# SurfaceLink: Using Inertial and Acoustic Sensing to Enable Multi-Device Interaction on a Surface

**Mayank Goel[1,2], Brendan Lee[1], Md. Tanvir Islam Aumi[1], Shwetak Patel[1], Gaetano Borriello[1], Stacie Hibino[2], James Begole[2]**

[1]University of Washington, DUB, UbiComp Lab
Seattle, WA, USA
{mayankg, lee3, tanvir, shwetak,gaetano}@uw.edu

[2]Samsung Research America
San Jose, CA, USA
{stacie.h, james.begole}@samsung.com

## ABSTRACT

We present *SurfaceLink*, a system where users can make natural surface gestures to control association and information transfer among a set of devices placed on a mutually shared surface (*e.g.*, a table). SurfaceLink uses a combination of on-device accelerometers, vibration motors, speakers, and microphones (and, optionally, an off-device contact microphone for greater sensitivity) to sense gestures performed on the shared surface. In a controlled evaluation with 10 participants, SurfaceLink detected the presence of devices on the same surface with 97.7% accuracy, their relative arrangement with 89.4% accuracy, and a set of single- and multi-touch surface gestures with an average accuracy of 90.3%. A usability analysis showed that SurfaceLink has advantages over current multi-device interaction techniques in a number of situations.

### Author Keywords

Multi-device interaction, inertial sensing, acoustic sensing, mobile phones, surface interaction.

## INTRODUCTION

It is common to have multiple computing devices co-located in the same environment. In a number of these situations, it is useful for these devices to communicate with each other. Photo sharing is a canonical example, where a user may want to take a picture on their smartphone and share it with another smartphone, any number of computers and tablets, or even a local projector. Although the networking and data transfer aspects are largely solved using technologies such as Wi-Fi, Bluetooth, and NFC, users still need simple and natural ways to indicate the intent for creating logical connections between devices.

The radio signal strength can be used to identify proximate devices, but in today's device-dense environment, users must select which devices within wireless proximity should belong to a particular group. For instance, if the user wants to send a photo from their mobile phone to another person's computer through Bluetooth, they need to identify the computer's name from a list of nearby devices in order to pair with it.


*CHI 2014,* April 26–May 1, 2014, Toronto, Ontario, Canada.

http://dx.doi.org/10.1145/2556288.2557120

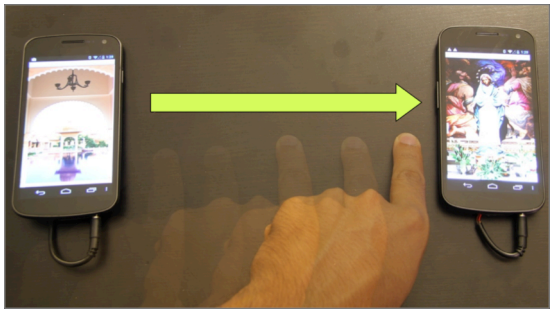

**Figure 1.** User interacting with two mobile devices on a shared table using only on-device inertial and acoustic infrastructure and inexpensive contact microphones.

A number of techniques [13,14,15,16,24,27,29] have been developed to establish a connection between devices based on "*context proximity*" [15], but these techniques do not scale well with increasing numbers of devices. Most of these techniques assume one or two devices per user, but it is increasingly common for users to have multiple devices for varying purposes. Thus, prior "pairing" techniques, which were suitable for a literal *pair* of devices, are inadequate for today's multi-device environments.

One common setting where devices and their users exchange information is around a table or other shared surface. For example, colleagues collaborate around a conference table, friends share photos around a coffee table, or a single user has their phone, tablet, laptop, *etc.* on a single desk. Hence, this commonly available flat surface provides an expedient medium for interacting across multiple devices within a bounded context (the shared surface). In order to enable multi-device interaction on a shared surface, three attributes need to be sensed: (1) the presence of multiple devices on the surface, (2) their relative placement on the surface, and (3) the gestures the user performs to interact between devices. Although past work has relied on hardware such as cameras and IR-LED arrays [3,5,17,23,32], similar capabilities can be enabled through simple on-device inertial and acoustic sensing.

