information. The bin indices of the top two peaks (using SciPy argrelextrema) are extracted as features. We also iterate over the spectra and record the index of the two highest magnitude bins. We compute the mean, standard deviation, and center of the mass of the spectra as three additional high-level descriptive features. Our software also maintains a sliding window of the last five x, y touch locations reported by the touchscreen, as well as a counter for the number of audio callbacks since the last touch event, which we use as another feature. As described in Section 3 and illustrated in Figure 5, our micro-patterned surfaces have angular homophones, where different touch trajectories can result in the same sound signal. To account for this and help transform our features into a lower-dimensional representation (for easier learning), we convert the last three touch vectors into a homophone-normalized coordinate system (x, y components transformed into region zero; see Figure 5). In addition to the region-zero-local x, y components, we also record the touch vector's original region number (0-7) as features. We perform the same conversion on the summed x, y vector of the last 5 touch points, producing another three features. Finally, we compute the max x velocity and max y velocity over the last 5 touch points.

Our 34 features (normalized by a min-max scalar) are concatenated into a one-dimension input vector and fed to an extra-trees

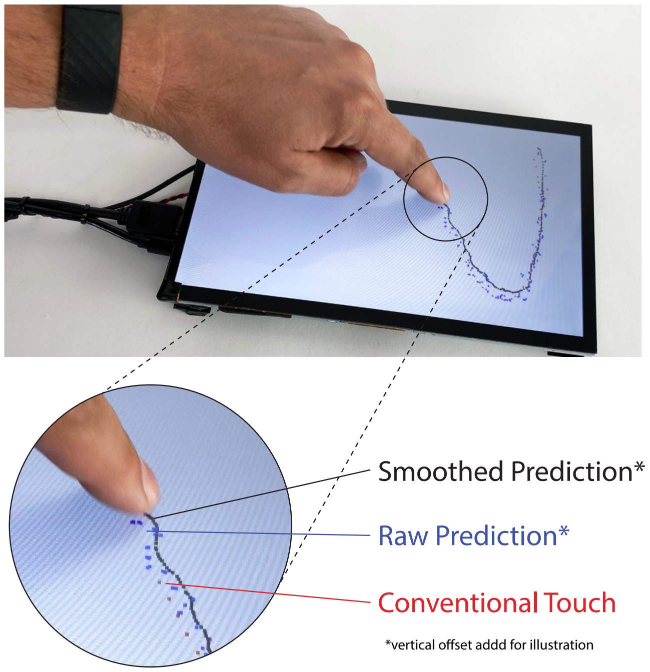


Figure 8: Photo from our real time latency compensation model. Conventional touch data is shown in red, while raw outputs of the acoustic+touch ML model run at 375Hz are shown in blue, and a smoothed output is shown in black. A vertical offset of 20px was applied in y to avoid data overlapping for illustration purposes.

regressor (Sklearn Ensemble ExtraTrees [42]), which takes 1.6ms to execute on average. The output is a two-dimensional vector containing the x, y displacement of the nth touch point (real time latency compensation is 9 touch points ahead), which is added to the last known touch point. At 375 touch predictions per second, our system is over 6x faster than the refresh rate of the visual display, which means we can employ smoothing with minimal cost in latency (not possible with conventional touchscreen events alone). For this, we use the 1€ Filter [6] (used in prior touch prediction work [36]) with β=0.008 and f_c=4 Hz. Figure 8 shows three touch paths (offset vertically for illustration). The red dots are the touchscreen's reported positions at 100 Hz. Shown in blue are our system's raw predictions at 375 Hz, as well as their smoothed versions in black.

To train our extra trees model, we collected nine minutes of touch input data from four users who were not participants in our later study. Users were asked to provide strokes of varying shapes, paths and speeds, but the task was otherwise open-ended to capture a variety of input strokes. In total, this provided 1,137,384 sound points (with 143,450 unique touch points). For model training, we used the mean squared error criterion and 15 estimators. We tested several other models such as nearest neighbor regressors, multilayer perceptrons, and support-vector machines and found that decision trees work the best. While our random forest model achieved similar performance on the same feature set, we ultimately chose extra-trees due to its fast execution time.

5 USER STUDY

To evaluate the performance of our technique, we recruited 11 participants (mean age 23.3, one left-handed) for a 30 minute study (each paid $20 USD). Users were seated in front of our test apparatus and completed a series of touch input trials. We modeled our tasks on [19], which used three input categories: drawing, writing, and (Fitts law style) dragging. Each collection phase lasted approximately 10 minutes, with presentation order randomized. Users were free to choose whichever finger they found most natural for input, though they were instructed verbally not to perform overly rapid swipes. This was to avoid velocities above 400mm/s, the practical maximum velocity limit for our prototype sound pipeline.

In the drawing task, users were shown a piece of paper with six exemplar shapes: square, circle, triangle, five-pointed star, house and zig-zag. The shapes were adapted from [36]. Note that these stimuli do not appear on the screen (which remained white), as we did not want users to trace shapes and have uniform inputs. To add further variability, shapes were requested in one of three screen sections (left, center, right) and at two sizes (small and large) using prompts at the top of the screen. Each shape, section, and size combination was repeated three times, and trial presentation order was fully randomized. This procedure generated a total of 108 shape trials (6 shapes x 3 section x 2 sizes x 3 repeats = 108 trials) per participant.

