

TapSense: Enhancing Finger Interaction on Touch Surfaces

Chris Harrison Julia Schwarz Scott E. Hudson

Human-Computer Interaction Institute and Heinz College Center for the Future of Work
Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh PA 15213
{chris.harrison, julia.schwarz, scott.hudson}@cs.cmu.edu

ABSTRACT

We present *TapSense*, an enhancement to touch interaction that allows conventional surfaces to identify the type of object being used for input. This is achieved by segmenting and classifying sounds resulting from an object's impact. For example, the diverse anatomy of a human finger allows different parts to be recognized – including the tip, pad, nail and knuckle – without having to instrument the user. This opens several new and powerful interaction opportunities for touch input, especially in mobile devices, where input is extremely constrained. Our system can also identify different sets of passive tools. We conclude with a comprehensive investigation of classification accuracy and training implications. Results show our proof-of-concept system can support sets with four input types at around 95% accuracy. Small, but useful input sets of two (e.g., pen and finger discrimination) can operate in excess of 99% accuracy.

ACM Classification: H.5.2 [Information interfaces and presentation]: User Interfaces - Graphical user interfaces; Input devices and strategies.

General terms: Human Factors

Keywords: Acoustic classification, tabletop computing, interactive surfaces, tangibles, tools, pens, stylus, finger, multi-user, touchscreen, collaborative, input.

INTRODUCTION

Computers are increasingly featuring direct touch interfaces, found in forms as diverse as kiosks and interactive tabletops, to tablet computers and handheld mobile devices. At present, finger input on touchscreens is handled very simplistically, essentially boiled down to an X/Y coordinate. However, human fingers are remarkably sophisticated, both in their anatomy and motor capabilities. We can form them into many poses and perform a wide variety of gestures.

We present an enhancement to touchscreen interaction that enables identification of the object used for input. Our system can recognize small sets of passive tools as well as discriminate different parts of the finger – tip, pad, knuckle

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. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

UIST'11, October 16–19, 2011, Santa Barbara, CA, USA.
Copyright © 2011 ACM 978-1-4503-0716-1/11/10... \$10.00.

and nail (Figure 1 and 2). The latter is especially valuable on mobile devices, where input bandwidth is limited due to small screens and “fat fingers” [16]. For example, a knuckle tap could serve as a “right click” for mobile device touch interaction, effectively doubling input bandwidth. Right-click-like functionality is currently achieved on touch surfaces with fairly unintuitive and un-scalable chording of fingers and tap-and-hold interactions. Finally, our approach requires no electronics or sensors to be placed on the user.

RELATED APPROACHES

Many technologies exist that have the ability to digitize different types of input. There are two main touch sensing approaches: active and passive.

The key downside of active approaches is that an explicit object must be used (e.g., a special pen), which is implemented with electronics (and batteries if not tethered). For example, pens augmented with infrared light emitters on their tips can be used on the commercially available Microsoft Surface [15]. There have also been efforts to move beyond pens, including, e.g., infrared-light-emitting brushes for painting applications [27]. Current systems generally do not attempt to discriminate among different pens (just perhaps pen from finger input). Variably-modulated infrared light enables identification, but requires specialized hardware. Additionally, ultrasonics can be used for input localization [19], and can provide pen ID as well. Capacitive coupling in [6,7] allows users or objects to be localized and identified, though this requires grounding plates or a physical connection to function.

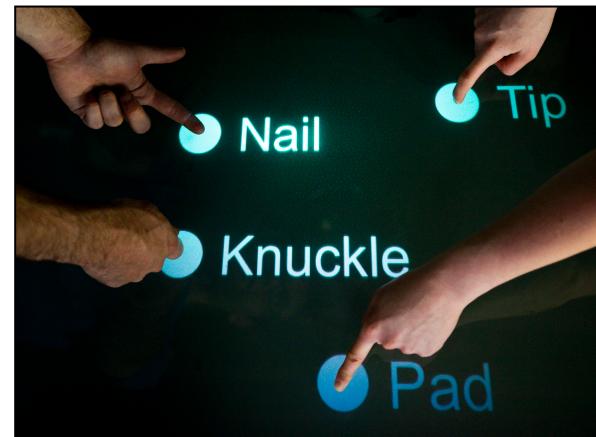


Figure 1. *TapSense* can classify different types of finger input.

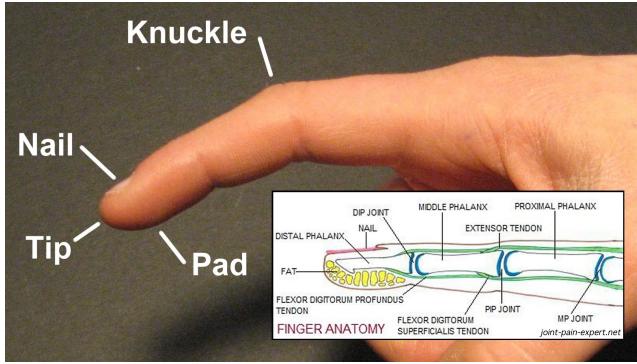


Figure 2. Different parts of the finger can be identified for use on touch surfaces.