We present *SurfaceLink* (Figure 1), a system that leverages inertial and acoustic sensing to enable multi-device

interaction on a shared surface. Using only the on-device sensors plus an optional 0.20 USD contact microphone plugged directly into the microphone jack, SurfaceLink detects the presence and placement of devices and gestures between them. Using sophisticated noise cancellation, the contact microphone becomes unnecessary, and SurfaceLink can be a completely software-based solution. Although not formally evaluated, we informally tested SurfaceLink with an active noise cancellation system and the performance was comparable to using external contact microphones.

SurfaceLink detects the presence of all devices on the same surface by using the surface material's ability to conduct vibrations. To induce vibrations on the surface, the user can either knock on the surface manually or the device can vibrate using its built-in vibration motors. These vibrations are sensed through accelerometers and contact microphones. Any devices that are not on the shared surface will not be able to sense these vibration patterns, thus will not consider themselves part of the same group. The surface therefore serves as a "location limited channel" [2] for situations where users want groups that only include devices directly on the shared surface.

In addition, SurfaceLink turns most surfaces into an input medium where different types of single and multi-touch gestures can be performed to interact between devices. When a finger is dragged over a surface, it produces vibrations on the surface that can be sensed as gestures using microphones. This phenomenon has been previously demonstrated by Harrison and Hudson [8]. SurfaceLink extends prior art by adding multi-touch, directionality, speed, and length to develop a richer set of gestures. SurfaceLink also detects the relative arrangement of devices on the surface by combining a user's surface gestures with acoustic stereo positioning.

We evaluated SurfaceLink in a controlled user study with 10 participants. Our findings showed that SurfaceLink detected device presence on the same surface with 97.7% accuracy and various surface gestures with an average accuracy of 90.3%. SurfaceLink also performed the relative localization of devices with 89.4% accuracy. Lastly, we qualitatively evaluated the usefulness of SurfaceLink and showed that SurfaceLink scales better to increasing numbers of devices than current multi-device interaction techniques.

The main contributions of this paper are: (1) multiple techniques to enable rich ad-hoc multi-device interactions using the surface as an input medium; empirical results from an evaluation of SurfaceLink showing that it (2) robustly detects the *presence* of multiple devices on the same surface and establishes *connections*, (3) continuously tracks *single*- and *multi-touch gestures* between multiple devices by inferring their direction, length, speed, and touch modes, and (4) accurately detect the *relative arrangement* of devices.

## RELATED WORK

### Vision-based Surface Interaction

For over 15 years researchers have emphasized that the surfaces around us can be used for interaction [28]. HoloWall [23] is one of the first systems that allows a user to interact with a glass wall. The system leverages IR-LEDs, a camera, and a projector. Similar to HoloWall, many recent techniques have used vision for making interaction with the surface more engaging. BonFire [17] uses a camera and two projectors to improve interaction between a laptop and a table's surface. Cuypers *et al.* [5] developed a technique to determine the position of a portable device on an interactive glass surface by projecting unique visual patterns and sensing the pattern received by the portable device's camera. BlueTable [32] connects multiple devices on a shared surface using vision-based handshaking. SideSight [3] uses IR-LEDs and vision for around-the-device interaction. All these systems, whether single- or multi-device, use computer vision to detect the devices and user-interaction with the surface. A vision-based system lacks portability and requires one or more cameras in the environment. These requirements make such a system unsuitable for impromptu interactions. In contrast, SurfaceLink uses lightweight inertial and acoustic sensing to detect devices and enable user-interaction. This approach allows for much more impromptu multi-device interactions.