In the writing task, participants were shown a single word at the top-right of the screen, which they then drew onto the display as they saw fit. No visual feedback was given to avoid user adaptation. Random words with more than four letters were drawn from the [33] dataset. Each participant completed 50 writing trials.

Finally, in our Fitts-law-style dragging task, users were shown two 1 cm squares on screen. They were asked to drag their finger from the blue square to the black square. No visual feedback of the path was shown, but the black square turned blue when the finger was on top. While it might seem natural to randomly generate start and end targets, this would almost certainly generate a diagonal line (i.e., the probability of generating the same x's or y's for the two targets is very low). However, in typical smartphone use, actions such as vertical scrolling or horizontal swiping are more common than a diagonal translation. To capture all three movement behaviors in a balanced way, we created three sub tasks: horizontal, vertical and diagonal drags. Within each, 9 trials were randomly generated, at two different lengths (4.2, 7.4 and 10.5cm) and at different locations on the screen. This yielded 72 trials per participant. In total, across all three tasks and 11 participants, we collected 1,869,668 sound points (containing 300,827 unique touch points).

6 RESULTS

We report two sets of results from this study. The first set looks at the velocity prediction power of three machine learning models, one trained on acoustic features only, one trained on touch features only, and one trained on both acoustic and touch features. The second set of results looks at the mean euclidean path error of three different prediction models (linear extrapolation, touch ML, and acoustic+touch ML) vs baseline conventional touchscreen operation. We examined this across various amounts of compensated latency, as well as across two different scan rates 50 Hz and 100 Hz. We

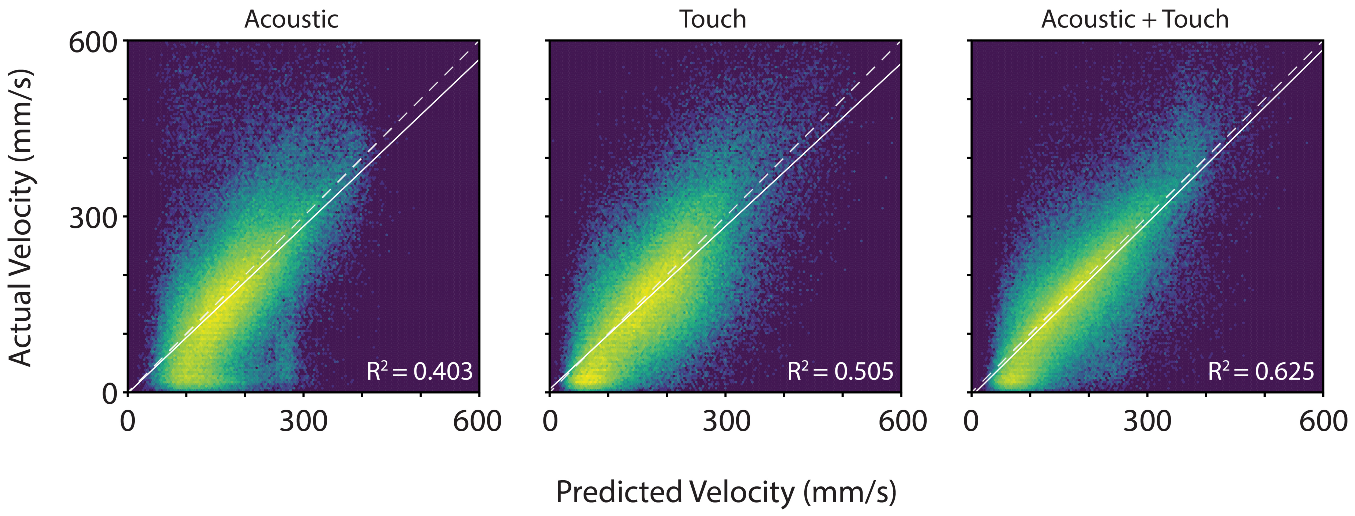


Figure 9: Predicted velocity vs actual velocity for three ML models across all evaluation data. R^2 is reported from a linear regression, as well as the slope of the line fit. Color represents the number of predictions at a given 4mm/s x 4mm/s bin

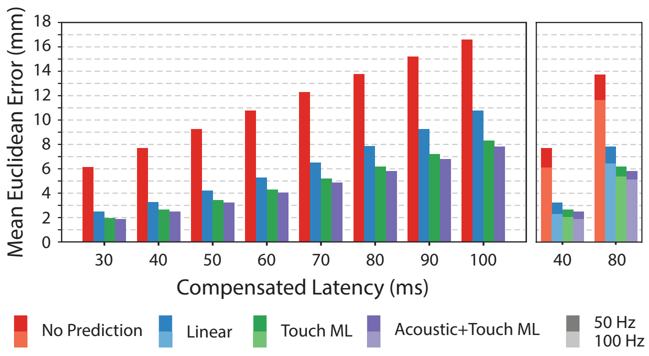


Figure 10: Mean Euclidean error as a function of 8 different compensated latency values 30-100ms. Errors also shown for 40ms and 80ms of touch predictions run at 100 Hz.

omitted every other sample in our touch data as a way of simulating 50 Hz. This was done to closely mimic the typical touch refresh rate (60 Hz) of smartphones and tablets. In our demo system we compute predictions at a rate of 375 Hz and apply a jitter filter, however, for the purposes of this analysis, we used a common prediction rate (of 50 or 100 Hz) and testing dataset across all models for maximum comparability.