Sensing based on electromagnetic resonance operates between active and passive. Although the tools and tangibles need not be powered, they contain a resonant coil that is excited by proximate EM fields, generated by the specialized tablet they operate on. Although highly capable, including the ability to provide ID, table-sized setups are prohibitively expensive at present. It is also possible to support object identification through a combination of RFID and vision processing [20], which offers greater scalability.

Fiducial markers are a truly passive approach. They provide the ID of an object through a uniquely patterned tag – often in a sticker form factor [17,24]. This method has been shown to be very capable – the only major drawback is the size of the marker, which in general, prevents placement on small surfaces like the tip of a pen. Fiducial markers can also work in a capacitive-sensing manner [26], allowing tags to be embedded in an object. Additionally, the shape of an object can be captured optically and used for classification (e.g., mice and keyboards in [12]).

In general, the aforementioned techniques require instrumentation of the object providing input, which is a non-starter for fingers. To side step this, Hambone [5] uses wrist-mounted acoustic sensors to classify finger-on-finger actions, such as pinching or flicking. Finger taps can also be localized on the body through acoustic fingerprinting [10]. However, the latter systems require sensors to be placed on the user (a key objective of *TapSense* was that users did not need to be instrumented). Without instrumentation, some areas of the finger can be determined through computer vision (e.g., pad vs. tip) [16,28]. Using accelerometers, soft and hard taps can be discriminated [13].

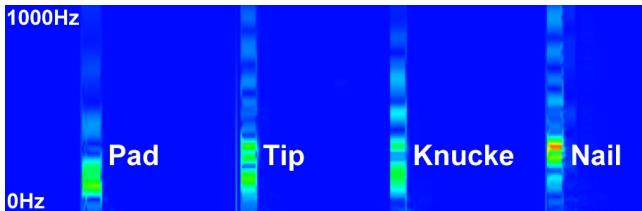


Figure 3. Spectrograms of our four finger input types.

Finally, The Interactive Window Project [21,22] uses time of flight to localize touch events on a large pane of glass. The work also briefly mentions that frequency distribution can be used to classify touch type, such as a finger vs. fist bash, an approach seemingly similar to *TapSense*. Unfortunately, no details are provided about classification or training. Further, no accuracy results are provided, making the system impossible to compare, evaluate, or improve upon.

SENSING AND PROCESSING

Our approach is comprised of two key processes operating in concert. The first is some method for detecting and tracking the position of input, either single- or multi-touch. A variety of existing technologies could be used, including optical, resistive, and capacitive touchscreens. The second component – and the chief contribution of this paper – listens, segments and classifies impacts on the interactive surface using acoustic features. It relies on the physical principle that different materials produce different acoustic signatures and have different resonant frequencies (Figure 3). Once a classification has been made, it is paired with the last event from the touch surface.

For acoustic sensing, we use a conventional medical stethoscope coupled with an inexpensive electret condenser microphone, seen in Figure 4 (many microphone technologies are suitable, as well as accelerometers). This is affixed to the surface of an input-capable display. This setup resembles [11], which uses the same sensor and also takes advantage of (solid) surface acoustic transmission, albeit with a different objective (gesture recognition on ad hoc surfaces vs. tip recognition on touchscreens).

When an object strikes the surface, an ensemble of mechanical vibrations propagate outward through the material. Typically, interactive surfaces use rigid materials, such as plastic or glass, which both quickly distribute and faithfully preserve the signal. As demonstrated in [10,21] (also passive acoustic based), the strength of taps does not need to be great. Indeed, one needs to tap no harder than required to type on a keyboard.

A key property of this approach is that items striking the surface do not require active components. Input objects are simply composed of different materials and are entirely passive.

Segmentation of input signals is straightforward. Stethoscopes naturally provide a high level of environmental noise suppression. This allows impacts to be readily segmented from any background noise with a simple amplitude threshold. Once the audio data for the impact has been captured, our software processes it, extracting a series of time-independent acoustic features.

We recommend sampling at 96KHz, using a sliding window of 4096 (the first 43ms of impact signal). An Fast Fourier Transform (FFT) of this window produces 2048 bands of frequency power; we discard all but the lower 500 bands, representing the acoustic power from 0 to ~10kHz

(though the lower 1kHz is the most expressive; see Figure 3). We then down-sample this data into an additional vector of length 50 (i.e., buckets of ten), providing a different aliasing. We also include the average absolute amplitude, total absolute amplitude, standard deviation of the absolute amplitude, and center of mass for both the segmented input signal and the FFT (8 features). Lastly, we also calculate the fundamental frequency of the impact waveform. These features are largely based on those employed successfully in Skinput [10], which used a similar acoustic approach for localizing inputs on the skin.

This process yields 559 features, on which we rely for classification. We use a support vector machine (SVM) implementation provided by the Weka Machine Learning toolkit [9] (SMO; polynomial kernel with default parameters). A full description of SVMs is beyond the scope of this paper; see [3] for a tutorial (note: as will be discussed in more detail later, before the classifier can be used, it must first be trained on each class' acoustic signature).