### Detecting Contextual Presence

Apart from the vision techniques mentioned above, others have proposed techniques for detecting the presence of multiple devices that shared some context with each other. Smart-Its-Friends [15] called this concept "*context proximity*". The authors described how artifacts established connection with each other by detecting a shared phenomenon in their relative contexts. For example, two devices shared data when a user shook both of the devices simultaneously. This technique of detecting synchronous events has been explored by a number of researchers. Ideas such as bumping two devices together [24] or making contact with touch screens at the same time [27] provide a tangible way of initiating connections between two physically disconnected devices. PhoneTouch [29] detects co-occurrence of device vibration and surface touch to facilitate interaction between an interactive surface and a mobile device. Hutama *et al.* [16] used tilt correlations to facilitate interaction between multiple devices. All these techniques have a scalability problem as they either do not work for more than two devices, or assume a limit of about one user per device. Stitching [13] provides a very interesting way of connecting multiple devices together. The user draws a line with their finger from the touchscreen of the sending device to the touchscreen of the receiving device. This is a very intuitive action for connecting two devices, but it requires both devices to have a touchscreen and is less intuitive for connecting multiple devices. SurfaceLink avoids the scalability problem by leveraging

the mutually sensed vibrations of a shared surface to allow any number of devices to be easily connected.

### Device Arrangement

Detecting where the devices are kept in the interaction environment is relatively straightforward in the earlier mentioned vision-based techniques. The main disadvantages of such techniques are their relatively demanding hardware requirements. These technologies cannot be used for impromptu interactions because it is not common to have calibrated cameras set up in the environment. Dearman *et al.* [6] developed a lightweight system where a group of mobile devices used their backside camera to infer their relative orientation. Lucero *et al.* [21,22] explored how radio tracking technology can be used to determine devices' relative positions, but this approach fails when the devices are densely packed. Similar to SurfaceLink, Kortuem *et al.* [18] used ultrasonic audio signals for relative localization of devices, but they used external hardware and were limited to only detecting whether the device is *to the left, to the right, approaching,* or *moving away* from first device. SurfaceLink, in contrast, combines surface gestures and stereo acoustic sensing to infer a multi-device arrangement on a surface in both x and y directions.

### Surface Gestures using Acoustic Sensing

Prior to SurfaceLink, others have proposed techniques for enabling interactions with a surface using acoustic sensing. Paradiso and Checka did some of the early work in this area [26]. They proposed a system that tracked a user's knock on a glass wall by measuring the time delay of arrival (TDOA) of vibrations at 4 different contact piezoelectric pickups. The work by Crevoisier and Polotti [4] was one of the earlier works that explored interaction with day-to-day physical artifacts through sensing the surface vibrations. ScratchInput [8] demonstrated that when a user drags their nails on a textured surface, the resultant vibrations could be used to convert the surface into an ad-hoc interactive surface. ScratchInput just scratched the surface of possibilities in this area, and SurfaceLink builds on this work to additionally provide single- and multi-touch gestures with varying speed and length. The gestures can be tracked in real-time and because SurfaceLink performs spectral analysis on the acoustic signal, it performs well even when the signal-to-noise ratio is low. Hence, the user no longer is limited to scratching with just the fingernails. The user can interact using any part of the hand. SurfaceLink is also first to explore how these gestures can be used in a multi-device environment and presents a set of gestures and enabling technologies for effectively interacting among multiple devices.

Seniuk and Blostein [30] explored how the surface vibrations generated by pens on a known textured surface could be used for recognizing a fixed dictionary of 26 words written by the user. Murray-Smith *et al.* [25] also used such surface vibrations to develop a handheld interaction device that could be controlled by tapping, scratching, or rubbing the surface. In a similar effort, Harrison *et al.* [11] demonstrated how structured patterns of physical notches could be used as acoustic barcodes when swiped with a hard object like a fingernail or a smartphone. TapSense [10] and similar work by Lopes *et al.* [20] leveraged the contact microphone on a touch screen to detect different touch modes while a user tapped on the screen. SurfaceLink applies this same approach, and further extends the gesture set. To the best of our knowledge, SurfaceLink is the first work to demonstrate and evaluate gesture length, speed, direction, multi-touch, and application of these gestures in a multi-device environment.