We also use the term "compensated latency" in this section, which is analogous to the latency compensation parameter in [36] and the prediction parameter in [19, 31]. Put simply, this is how far into the future the model is predicting. As a reminder, our system was measured in Section 4.3 to have a mean reported touch-to-completed visual rendering (motion-to-photon) time of 96ms, and we set a benchmark of completing rending within one frame (16.6ms) of this time (a general psychophysical limit of dragging latency [10]). This set our benchmark at 80ms of compensated latency.

6.1 Velocity Prediction

As discussed at length previously, our acoustic signal should be descriptive of touch velocity (but somewhat oblivious to absolute direction). As a measure of the predictive power of our models, we plotted all of our models' predictions versus the ground truth velocity 80ms in the future and computed the coefficient of determination (see Figure 9). Predictions were made at a simulated rate of 50 Hz. To aid in viewing the extremely large number of data points, velocities were binned in widths of $4mm/s$, and the number of predictions per bin were logarithmically color mapped. The R^2 values were 0.403, 0.505, and 0.625 respectively for acoustic, touch, and acoustic+touch ML models.

6.2 Path Accuracy Between Methods

As another metric to evaluate our models, we calculated the mean euclidean distance between the predicted touch x, y and the eventual ground truth x, y. This was done for eight compensated latencies (30-100 ms) at a simulated prediction rate of 50 Hz. The baseline "no prediction" case shows the error inherent in the device, which leads to the so-called rubber-banding effect. The linear prediction model is a basic model that takes the previous two touch points, and linearly extrapolates into the future (an approach that tends to overshoot). Our machine learning models take data from touch only, or touch+acoustic data.

Data means are shown in Figure 10. For compensated latency values of [30, 40, 50, 60, 70, 80, 90, 100] ms our means errors (in mm) were [29.3, 36.7, 44.0, 51.3, 58.4, 65.4, 72.3, 79.9] (SD=[18.5, 22.7, 26.7, 30.6, 34.4, 38.1, 41.6, 45.1]) for no prediction, [11.7, 15.4, 19.9, 25,2, 31.0, 37.3, 44.1, 51.3] (SD=[9.34, 12.2, 15.6, 19.4, 23.6, 28.1, 32.8, 37.8]) for linear prediction, [9.25, 12.6, 16.3, 20.3, 24.7, 29.4, 34.4, 39.5] (SD=[7.23, 9.77, 12.5, 15.4, 18.6, 22.1, 25.5, 29.2]) for touch ML, and [8.83, 11.9, 15.3, 19.2, 23.2, 27.7, 32.3, 37.3] (SD=[7.13, 9.54, 12.2, 15.3, 18.3, 21.6, 25.1, 28.5]) for touch+acoustic ML.

As an additional reference, we also calculated the error values for 100 Hz predictions, the native touch report rate of our touchscreen (but above that of any smartphone we tested), at compensated latency values of 40 and 80 ms. The mean error (mm) for these was [29.1, 54.4] (SD=[18.6,33.6]) for no prediction, [10.8, 30.6] (SD=[8.29, 22.7]) for linear prediction, [9.73, 25.5] (SD=[7.18, 18.2]) for touch ML, [9.03, 24.4] (SD=[7.02, 18.6]) for touch+acoustic ML.

7 DISCUSSION

By looking at the scatter plots in Figure 9, it can be seen that our acoustic method clearly has velocity prediction power, and that it is similar to the predictive power of a model utilizing spatial touch data. As a reminder, this model is only taking in 1D acoustic data and is making a prediction of velocity magnitude. Notably, however, our acoustic model has trouble making predictions of velocities below around 50mm/s, and above velocities of around 400mm/s (despite there being actual velocities in this range). Converting these velocities into vibration frequencies, this corresponds to vibrations below 10kHz and above 80kHz. The lower limit aligns with the high-pass cuttoff frequency that was designed into the piezo-preamplifier, while the higher limit corresponds to the audio interface anti-aliasing filter low pass frequency (specified -1dB point is 60kHz). It appears that our model cannot make predictions beyond these points, presumably as there is little input signal.

Looking at the touch only model, it has a slightly higher R^2 value, and no obvious cutoff points like the acoustic only model. The spread of predictions, however, appears wider than the acoustic only model, despite the fact the it has direct access to touch data during training. Looking at the acoustic+touch case, we see that this model is able to make a more correlated prediction of velocity magnitude than either model alone. This shows that not only is there valuable velocity data contained in the acoustic data stream, but that integrating this into the touch data model yields, on average, an improved prediction. Qualitatively, the velocity predictions also look more concentrated along the x=y line, something backed up by the higher value of R^2, and the fit line being more aligned with the x=y axis.