Once a classification has been made, the resulting type is used to label an unassigned input point (digitized by one of many touch sensing technologies and reported to our software). This matching process could be done several ways – we found selecting the input event with the closest onset timestamp was sufficient. The entire classification process, starting from the onset of impact, can occur in approximately 100ms, allowing for real-time interaction.

In practice, this method appears to work fairly well, and enables several users with multiple input objects to work on a common surface simultaneously. However, there is one special case where this process breaks down and for which there is no immediate solution – timing collisions. In particular, if two objects strike the surface in sufficiently close temporal proximity, their acoustic signals will not be segmented separately, and therefore not allow for accurate classification.

The duration of this period is defined by four factors: 1) the aggressiveness of the segmenter (because we use time-independent acoustic features, classification could poten-

tially use e.g., the first 10ms of audio), 2) the resonant frequencies of the impact material (lower frequencies take longer to dissipate), 3) the dampening of the surface (to dissipate previous impacts) and 4) the size of the surface (more mass takes longer to dampen). With a carefully selected input set (almost exclusively materials with higher frequency resonance), a safe separation between impacts could be roughly 300ms on a large surface, such as a table or wall.

An alternative solution is to employ sophisticated algorithms that can localize impact sounds and separate them into distinct acoustic waveforms [1]. Although this is generally applied to environmental noises, such as speech, the principles should also apply to touch surfaces.

Fortunately, this issue is mostly moot on mobile devices, which, due to their small size and light weight, quickly diminish acoustic energy. Taps can occur as close as ~50ms apart on our mobile setup. Furthermore, mobile devices typically have a single user. This almost entirely eliminates the possibility of simultaneous impacts (the fastest the authors could double finger tap was 66ms - achieved in a contrived way).

PROOF-OF-CONCEPT SETUPS

To evaluate the scalability of our approach and better consider fruitful application areas, we constructed two proof-of-concept, *TapSense*-augmented, interactive surfaces: a hand-held mobile device and a full-sized multitouch table.

Mobile Device

To evaluate the performance of our approach on handheld devices, we instrumented an Apple iPod Touch (Figure 4). We use the iPod's 7.6 x 5.1 cm capacitive screen for input tracking. We connect our acoustic sensor to a conventional computer, which also runs our classifier and interface demos. To provide a graphical interface on the iPod, we simply VNC to the aforementioned computer. Although introducing some latency, this afforded us a common code base for our mobile device and multitouch table, enabling faster prototyping.

In a real product, a small microphone would likely be coupled to the screen at the cost of a few tens of cents. It may even be possible to use built-in microphones found in e.g., mobile phones. During early prototyping, it appeared that commercial devices applied various software/hardware filters for superior voice capture and ambient noise suppression, which degraded performance in the acoustic space we wished to use. However, with raw access to a device's microphone, it seems likely *TapSense* could be enabled with no additional hardware. Finally, it may be possible to realize superior sensing resolution through the use of multiple microphones [21,22].

Multitouch Table

We also built a tabletop multitouch setup (Figure 4) to 1) evaluate how *TapSense* scaled to larger surfaces, 2) consid-

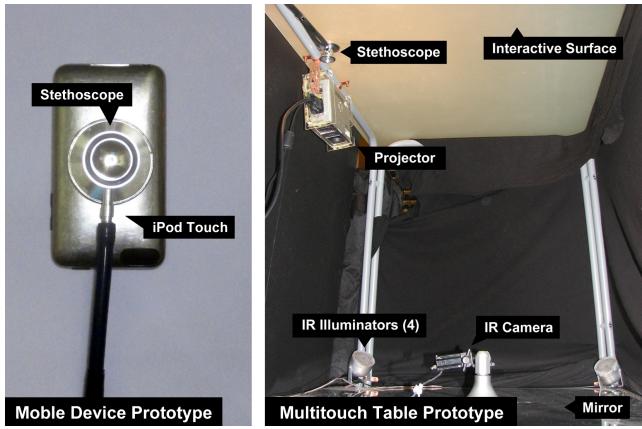


Figure 4. Chief components of our two prototype setups.