## DESIGN OF *SURFACE-LINK*

SurfaceLink enables end-to-end fluid multi-device interactions. It first detects contextually proximate devices, then enables a rich set of single- and multi-touch gestures, and combines them with stereo acoustic sensing to determine the relative positions of the devices. A contextual group of devices present on the same surface is created. Then the user can interact with these devices through gestures on the surface. SurfaceLink enables a number of on-surface gestures including single-finger directional swipes, multi-finger pinch and expand, and grouping gestures. It also enables robust detection of the length and speed of these gestures, which can be used for continuous, fluid interaction between devices. In order to provide the system with a much better understanding of the arrangement of devices, SurfaceLink combines stereo positioning with user gesture data to accurately detect the relative positions of devices in a 2-dimensional space.

SurfaceLink leverages the shared surface as a communication channel between the devices that are placed on it. It uses a combination of the on-device accelerometer, vibration motor, speakers, microphones, and off-device contact microphones for establishing connections and interactions between the devices.

### Detecting Devices on the Same Surface

SurfaceLink uses the fact that hard, flat surfaces conduct vibrations well. These vibrations can either be user-induced by knocking on the surface, or device-induced using a vibration motor. These vibrations travel through the dense material of the surface and are sensed by the device sensors. Harrison *et al.* [9] and Kunze *et al.* [19] demonstrated techniques to detect the nature of the surface on which a device is kept. In contrast, SurfaceLink detects if the devices are on the same surface, irrespective of the material. For example, SurfaceLink can distinguish between whether devices A and B are kept on the *same* or *different* tables, whereas [9] and [19] detected if a device was kept on a metal or wooden table.

#### *Using User-Induced Vibrations*

When a user knocks on a surface, it generates strong vibrations that travel through the dense material of the

surface. These vibrations can be sensed by the on-device accelerometers.

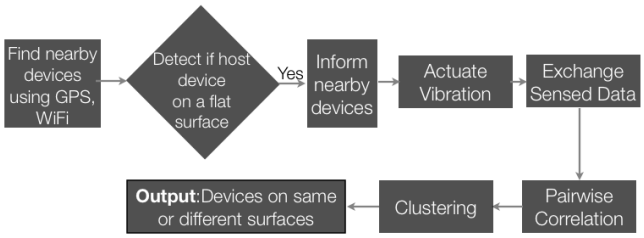


**Figure 2.** Block diagram of major components of SurfaceLink's detection of devices on the same surface. When sensing user-induced vibrations, inertial sensor data is used, and when sensing device-induced vibrations, the contact microphone data is used.

Initially, when a user wants to connect to other devices on the same surface, the user's device acts as a host device and uses GPS and Wi-Fi information to find the nearby devices. Once the device has a list of nearby devices, it instructs them to start sampling their accelerometers (Figure 2). It then instructs the user to knock on the surface in any user-defined pattern and samples its accelerometer at 100 Hz. The devices share their observed accelerometer data. A device might experience a number of vibrations in a busy environment but most of these vibrations are relatively low frequency when compared to those generated by the user's knocks. Hence, the accelerometer data of all devices is passed through a high-pass filter. Then a pairwise cross correlation is performed between these data samples. The devices are then clustered into two sets (*on same surface*, and *not*) on the basis of their cross-correlations. We use the Single Linkage algorithm that uses the smallest distance between objects in different groups to cluster them.