When looking at the plot of mean errors, several trends stand out. First, all models lead to much reduced error from the baseline. This is actually a difference from [20], who saw the baseline out performing the linear prediction model at high latencies (>30ms). This study was on an entirely different hardware and software setup, so it is hard to directly compare, but it is notable that our metric did not reproduce this trend. Moving to a ML model, we see similar improvements as [19], reinforcing the ability and applicability of machine learning to the problem of latency correction. Importantly, we see improvement again moving from a touch only ML model to an acoustic+touch model. The gains are modest, ranging from 4.6-6.1% decrease in error, but they are computed across all conditions, even at the lowest levels of compensated latency. The metric we have selected here may be too coarse to fully express how the addition of the acoustic data helps achieve a lower mean error, but we take this as an encouraging sign in the potential of this technique. Additional trends show that error increases monotonically for all methods as compensated latency increases (as expected) and that a higher touch rate (100 Hz vs 50 Hz) does lead to lower errors.

8 COMMERCIAL INTEGRATION

We wish to stress that this instantiation of TriboTouch is a first prototype, meant for research purposes, and that there remains room for improvement. As a first of its kind hardware system, we assembled it with easily accessible parts and software libraries. In the future, we envision a number of changes to improve system performance, cost, reliability, and usefulness. For example, we have already tested and confirmed that the piezo element can be integrated at a different location of the device (i.e., behind the display or on another rigid part attached to the display). This would allow the sensor to be embedded and protected within the device itself.

We have also investigated the use of etched glass surfaces instead of diffraction grating films. Commercial etched glasses are on the market which have textural features on a similar spatial scale as diffraction gratings. These so-called "anti-glare" cover glass surface treatments are designed to reduce screen glare while preserving high optical quality clarity and transparency (avoiding optical "sparkle" of the display) [4, 21]. In fact, similar processes are already available on consumer devices (such as the 2020 iMac with "nano-textured" glass). Patterning processes are offered by two leading glass manufacturers (Corning[1] and AGC[2]) and are compatible with other "anti-fingerprint" and "anti-reflective" coatings. These glass treatments would mean the TriboTouch layer would be integrated into the device cover glass itself, obviating the need of an additional layer. There are also various surface treatments that manufacturers use on the backs and sides of their devices (e.g. anodizing, sandblasting, or soft touch treatments) which could yield interesting and useful frictional patterns, opening up these surfaces for touch input.

Finally, we take advantage of the low latency in audio processing pipelines, but we, by no means, have performed an exhaustive optimization of audio latency. It's not uncommon for consumer grade hardware and software to provide latencies that are an order of magnitude faster than what we measured (3ms instead of 30ms). Further optimization of this pipeline can be done with additional engineering cost and overhead, and generally requires low-level software/hardware access typically only available to device and component manufacturers. We note that some audio functions, however, have already been engineered to meet some of the most strict latency specifications.

While our current prototype is computationally expensive (running machine learning in Python on a CPU), we envision a commercial implementation using dedicated embedded hardware for prediction. This would not be very architecturally different from how devices are designed today, with an independent touch ASIC that interrupts the main application processor and kernel once touch events have been sensed, processed, and filtered. Indeed, a more computationally-capable touch controller could sample the audio with a dedicated high speed ADC and do the sensor fusion and prediction locally. We believe that with changes such as these, a commercial implementation of TriboTouch is eminently feasible.

[1]https://corning.com/worldwide/en/innovation/materials-science/surface-impressions.html
[2]https://feelinglass.eu/technical-details/

9 LIMITATIONS & FUTURE WORK

TriboTouch is a novel technique, and though we have outlined some paths for improvement in Section 8, there are some limitations to discuss. First and foremost, we are adding additional hardware to touchscreens, which means additional power and cost. We are not, however, adding appreciable weight, or an entirely new electronics challenge. Implementing this technique, therefore, could simply a question of a cost/benefit tradeoff. High performance, lower power audio sampling ASICs already exist, and can be leveraged with minimal technical issue. The piezo sensor material (PZT) is ubiquitous, cheap, and does not require power to operate (only signal buffering), and, as described earlier, the surface pattern layer could simply be integrated into existing etching processes. Even patterning of epoxy resins (such as the method used to create diffraction gratings), is a inexpensive "roll to roll" film process. In terms of reliability, long term durability of this film must be tested, but we saw no dramatic wear patterns during our testing, and did not ever need to replace the film on our prototype hardware. We were able to clean with isopropyl alcohol routinely without damage.

The 192kHz sampling rate we use is high, much higher than most 44.1kHz or 48kHz audio chips, but this is not an inherent limitation with the technique, it is only what we were able to implement here with off-the-shelf materials and components. The sampling rate sets the highest linear velocities that can be sensed. This value can be engineered by altering the surface pattern with only slightly wider elements, for example, 20μm elements would reduce all signal frequencies by a factor of 4, allowing similar velocity information to be sensed with a 48kHz ADC.