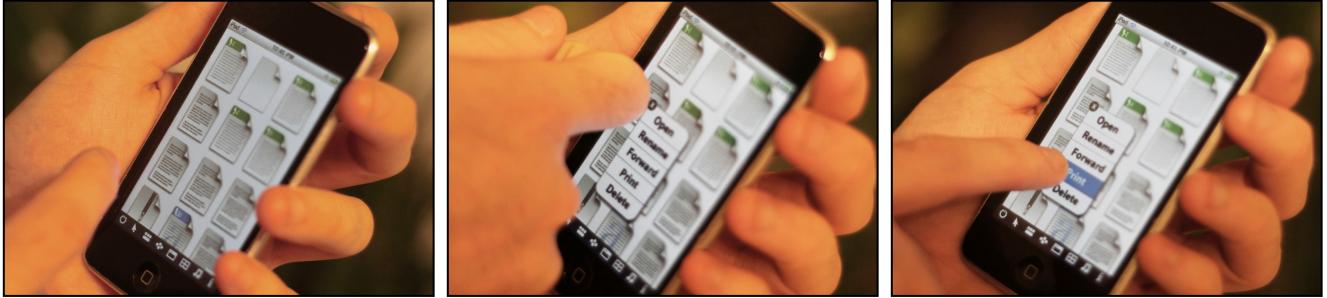


Figure 5. Left: a directory of files; items can be opened or dragged using the traditional finger pad tap. Center: user alt-clicks a file with a knuckle tap, triggering a contextual menu. Right: User pad taps on the print option.

er collaborative applications (i.e., several collocated users), and 3) explore the uses and performance of handheld tools.

The tabletop surface consists of a 110 x 75 cm frosted glass sheet that provides a diffuse surface for both diffused illumination sensing [18] and graphical projection. In brief testing, plastics such as acrylic and polycarbonate appear to work just as well acoustically – we chose glass because of availability, low cost, and scratch resistance. Furthermore, the technique should function on curved or irregular interactive surfaces (e.g., [2]).

To create a table-like form factor, the glass surface is supported by a simple metal frame. To reduce undesirable chattering, the glass is supported on the frame by several rubber nubs. This also helped to isolate the touch surface from the support members and dampen mechanical energy from impacts. We centered our acoustic sensor above the interactive area, though many placements are possible.

FINGERS

Contemporary interactive surfaces generally treat finger touches as a single class of input (a partial exception to this is [16], which captures a high-resolution fingerprint image to infer the 3D “posture” of a finger; area of contact via optical sensing is used as an extra input dimension in [4,28]). However, this is a gross simplification - fingers are diverse appendages, both in their motor capabilities and their anatomical composition. Indeed, a single digit contains one major and two minor knuckles, a boney tip, a fleshy pad, and a fingernail – most of which can be readily discriminated by our acoustic approach. This ability to identify which part of the finger was used for input is unique to our system. Supporting additional dimensions of finger input has largely been ignored because instrumenting the user with active or passive components is invasive.

This approach has the potential to mitigate two significant problems faced in touch interaction:

1) Finger Overloading: At present, in order for a finger to perform different operations at a single point in space, it must be overloaded, typically triggered by a tap-and-hold period or chording of the fingers (e.g., two-finger-tap for right click). This often then triggers a transient contextual menu, which allows a user to select one of several actions.

The power of contextual menus could be easily coupled to a finger “right click” (e.g., knuckle tap). The conventional finger pad tap could then operate as usual (selection, opening, dragging). The value of mice with two buttons is clear in desktop-class interaction. *TapSense* immediately enables this (and several additional click types) for touch screen input, with no user instrumentation. Figure 5 illustrates a simple sequence (see Video Figure for extended examples).

2) Breaking Out Functionality: An alternative to finger overloading is breaking functionality out into one or more buttons – for example, a button for minimizing a window and another for closing it. However, this is problematic for mobile devices with limited screen real estate.

TapSense offers a solution that entails no buttons. For example, the finger pad could operate as usual. However, a finger nail tap anywhere on the surface could trigger a minimization or “go back” action. Window maximization or “go forward” could be done similarly with a knuckle tap.

Example Finger Applications

TapSense is an enabling technique that can be used in a wide variety of application domains and use contexts. To help motivate our approach and underscore its utility, we developed several example applications we believe to be particularly interesting.

Soft keyboards on mobile devices are particularly problematic – there are many keys that need to be provided and very little space. In response, keyboards are typically broken up into several “pages” of keys, toggled with modal buttons. Not only does this add extra clicks to typing interactions, but also further crowds the small screen.

As a demonstration, we developed a *TapSense*-augmented soft keyboard that aimed to alleviate some of these problems. It features two key sets that operate in parallel, seen in Figure 6. To type a primary character, users can use their finger pad as usual. To type an alt character, a finger tip is used. Thus, users have access to the entire character set without having to switch pages. To backspace, the user can nail tap anywhere on the screen (i.e., no button necessary).

Another application we developed was a simple painting interface. To draw freehand, the user simply uses their finger pad like a brush. To draw line segments, a user tip



Figure 6. Our *TapSense*-augmented soft keyboard.

taps the screen and then drags to a desired location. To undo the last stroke, users can nail tap anywhere on the screen. This, like our keyboard demo, illustrates a simple way to remove modal buttons from the interaction and push this complexity to our highly dexterous fingers. Other interactions could involve rapid switching between tools (e.g., fill tool, erase tool) and modes (e.g., brush thickness, opacity, color).

Leveraging Existing Finger Behaviors

It is interesting to note that humans use different parts of their fingers in different ways – to scratch an itch, type on a keyboard, tap a co-worker on the shoulder, or knock on a door. With careful design, it may be possible to leverage these norms such that existing finger behaviors could be ported to and made relevant in digital domains.