*Using Device-Induced Vibrations*

In a number of cases, interface designers might not want to require a user to generate the stimulus by knocking on a surface to pair devices. In such cases, we can use the built-in vibration motor to generate the stimulus. The inertial sensors on most mobile devices cannot detect device-induced vibrations because they do not have the sensitivity and resolution required. Hence we use a microphone to pick up these subtle vibrations. These vibrations are usually coupled to the air and can be heard by a microphone on some other surface as well. In order to pick up these subtle surface vibrations, it requires either noise cancellation using two microphones on the device (*i.e.*, one touching the surface and one in the opposite direction), or a low-cost external contact microphone with a resonant frequency of 6.3 kHz. The device uses on-device inertial sensors to automatically detect if it is placed on a stable, flat surface [7]. In the same way as mentioned above for the user-induced case, the device obtains a list of nearby devices. These clients are then informed to sample their microphones at 44.1 kHz. When a device vibrates on a hard, flat surface it periodically hits the surface, thereby generating a low frequency sound. The system performs noise cancellation by subtracting the audio from the microphone on the top from the microphone touching the surface. When using a contact microphone, the microphone couples strongly to the surface and only records the surface vibrations, hence no filtering is needed. The devices then share their data and a pairwise cross-correlation and clustering is done as described previously.

In some cases, where there are a number of devices on a relatively large table, the host device can be far from some potential client devices. This often leads to situations where the devices are unable to reliably experience the induced vibrations. In such a situation, the system assumes that these far-away devices would still be relatively near to some of the other devices and employs a "vibration daisy chaining" technique where identified devices induce new vibrations to extend the reach of the device network.

**Detecting Surface Gestures**

SurfaceLink uses the concept that a finger/hand dragged over most hard surfaces produces measurable vibrations. These vibrations can be detected using the on-device microphone or an off-device contact microphone.

*Gesture Class*

We observe that the vibrations generated due to dragging of finger on a surface have different characteristics in different directions. One of the differences is that in many cases they are louder when near the device and softer when far. The difference in swipe direction also results in distinctly different patterns in the frequency domain. As a user's finger comes near a device, there is a decrease in the resonant frequency (Figure 3A), and the opposite effect is observed when the finger moves away from the device (Figure 3B).

A swipe gesture may be performed between two devices, but in a number of situations, more than two devices might be present on the surface. The devices not involved in the gesture would also hear the vibrations on their

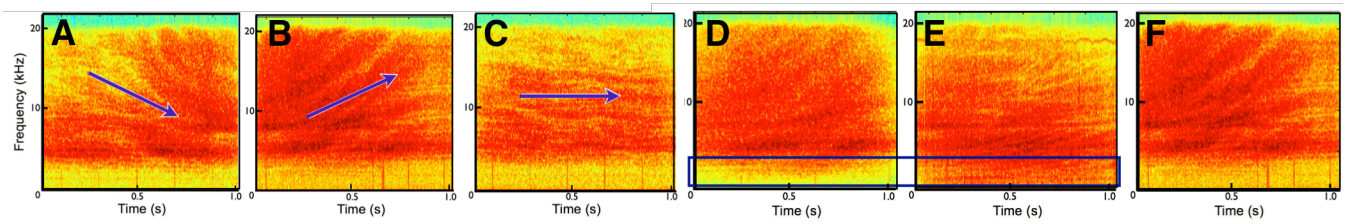


**Figure 3 (A)** Swipe towards the device. **(B)** Swipe away the device. **(C)** Vibrations experienced by the non-participating device. **(D)** Gesture performed with fingertip. **(E)** Gesture performed with fingernail. **(F)** Gesture performed with fist

microphones. Hence there is a need to disambiguate between participating and non-participating devices. In the case of a unidirectional swipe gesture between two devices, a non-participating third device can experience one of three possibilities. It will observe vibrations similar to the two participating devices (Figure 3A or 3B) if it is right next to one of the participating devices. In this case, the overall spectral energy of the vibrations will be much higher for the participating devices. If the non-participating device is not near to either of the participating devices, it might observe similar energy levels, but now the swipe is neither going towards nor away from it, and hence it will observe vibrations similar to those shown in Figure 3C (*i.e.*, no change in resonant frequency over time).