The current implementation of TriboTouch is single touch. We did not explicitly investigate multitouch feasibility with this technique, but can envision approaches where multiple acoustic datastreams (via multiple piezos) could be fused with multitouch data to help with individually ascribing sounds with their touch point, essentially turning the problem into one of sound separation. We leave this investigation for future work.

We also report that while the TriboTouch layer leaves minimal tactile, audible, or visual impact on the user, it is not totally imperceptible. The sparkling of the optics with the display can be seen, and surface reflections obviously reveal rainbow patterns which this film was designed for. Additionally, some users can also feel subtle tactile artifacts at the start and stop of their finger on the film, or while moving in circles. Similar artifacts are also emitted audibly from the device, though, generally, this sound is not noticed unless the ear is very close to the surface. With the look and feel of commercial products at such a high bar, these perceptual artifacts might have to be mitigated. We believe, however, that moving to a more randomly rough pattern (that is, an etched surface without such rectangularly arranged features) would help alleviate many of these artifacts.

Looking towards the future, we see multiple avenues for continuing work. Large surfaces may be particularly suited to this technique, as user velocities are larger, and both touch sensor and display rates tend to be lower. Investigating the effectiveness of TriboTouch as displays get larger could yield interesting results. Exploring TriboTouch physics could also yield powerful results. For example, velocity prediction in a single direction with a 1D grating is trivially simple, as linear velocity directly correlates with an object's velocity in that axis. This could lead to cheap, extremely high performance swipe gesture sensors. Understanding the physics behind the effect could also lead to new hybrid ML models, where ML is used for velocity prediction, but then a linear model is used to integrate to an object position. Additionally, an ML model could be trained to recognize various 2D gestures without the need of a full touchscreen.

Any of these techniques could also benefit with the addition of just a few more piezos (i.e., cents worth of hardware), placed at different locations. Initial data suggests a strong correlation between object distance and vibration amplitude. If a relationship between vibration amplitude and object location could be established, then a combination of frequency and relative amplitude between piezo signals could be used to associate a particular sound to a touch location reported by the capacitive screen. When multiple sounds occur, they could be separated and associated with different fingers.

10 CONCLUSION

In this work we introduced a novel touch sensing hardware technique called TriboTouch, and applied it to the use case of touchscreen latency, a longstanding problem in the field. We detailed a combined hardware+software approach where conventional touch and acoustic data are fed into a machine learning model to produce predictions of future touch location. We also outlined the basic principles of TriboTouch, precisely describing how to build a TriboTouch system with cheap and ubiquitous materials and methods. In our evaluations, we show that we can reduce latency from 96 to 16 ms with mean distance error of 5.13mm. Importantly, we show measured improvement associated with incorporating TriboTouch data into the touch prediction pipeline, as our combined model outperforms the touch only models.

ACKNOWLEDGMENTS

Special thanks to Yue Jiang and Sven Mayer, who worked on an early version of this concept with randomly rough surfaces.

REFERENCES

[1] Adnan Akay. 2002. Acoustics of friction. *The Journal of the Acoustical Society of America* 111, 4 (April 2002), 1525–1548. https://doi.org/10.1121/1.1456514 Publisher: Acoustical Society of America.

[2] Glen Anderson, Rina Doherty, and Subhashini Ganapathy. 2011. User Perception of Touch Screen Latency. In *Design, User Experience, and Usability. Theory, Methods, Tools and Practice (Lecture Notes in Computer Science)*, Aaron Marcus (Ed.). Springer, Berlin, Heidelberg, 195–202. https://doi.org/10.1007/978-3-642-21675-6_23

[3] Christiane Attig, Nadine Rauh, Thomas Franke, and Josef F. Krems. 2017. System Latency Guidelines Then and Now – Is Zero Latency Really Considered Necessary?. In *Engineering Psychology and Cognitive Ergonomics: Cognition and Design (Lecture Notes in Computer Science)*, Don Harris (Ed.). Springer International Publishing, Cham, 3–14. https://doi.org/10.1007/978-3-319-58475-1_1

[4] Michael E. Becker. 2015. Sparkle measurement revisited: A closer look at the details. *Journal of the Society for Information Display* 23, 10 (2015), 472–485. https://doi.org/10.1002/jsid.391

[5] Bill Buxton. 2020. Multi-Touch Systems that I Have Known and Loved. http://www.billbuxton.com/multitouchOverview.html

[6] Géry Casiez, Nicolas Roussel, and Daniel Vogel. 2012. 1 € filter: a simple speed-based low-pass filter for noisy input in interactive systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 2527–2530. https://doi.org/10.1145/2207676.2208639

[7] Elie Cattan, Amélie Rochet-Capellan, and François Bérard. 2015. A Predictive Approach for an End-to-End Touch-Latency Measurement. In *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces (ITS '15).*

Association for Computing Machinery, New York, NY, USA, 215–218. https://doi.org/10.1145/2817721.2817747