For example, consider a system where a knuckle “knock” is used to open files or applications. A tap with the tip of the finger (i.e., poke) could be used to bring something to attention, perhaps maximized or given focus, whereas a fingernail “flick” could be used to minimize or delete an item. This functionality could operate in harmony with conventional finger-driven interfaces, which tend to rely on finger pads for pointing and “clicking”.

PENS, POINTERS, STAMPS AND TOOLS

Humans have remarkable dexterity with handheld tools and, unsurprisingly, numerous research projects have introduced physical manipulators to interactive systems (see e.g., [14,15,17,23,25]). These often come in the form of pens, pointing devices, stamps (e.g., for instantiation) and miscellaneous tools (e.g., dials, sliders, guides).

Such items could easily incorporate acoustically-distinct materials, and be made small and pointy, like real pens and pencils. These would be extremely durable and inexpensive to mass produce. To accompany our multitouch table setup, we built a set of input objects, shown in Figure 7. These are simply different materials glued to the heads of dry erase markers, and were not specially engineered to achieve top performance.

Having even a small set of input objects could be valuable. For instance, painting applications on conventional interactive surfaces typically use a palette-based color mode selec-

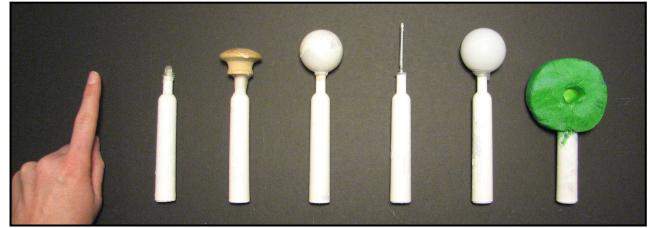


Figure 7. Finger and six tools with different materials affixed to their tips. Left to right: finger, polycarbonate nub, wood knob, acrylic ball, metal screw, ping-pong ball, foam.

tion. When a color is chosen, all users are forced to switch to this color. However, our system allows multiple users to pick up one of several color “brushes” and use them simultaneously (Figure 8). We also created a tool with different materials on each end, which could be used to draw and then flipped around to erase, like a pencil.

Another possibility is to assign users uniquely identified input tools. This would allow actions on a system to be attributed to a particular person (Figure 8) – a capability shared with [7]. This could be used for e.g., collaborative document editing, individualized undo stacks, and read/write permissions; see [8] for more complete discussion of techniques for collaboration in shared workspaces.

EVALUATION

To understand the feasibility and accuracy of our approach, we collected data from 18 participants. Nine participants (four female) completed the study on our prototype mobile device, and another nine participants (five female) completed the study using our multitouch table setup. Participants were paid \$10 for their involvement.

For the mobile device, we evaluated an input set comprised of four finger locations: pad, tip, knuckle and nail (Figure 2). Two participants had long nails that prevented them from tapping with the boney tip of their fingers. No special changes were made for these users – they were simply allowed to tap using the tip of their nail (i.e. perpendicular to the touch surface, distinct from the nail tap, which required the hand to be palms-up). Anecdotally, users had no trouble performing these different taps, perhaps because they leverage existing finger behaviors (see section on left).

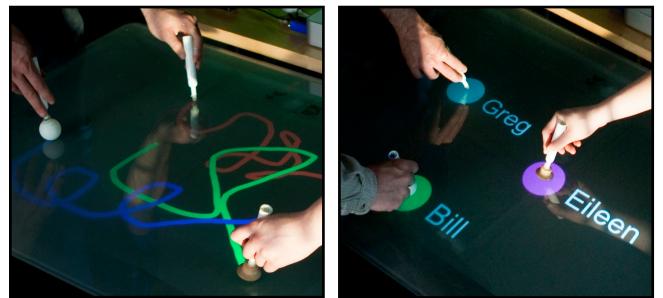


Figure 8. Left: tools representing different “brush” colors allow several users to paint simultaneously, without color or mode switching. Right: unique pens could allow interactive surfaces to identify which user was performing what action.

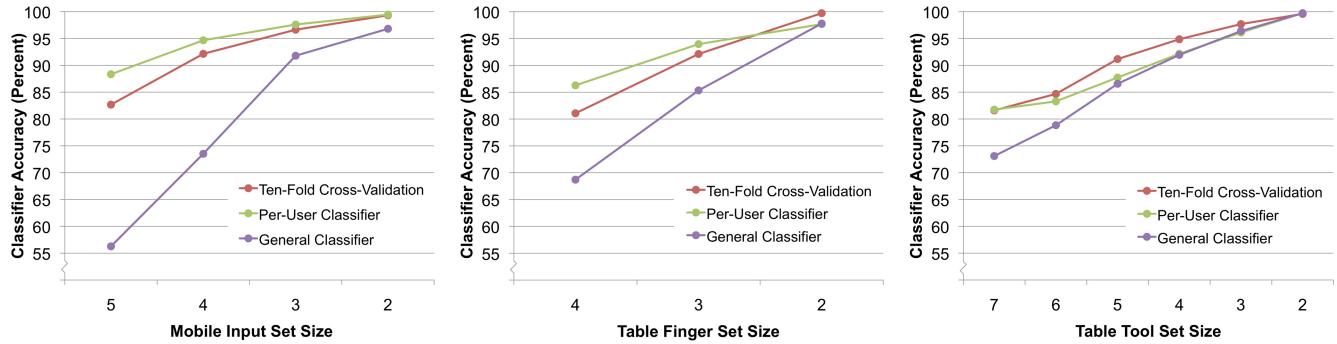


Figure 9. Classification accuracy results for input sets of different sizes. Three classification approaches are evaluated.