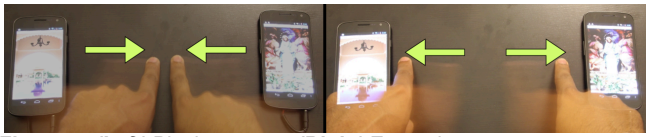

**Figure 4.** ***(Left)*** Pinch gesture, ***(Right)*** Expand gesture

SurfaceLink also supports two multi-finger gestures: *pinch* and *expand*. In the pinch gesture (Figure 4, *Left*), the user starts the swipe near each device and draws them together toward the middle. This gesture can be used to connect two devices together. In the case of such a gesture, both devices experience a swipe away from them (Figure 3B). The complimentary gesture, *expand* (Figure 4, *Right*) generates the equivalent of swiping toward both devices (Figure 3A) and can be used for disconnecting two devices.

The speed of a gesture on a surface is directly correlated to the amplitude of the resonant frequency. In addition, the duration of all gestures are bounded since they are mostly performed between two devices. Hence, if a user does a fast swipe, the gesture accelerates and decelerates quickly. Therefore, in addition to the higher amplitude, the slope of increase and decrease in amplitude is also steeper. The combination of these two phenomena can be used to differentiate between different gesture speeds. In order to make it easy for the user to discern different speed levels, SurfaceLink supports only two speeds: slow and fast.

### *Gesture Length*

Considering that SurfaceLink can estimate the speed of a gesture and the length of a gesture is bounded by the distance between two devices, we can also detect the length of a gesture. A smaller gesture can be compared to a longer gesture of same speed as both will have the same amplitude and slope but different durations. SurfaceLink supports three gesture lengths: full, half, and quarter length.

### *Touch Modes*

SurfaceLink uses the fact that different finger locations generate different vibrations when dragged across a surface. SurfaceLink employs the techniques demonstrated by Harrison *et al.* [10] to support three touch modes: fingertip, fingernail, and fist. The nail, being harder than the fingertip, produces a very different frequency profile (Figure 3D and 3E). The differentiation between fingertip and fist (Figure 3F) is slightly different. The fingertip and fist are made up of the same material, so the frequency response for the two modes is not different. However, because the area of contact is much larger in the case of the fist, it creates a higher amplitude sound.

### *Gesture Shape*

ScratchInput [8] demonstrated that analysis of peak counts and amplitude variation could be used to infer a fixed dictionary of gesture shapes, such as a line, circle, triangle, and square. We employ similar techniques, albeit using the spectral information and performing pattern matching.

### *Implementation for Detecting the Surface Gestures*

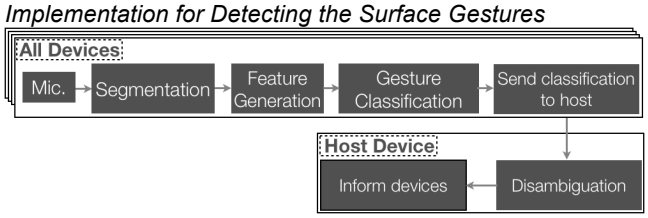


**Figure 5.** Block diagram of major components of SurfaceLink's gesture detection. Pattern matching results from all devices are sent to the host device for disambiguation.

Every device on the surface records audio data at 44.1 kHz. The data is then segmented into individual gestures. This segmentation is done by thresholding the audio data between 5 kHz and 15 kHz. We did not formally evaluate the segmentation but in our informal evaluation with 3 participants, there was virtually no segmentation error because the contact microphone couples very well to the surface and is immune to ambient noise. The segmented audio data is then used to generate machine learning features before going through gesture classification (Figure 5). Each device individually infers the gesture, and then shares the classification result with the host device. SurfaceLink classifies each gesture into four different gesture properties:

1. *Gesture Class*: Away, Towards, Non-Participating, Pinch, Expand, or Fast Swipe.
2. *Gesture Length*: Quarter, Half, or Full.
3. *Touch Mode*: Fingertip, Fingernail, or Fist.
4. *Gesture Shape*: Line, Polygon, Triangle, Circle, Semi-Circle.