[8] Elie Cattan, Amélie Rochet-Capellan, Pascal Perrier, and François Bérard. 2015. Reducing Latency with a Continuous Prediction: Effects on Users' Performance in Direct-Touch Target Acquisitions. In *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces (ITS '15)*. Association for Computing Machinery, New York, NY, USA, 205–214. https://doi.org/10.1145/2817721.2817736

[9] Walker David. 2016. Notes on an 'R & J Beck Ltd' replica Rowland diffraction grating. http://www.microscopy-uk.org.uk/mag/artapr11/dw-grating.html

[10] Jonathan Deber, Ricardo Jota, Clifton Forlines, and Daniel Wigdor. 2015. How Much Faster is Fast Enough? User Perception of Latency &amp; Latency Improvements in Direct and Indirect Touch. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1827–1836. https://doi.org/10.1145/2702123.2702300

[11] Benoit Delhaye, Vincent Hayward, Philippe Lefèvre, and Jean-Louis Thonnard. 2012. Texture-induced vibrations in the forearm during tactile exploration. *Frontiers in Behavioral Neuroscience* 6 (2012), 37. https://doi.org/10.3389/fnbeh.2012.00037

[12] Paul Dietz and Darren Leigh. 2001. DiamondTouch: a multi-user touch technology. In *Proceedings of the 14th annual ACM symposium on User interface software and technology (UIST '01)*. Association for Computing Machinery, New York, NY, USA, 219–226. https://doi.org/10.1145/502348.502389

[13] Charles M. Greenspon, Kristine R. McLellan, Justin D. Lieber, and Sliman J. Bensmaia. 2020. Effect of scanning speed on texture-elicited vibrations. *J R Soc Interface* 17, 167 (June 2020), 20190892. https://doi.org/10.1098/rsif.2019.0892

[14] Yizheng Gu, Chun Yu, Zhipeng Li, Weiqi Li, Shuchang Xu, Xiaoying Wei, and Yuanchun Shi. 2019. Accurate and Low-Latency Sensing of Touch Contact on Any Surface with Finger-Worn IMU Sensor. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. Association for Computing Machinery, New York, NY, USA, 1059–1070. https://doi.org/10.1145/3332165.3347947

[15] Jefferson Y. Han. 2005. Low-cost multi-touch sensing through frustrated total internal reflection. In *Proceedings of the 18th annual ACM symposium on User interface software and technology (UIST '05)*. Association for Computing Machinery, New York, NY, USA, 115–118. https://doi.org/10.1145/1095034.1095054

[16] Chris Harrison, Hrvoje Benko, and Andrew D. Wilson. 2011. OmniTouch: wearable multitouch interaction everywhere. In *Proceedings of the 24th annual ACM symposium on User interface software and technology (UIST '11)*. Association for Computing Machinery, New York, NY, USA, 441–450. https://doi.org/10.1145/2047196.2047255

[17] Chris Harrison and Scott E. Hudson. 2008. Scratch input: creating large, inexpensive, unpowered and mobile finger input surfaces. In *Proceedings of the 21st annual ACM symposium on User interface software and technology (UIST '08)*. Association for Computing Machinery, New York, NY, USA, 205–208. https://doi.org/10.1145/1449715.1449747

[18] Chris Harrison, Robert Xiao, and Scott Hudson. 2012. Acoustic barcodes: passive, durable and inexpensive notched identification tags. In *Proceedings of the 25th annual ACM symposium on User interface software and technology (UIST '12)*. Association for Computing Machinery, New York, NY, USA, 563–568. https://doi.org/10.1145/2380116.2380187

[19] Niels Henze, Markus Funk, and Alireza Sahami Shirazi. 2016. Software-reduced touchscreen latency. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '16)*. Association for Computing Machinery, New York, NY, USA, 434–441. https://doi.org/10.1145/2935334.2935381

[20] Niels Henze, Sven Mayer, Huy Viet Le, and Valentin Schwind. 2017. Improving software-reduced touchscreen latency. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '17)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3098279.3122150

[21] Darren K. P. Huckaby and Darran R. Cairns. 2009. 36.2: Quantifying "Sparkle" of Anti-Glare Surfaces. *SID Symposium Digest of Technical Papers* 40, 1 (2009), 511–513. https://doi.org/10.1889/1.3256829

[22] Hiroshi Ishii, Craig Wisneski, Julian Orbanes, Ben Chun, and Joe Paradiso. 1999. PingPongPlus: design of an athletic-tangible interface for computer-supported cooperative play. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems (CHI '99)*. Association for Computing Machinery, New York, NY, USA, 394–401. https://doi.org/10.1145/302979.303115

[23] Ricardo Jota, Albert Ng, Paul Dietz, and Daniel Wigdor. 2013. How fast is fast enough? a study of the effects of latency in direct-touch pointing tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 2291–2300. https://doi.org/10.1145/2470654.2481317

[24] Topi Kaaresoja and Stephen Brewster. 2010. Feedback is... late: measuring multimodal delays in mobile device touchscreen interaction. In *International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction (ICMI-MLMI '10)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/1891903.1891907