As an additional, fifth input type, we used an iPod compatible capacitive stylus (which uses conductive foam inside). Styli, although increasingly uncommon in mobile devices, have attractive qualities for some interactive applications [14,15]. For these applications, it would be useful to discriminate between finger and pen input.

For our multitouch table, two input sets were evaluated: our four finger types (tip, pad, knuckle and nail – Figure 2) and seven input tools (Figure 7). The finger was included in the latter tool set since it is unlikely an interactive surface would want to give up finger discrimination capability, even if several tools were present.

For each input type, participants were asked to provide ten taps to different locations on the screen. Participants completed this sequence four times, going through all input types first, before looping back around to complete the next round (this ensured better temporal independence). This produced 40 data points per input type. Although our classifier would likely benefit from additional training data, we found user boredom and fatigue became problematic in longer study durations. Thus, results presented in this paper should be considered as an accuracy lower bound.

This procedure produced, per participant, 200 data points for the *mobile input set*, 160 data points for the *table finger set*, and 280 data points for the *table tool set*. In total, 5760 tap events were collected.

RESULTS

The central questions we sought to answer in our evaluation were 1) the accuracy of our approach and 2) how large of an input set could be supported with this acoustic approach. To assess this, we purposely created input sets

larger than what we believed our system could accurately classify. This allowed us to post-hoc prune down the input sets to analyze how performance would improve. Ultimately, this procedure allows a size of set vs. accuracy balance to be identified.

We evaluated the classification performance of the *mobile input set* at four different sizes: 5, 4, 3, and 2 input types. The process was initiated by including all five input types. In each subsequent round, we removed the input type with the least accuracy (i.e. highest confusion). The only location not allowed to be eliminated was the finger pad, as this was the standard pointing modality. The same procedure was used for the *table finger set* (4 types) and the *table tool set* (7 types). In the *table tool set*, the finger was not allowed to be eliminated; likewise for pad location in the *table finger set*.

Ten-Fold Cross Validation

To get a general sense of our system's performance, we conducted a conventional ten-fold cross-validation using all of our data (1800, 1440 and 2520 data points for the *mobile input set*, *table finger set* and *table tool set* respectively). However, this statistic tends to be generous, since train and test data sets can contain points from the same user and points adjacent in time (which will naturally tend to be more similar). Nonetheless, it provides a good baseline - this result is illustrated in Figure 9.

Per-User Classifiers

To better understand how our system would perform in the real world, we assessed the accuracy of user-specific classifiers. Specifically, we trained the SVM on three rounds of a user's input data, and tested on the fourth. We evaluated all

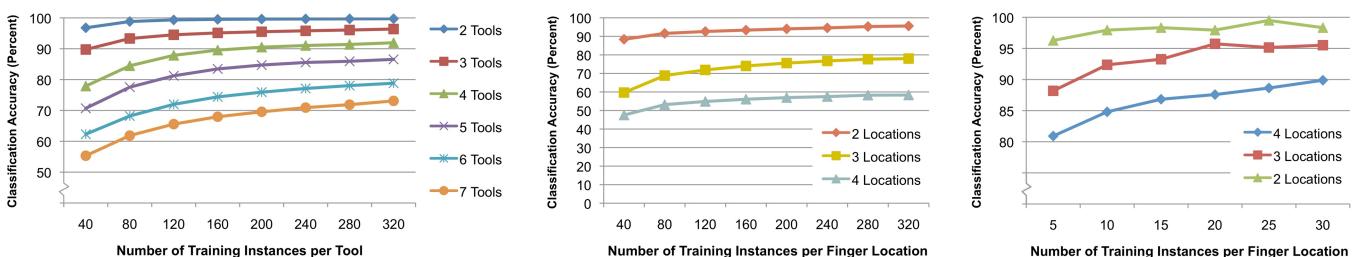


Figure 10. Left & middle: accuracy results using a general classifier. Right: accuracy results using a per-user classifier. The volume of training data was varied to see how performance was affected.

train/test combinations, and averaged the results per user (i.e. four-fold cross-validation). After computing this accuracy measure for our participants, we combined the means (see per-user plots in Figure 9).