In order to generate the features, we produce the magnitude spectrogram of the audio signal using a 1024-point FFT with a 100 sample Hamming window. In order to reduce the number of features, the |FFT| data over the duration of the gesture was down-sampled into 6 frequency bins and 10 time frames using median filtering. This forms a 6x10 matrix representing the first 60 features used for classification. Next, in order to capture the temporal variation for each gesture, we band-pass filtered (5 kHz to 15 kHz) the temporal data and then down-sampled into 10 time windows. Thereby providing 10 new features. The last

2 features are the total energy in the first and second half of the gesture. These 72 features are used to inform a kNN classifier ($k$ = 2) on each device. We have separate kNN classifiers for each gesture property (*Gesture Class*, *Gesture Length*, *Touch Mode*, and *Gesture Shape*). It helps in keeping the number of training samples to a minimum. A detailed analysis of the effect of training size on the performance of the system is provided in the Results section.

Once each device makes a decision for each gesture, it sends the classification result, classification confidence, and the total observed energy for the gesture to the host device. The inverse of normalized distance between the observation and nearest neighbor is used as the classification confidence. For example, if the normalized kNN distance between the observation and nearest neighbor on device A is 10, and on device B is 20, then their respective classification confidences are 0.1 and 0.05. The host device then checks if all the devices agree on the classification. In case there is ambiguity, the host device checks the confidence metric for each device and selects the decision of the device with the highest confidence. In cases where multiple devices predict that they were participants in a gesture, the device with higher total observed energy is selected.

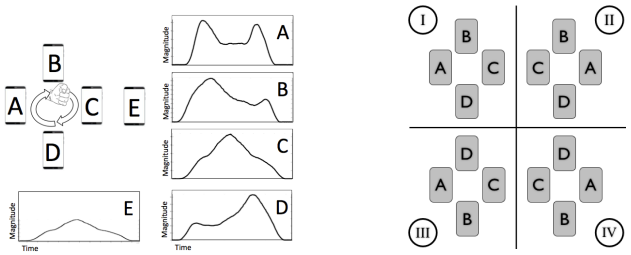


**Figure 6.** ***(Left)*** The timestamp of the audio peak for each device informs the order of devices. ***(Right)*** Four possible arrangements of devices.

### Detecting Device Arrangement

SurfaceLink also infers device arrangement that can be used to enable placement-aware interactions. It leverages the knowledge of the shape of the gesture to infer the arrangement of devices. For example, in Figure 6*, Left*, the circular gesture began and ended at Device A. As a result, different devices on the table observed a peak in the vibrations at different times and the non-participating device E, as expected, observed significantly lower vibrations. From these signals, the ordering and placement of the devices can therefore be inferred. To do so, the system places the objects on the edge of the drawn shape (*e.g.*, a circle as shown in Figure 6*, Left*). To determine the distances between the devices, the system uses the time intervals between the peaks in the audio received by each device. For example, if B receives a peak closer in time to A than C, then the system infers that B is physically closer to A on the edge of the circle. Also, if device E observes an attenuated version of the peak observed by C, but with the peak at the same time, then the system infers that C and E are at the same angle on the circle, but at different radii (*i.e.*, E is farther from the gesture).