[25] Shaun K. Kane, Daniel Avrahami, Jacob O. Wobbrock, Beverly Harrison, Adam D. Rea, Matthai Philipose, and Anthony LaMarca. 2009. Bonfire: a nomadic system for hybrid laptop-tabletop interaction. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology (UIST '09)*. Association for Computing Machinery, New York, NY, USA, 129–138. https://doi.org/10.1145/1622176.1622202

[26] Ryosuke Kawakatsu and Shigeyuki Hirai. 2018. Rubbinput: An Interaction Technique for Wet Environments Utilizing Squeak Sounds Caused by Finger-Rubbing. In *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. 512–517. https://doi.org/10.1109/PERCOMW.2018.8480335

[27] Usman Khan, Tae-Ho Kim, Hanjun Ryu, Wanchul Seung, and Sang-Woo Kim. 2017. Graphene Tribotronics for Electronic Skin and Touch Screen Applications. *Advanced Materials* 29, 1 (2017), 1603544. https://doi.org/10.1002/adma.201603544

[28] Ji-Eun Kim, John Sunwoo, Yong-Ki Son, Dong-Woo Lee, and Il-Yeon Cho. 2007. A gestural input through finger writing on a textured pad. In *CHI '07 Extended Abstracts on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 2495–2500. https://doi.org/10.1145/1240866.1241030

[29] Takumi Kusano and Takashi Komuro. 2015. 3D Tabletop User Interface with High Synchronization Accuracy using a High-speed Stereo Camera. In *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces (ITS '15)*. Association for Computing Machinery, New York, NY, USA, 39–42. https://doi.org/10.1145/2817721.2817724

[30] Gierad Laput and Chris Harrison. 2019. SurfaceSight: A New Spin on Touch, User, and Object Sensing for IoT Experiences. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300559

[31] Huy Viet Le, Valentin Schwind, Philipp Göttlich, and Niels Henze. 2017. PredicTouch: A System to Reduce Touchscreen Latency using Neural Networks and Inertial Measurement Units. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces (ISS '17)*. Association for Computing Machinery, New York, NY, USA, 230–239. https://doi.org/10.1145/3132272.3134138

[32] Darren Leigh, Clifton Forlines, Ricardo Jota, Steven Sanders, and Daniel Wigdor. 2014. High rate, low-latency multi-touch sensing with simultaneous orthogonal multiplexing. In *Proceedings of the 27th annual ACM symposium on User interface software and technology (UIST '14)*. Association for Computing Machinery, New York, NY, USA, 355–364. https://doi.org/10.1145/2642918.2647353

[33] I. Scott MacKenzie and R. William Soukoreff. 2003. Phrase sets for evaluating text entry techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 754–755. https://doi.org/10.1145/765891.765971

[34] Louise R. Manfredi, Hannes P. Saal, Kyler J. Brown, Mark C. Zielinski, John F. Dammann, Vicky S. Polashock, and Sliman J. Bensmaia. 2014. Natural scenes in tactile texture. *Journal of Neurophysiology* 111, 9 (May 2014), 1792–1802. https://doi.org/10.1152/jn.00680.2013 Publisher: American Physiological Society.

[35] Roderick Murray-Smith, John Williamson, Stephen Hughes, and Torben Quaade. 2008. Stane: synthesized surfaces for tactile input. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. Association for Computing Machinery, New York, NY, USA, 1299–1302. https://doi.org/10.1145/1357054.1357257

[36] Mathieu Nancel, Stanislav Aranovskiy, Rosane Ushirobira, Denis Efimov, Sebastien Poulmane, Nicolas Roussel, and Géry Casiez. 2018. Next-Point Prediction for Direct Touch Using Finite-Time Derivative Estimation. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology (UIST '18)*. Association for Computing Machinery, New York, NY, USA, 793–807. https://doi.org/10.1145/3242587.3242646

[37] Albert Ng, Michelle Annett, Paul Dietz, Anoop Gupta, and Walter F. Bischof. 2014. In the blink of an eye: investigating latency perception during stylus interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. Association for Computing Machinery, New York, NY, USA, 1103–1112. https://doi.org/10.1145/2556288.2557037

[38] Albert Ng, Julian Lepinski, Daniel Wigdor, Steven Sanders, and Paul Dietz. 2012. Designing for low-latency direct-touch input. In *Proceedings of the 25th annual ACM symposium on User interface software and technology (UIST '12)*. Association for Computing Machinery, New York, NY, USA, 453–464. https://doi.org/10.1145/2380116.2380174

[39] Jakob Nielsen. 1994. *Usability Engineering*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[40] J. A. Paradiso, K. Hsiao, J. Strickon, J. Lifton, and A. Adler. 2000. Sensor systems for interactive surfaces. *IBM Systems Journal* 39, 3.4 (2000), 892–914. https://doi.org/10.1147/sj.393.0892 Conference Name: IBM Systems Journal.