Using all five types, the *mobile input set* achieves an accuracy of 88.3% (SE=1.7%). The tip type was the worst performing in the set, contributing 47.2% of the misclassifications. Anecdotally, the tip location appears to be the least well defined among users (compared to e.g., the knuckle), leading to higher variance. When tip is removed, accuracy jumps to 94.7% (SE=1.1%). A set containing just finger pad and the stylus achieves an average classification accuracy of 99.4% (SE=0.3%), or roughly one error in every 200 taps.

Turning to the *table finger set* - classification accuracy when using all four finger input types stands at 86.3% (SE=3.1%). As with the mobile device, tip is the worst performing; accuracy jumps to 94.0% (SE=1.4%) once tip is eliminated. Pad and nail are the best performing pair, with an average accuracy of 97.7% (SE=0.7%). Of note in the *table tool set*, finger and pen - perhaps the most common mixed-use modality (see e.g., [14,15]) - achieves 99.7% accuracy (SE=0.2%).

General Classifier

In this analysis, we evaluate system performance without per-user classifiers. In essence, we are simulating “walk up” users and estimating their performance without any training for that particular individual. We achieve this by combining data from eight users into a single aggregate training set, and then use a ninth user as the test (all combinations, i.e. nine-fold per-user holdout cross-validation).

Favorably, in the case of the *table tool set*, performance is as good as classifiers trained on a specific user (1% or less loss of accuracy at four or fewer items, well within statistical variation). This suggests the most important classification features are user-independent, and that the acoustic features of the objects are most influential.

This result was less true for the two finger-centric input sets. Given that different users have different fingers, it is not surprising per-user models would yield better results than a general classifier (per-user models also make more sense with mobile devices, as there is more opportunity and need to personalize to a single user). On average, the *mobile input set* and the *table finger set* saw their performance drop 12.1% compared to the per-user classifiers. However, this performance gap is mostly closed by the time the input sets have been reduced to just pad and nail; with 96.8% (SE=1.2%) and 97.8% (SE=1.5%) accuracy for the *mobile input set* and *table finger set* respectively (vs. 97.7%). The general classifier plots in Figure 9 illustrate the above accuracy results.

Volume of Training Data

A central question when building systems using machine learning is how much data is needed for training before

accuracy levels off. In other words, where is the point where more training data is of marginal benefit? As an initial estimation of this point in our system, we analyze data from our worst-performing table setup.

We trained our classifier on incrementally larger sets of data, starting from 40 randomly selected instances per input type to 320. Our full data set is 360 points per type, but we withheld 40 points, all from a random single user, to serve as the test set on each fold. For each size input set, we did folds for every combination of users. This procedure ensured a single user’s data was either in the test set or train set, but not both – allowing for a fair evaluation and numbers comparable to our general classifier results. Figure 10 (left and center) contains accuracy plots for each input set size in the table’s tool and finger modalities.

The *table tool set* results suggest sets of two or three plateau rapidly, with perhaps only 200 data points needed for training. However, for tool sets of four or larger, no plateau is reached. The trajectory is clearly upward even when the classifier is utilizing our full data set for training (320 points). Thus, more training data may further improve accuracy. With thousands of training instances provided by a wide variety of users, it is conceivable our full tool set of seven types could operate at useable accuracies.

A different effect is seen with the *table finger set*. General classifier accuracy improves by an average of 6.0% from a training set of 40 to 80, but only another 6.1% from 80 to 320 data points. Unlike the *table tools set* results, however, the upward trajectory is modest. Expanding from 280 to 320 training instances improves accuracy a mere quarter of a percent on average. These results paint a clear picture: a general classifier using our current table setup and software implementation would seem unlikely to ever support finger sets above two – at least not at accuracies above 90%.

These results directly support our earlier hypothesis - that per-user classifiers are most appropriate (and ultimately needed) for finger-centric input sets. If we repeat our training volume analysis using a per-user classifier, the results are notably different (Figure 10, right). Ten random instances are used as the test set, leaving at most 30 instances for training. Foremost, even with remarkably small training sets - five instances per finger type - accuracies start above 80% with four finger input types. This compared to below 50% accuracy with a general classifier using 40 training instances from random users. With 30 training instances (a very small training set by any measure), accuracies exceed 90% for *table finger sets* of 2, 3 and 4 types.

Even more encouraging, perhaps, is the fact that the *table finger set* with four input types looks to have a strong and consistent upward trajectory (Figure 10, right). It seems likely our full input set (tip, pad, knuckle and nail) could achieve 95%+ accuracies if users supplied several hundred or more training instances. This would not be unreasonable if a setup is deployed in the home or workspace.

CONCLUSION

In this paper, we have presented our acoustic-based input classification approach called *TapSense*. It relies on the unique acoustic signatures different objects create when striking a touch surface. A support vector machine is used to classify impact instances using a series of time-independent acoustic features. Software then pairs the resulting classification with an input event tracked by a variety of possible digitizing technologies. A distinguishing feature of our approach is its ability to classify different types of finger input – specifically the pad, tip, knuckle and nail – opening new interaction opportunities for touch surfaces, especially those with limited surface area. Our user study showed the technique is immediately feasible, with accuracies in excess of 95%.