[41] Joseph A. Paradiso, Che King Leo, Nisha Checka, and Kaijen Hsiao. 2002. Passive acoustic knock tracking for interactive windows. In *CHI '02 Extended Abstracts on Human Factors in Computing Systems (CHI EA '02)*. Association for Computing Machinery, New York, NY, USA, 732–733. https://doi.org/10.1145/506443.506570

[42] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron

Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12, 85 (2011), 2825–2830. http://jmlr.org/papers/v12/pedregosa11a.html

[43] D. T. Pham, M. Al-Kutubi, Z. Ji, M. Yang, Z. Wang, and S. Catheline. 2005. Tangible acoustic interface approaches. In *Proceedings of IPROMS 2005 Virtual Conference*. Citeseer, 497–502.

[44] D. T. Pham, Z. Ji, O. Peyroutet, M. Yang, Z. Wang, and M. Al-kutubi. 2006. Localisation of impacts on solid objects using the Wavelet Transform and Maximum Likelihood Estimation. In *Intelligent Production Machines and Systems*, D. T. Pham, E. E. Eldukhri, and A. J. Soroka (Eds.). Elsevier Science Ltd, Oxford, 541–547. https://doi.org/10.1016/B978-008045157-2/50095-X

[45] Jun Rekimoto. 2002. SmartSkin: an infrastructure for freehand manipulation on interactive surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '02)*. Association for Computing Machinery, New York, NY, USA, 113–120. https://doi.org/10.1145/503376.503397

[46] Walter Ritter, Guido Kempter, and Tobias Werner. 2015. User-Acceptance of Latency in Touch Interactions. In *Universal Access in Human-Computer Interaction. Access to Interaction (Lecture Notes in Computer Science)*, Margherita Antona and Constantine Stephanidis (Eds.). Springer International Publishing, Cham, 139–147. https://doi.org/10.1007/978-3-319-20681-3_13

[47] Naoki Sugita, Daisuke Iwai, and Kosuke Sato. 2008. Touch sensing by image analysis of fingernail. In *2008 SICE Annual Conference*. 1520–1525. https://doi.org/10.1109/SICE.2008.4654901

[48] Geoff Walker. 2012. A review of technologies for sensing contact location on the surface of a display. *Journal of the Society for Information Display* 20, 8 (2012), 413–440. https://doi.org/10.1002/jsid.100

[49] Wayne Westerman, John G. Elias, and Alan Hedge. 2001. Multi-Touch: A New Tactile 2-D Gesture Interface for Human-Computer Interaction. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 45, 6 (Oct. 2001), 632–636. https://doi.org/10.1177/154193120104500612 Publisher: SAGE Publications Inc.

[50] Michaël Wiertlewski, Charles Hudin, and Vincent Hayward. 2011. On the 1/f noise and non-integer harmonic decay of the interaction of a finger sliding on flat and sinusoidal surfaces. In *2011 IEEE World Haptics Conference*. 25–30. https://doi.org/10.1109/WHC.2011.5945456

[51] Andrew D. Wilson. 2004. TouchLight: an imaging touch screen and display for gesture-based interaction. In *Proceedings of the 6th international conference on Multimodal interfaces (ICMI '04)*. Association for Computing Machinery, New York, NY, USA, 69–76. https://doi.org/10.1145/1027933.1027946

[52] Andrew D. Wilson. 2005. PlayAnywhere: a compact interactive tabletop projection-vision system. In *Proceedings of the 18th annual ACM symposium on User interface software and technology (UIST '05)*. Association for Computing Machinery, New York, NY, USA, 83–92. https://doi.org/10.1145/1095034.1095047

[53] Haijun Xia, Ricardo Jota, Benjamin McCanny, Zhe Yu, Clifton Forlines, Karan Singh, and Daniel Wigdor. 2014. Zero-latency tapping: using hover information to predict touch locations and eliminate touchdown latency. In *Proceedings of the 27th annual ACM symposium on User interface software and technology (UIST '14)*. Association for Computing Machinery, New York, NY, USA, 205–214. https://doi.org/10.1145/2642918.2647348

[54] Robert Xiao, Chris Harrison, and Scott E. Hudson. 2013. WorldKit: rapid and easy creation of ad-hoc interactive applications on everyday surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 879–888. https://doi.org/10.1145/2470654.2466113

[55] Robert Xiao, Scott Hudson, and Chris Harrison. 2016. DIRECT: Making Touch Tracking on Ordinary Surfaces Practical with Hybrid Depth-Infrared Sensing. In *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces (ISS '16)*. Association for Computing Machinery, New York, NY, USA, 85–94. https://doi.org/10.1145/2992154.2992173

[56] Robert Xiao, Greg Lew, James Marsanico, Divya Hariharan, Scott Hudson, and Chris Harrison. 2014. Toffee: enabling ad hoc, around-device interaction with acoustic time-of-arrival correlation. In *Proceedings of the 16th international conference on Human-computer interaction with mobile devices & services (MobileHCI '14)*. Association for Computing Machinery, New York, NY, USA, 67–76. https://doi.org/10.1145/2628363.2628383

[57] Yang Zhang and Chris Harrison. 2018. Pulp Nonfiction: Low-Cost Touch Tracking for Paper. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3173574.3173691

[58] Yang Zhang, Gierad Laput, and Chris Harrison. 2017. Electrick: Low-Cost Touch Sensing Using Electric Field Tomography. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3025453.3025842