ACKNOWLEDGEMENTS

This work was funded in part by a Microsoft PhD Fellowship, a Graduate Research Fellowship from the National Science Foundation, and the Center for Future Work, Heinz College, Carnegie Mellon University.

REFERENCES

1. Asano, F., Goto, M., Itou, K. and Asoh, H. Real-time sound source localization and separation system and its application to automatic speech recognition. In *Proc. Eurospeech '01*. 1013-1016.
2. Benko, H., Wilson, A., and R. Balakrishnan, Sphere: multi-touch interactions on a spherical display. In *Proc. UIST '08*. 77-86.
3. Burges, C.J. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Know. Disc.*, 2.2, June 1998, 121-167.
4. Cao, X., Wilson, A., Balakrishnan, R., Hinckley, K., and Hudson, S.E. ShapeTouch: Leveraging contact shape on interactive surfaces. In *Proc. TABLETOP '08*. 129-136.
5. Deyle, T., Palinko, S., Poole, E.S., and Starner, T. Hambone: A Bio-Acoustic Gesture Interface. In *Proc. ISWC '07*. 1-8.
6. Dietz, P.H., Harsham, B., Forlines, C., Leigh, D., Yerazunis, W., Shipman, S., Schmidt-Nielsen, B., and Ryall, K. DT controls: adding identity to physical interfaces. In *Proc. UIST '05*. 245-252.
7. Dietz, P. and Leigh, D. DiamondTouch: a multi-user touch technology. In *Proc. UIST '01*. 219-226.
8. Gutwin, C., Greenberg, S., Blum, R. and Dyck, J. Supporting Informal Collaboration in Shared Workspace Groupware. HCI Report 2005-01, U. Saskatchewan, Canada. 2005.
9. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I.H. The WEKA Data Mining Software: An Update. *SIGKDD Explor.*, 11,1. 2009.
10. Harrison, C., Tan, D., and Morris, D. Skinput: appropriating the body as an input surface. In *Proc. CHI '10*. 453-462.
11. Harrison, C. and Hudson, S.E. Scratch Input: Creating Large, Inexpensive, Unpowered and Mobile finger Input Surfaces. In *Proc. UIST '08*. 205-208.
12. Hartmann, B., Morris, M.R., Benko, H., and Wilson, A.D. Augmenting interactive tables with mice & keyboards. In *Proc. UIST '09*. 149-152.
13. Hinckley K. and Song, H. Sensor synesthesia: touch in motion, and motion in touch. In *Proc. CHI '11*. 801-810.
14. 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.
15. Hinckley, K., Yatani, K., Pahud, M., Coddington, N., Rodenhouse, J., Wilson, A., Benko, H., and Buxton, B. Manual dexterity: an exploration of simultaneous pen + touch direct input. In *CHI '10 Ext. Abst.* 2793-2802.
16. Holz, C. and Baudisch, P. The generalized perceived input point model and how to double touch accuracy by extracting fingerprints. In *Proc. CHI '10*. 581-590.
17. Kaltenbrunner, M. and Bencina, R. ReacTIVision: a computer-vision framework for table-based tangible interaction. In *Proc. TEI '07*. 69-74.
18. Matsushita, N. and Rekimoto, J. HoloWall: designing a finger, hand, body, and object sensitive wall. In *Proc. UIST '97*. 209-210.
19. Mimio: <http://www.mimio.com>
20. Olwal, A. and Wilson, A.D. SurfaceFusion: unobtrusive tracking of everyday objects in tangible user interfaces. In *Proc. GI '08*. 235-242.
21. Paradiso, J.A. and Leo, C.K. Tracking and Characterizing Knocks Atop Large Interactive Displays. *Sensor Review*, 25(2), 2005, 134-143.
22. Paradiso, J., Hsiao, K., Strickon, J., Lifton, J., and Adler, A. Sensor Systems for Interactive Surfaces. *IBM Sys. Journal*, 39(3/4), Oct '00, 892-914.
23. Patten, J., Ishii, H., Hines, J., and Pangaro, G. Sensetable: a wireless object tracking platform for tangible user interfaces. In *Proc. CHI '01*. 253-260.
24. Rekimoto, J. and Saitoh, M. Augmented surfaces: a spatially continuous work space for hybrid computing environments. In *Proc. CHI '99*. 378-385.
25. Rekimoto, J. and Sciammarella, E. ToolStone: effective use of the physical manipulation vocabularies of input devices. In *Proc. UIST '00*. 109-117.
26. Rekimoto, J. SmartSkin: an infrastructure for freehand manipulation on interactive surfaces. In *Proc. CHI '02*. 113-120.
27. Vandoren, P., Van Laerhoven, T., Claesen, L., Taelman, J., Di Fiore, F., Van Reeth, F., and Flerackers, E. Dip - it: digital infrared painting on an interactive table. In *CHI '08 Ext. Abst.* 2901-2906.
28. Wang, F. and Ren, X. Empirical evaluation for finger input properties in multi-touch interaction. In *Proc. CHI '09*. 1063-1072.