This inference system has one limitation: the system would come back with the order of devices as A-B-C-D. The actual arrangement can be any of the four possible mirror images (Figure 6, *Right*). In order to reduce the ambiguity, SurfaceLink limits the user to perform clockwise gestures when devices are in a 2-dimensional arrangement. By adding this constraint, the system can eliminate arrangements shown in Figure 6, *Right* (III) and (IV). In order to reduce the remaining ambiguity, SurfaceLink employs an acoustic stereo positioning technique to calculate the right/left orientation of one device with respect to another. For example in Figure 6, *Left*, if Device A infers that Device C is to its right, then the only possible arrangement will be Figure 6 *Right* (I). In cases where devices are arranged in a single line (1-dimensional arrangement), there are only two mirror images. This ambiguity can be easily resolved by stereo positioning, hence there is no limitation on the user for gesture direction.

In order to determine whether a device is on the left or right side of another device, it emits two ultrasonic tones of different frequencies from both left and right speakers. Ultrasonic tones have been used for data transfer [1] and proximity detection [31] as well; however, these have never been used for relative arrangement detection. We use an 18 kHz sine wave for the left speaker and 18.5 kHz for the right speaker. If the client device is on the right side, then the observed amplitude of the 18.5 kHz tone will be higher than that of the 18 kHz tone. The frequency response for most commodity speakers is not linear at such high frequencies, hence there might be a constant offset between the magnitudes of the two frequencies. Such an offset can be easily accounted for by calibrating using the microphone of the emitting device.

In the ideal case, the amplitude of the two tones can be compared instantaneously, but we observed that the difference in amplitude could vary over time. This leads to some noisy results, hence SurfaceLink tracks the amplitude difference for 3 seconds and passes it through a low pass filter before deciding on the orientation. After this decision, the system does not re-compute the orientation until the user moves the devices.

Such placement-aware systems have been studied earlier as well. Hinckley *et al.* [13] demonstrated the utility of such a system using touchscreens and the slope of a user's swipe while connecting devices. This knowledge of relative placement of devices can be used to facilitate more natural touch-screen interactions between devices. For example, if a user swipes from left to right on the touchscreen of device A, the system detects whether device B is on right or left. If B is on left, then it means that the gesture went from B to A and so on.

## EVALUATION & RESULTS

The performance of SurfaceLink was evaluated in a controlled user study. Ten participants (6 males, 4 females) ranging in age from 21 to 32 ($\mu$=26.9, $\sigma$=3.6) were recruited. All participants had more than 10 years of experience with computers and self-rated as intermediate to expert computer and smartphone users. The evaluation was performed using a custom developed application deployed on 4 Samsung Galaxy Nexus smartphones.

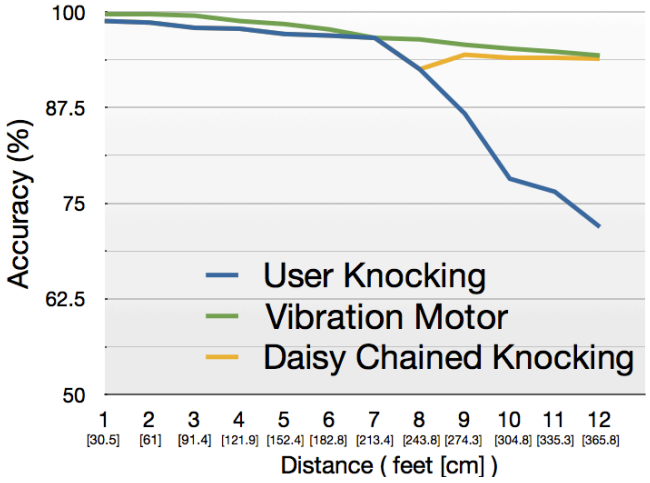


**Figure 7.** Performance of detecting devices on the same surface. User knocking performance drops significantly over larger distances, but this can be nullified using daisy chaining.

### Detecting Devices on the Same Surface

The device detection system can operate using user-induced or device-induced vibrations. In the evaluation, the order in which these two vibration sources were used was randomized. The experiments were performed on three different surfaces: wood, laminate, and metal. When using user-induced vibrations, the participants were asked to put the phone onto one of the three tables and then they were instructed to knock on the table when the application prompted them. Each table had a host device on it to detect co-presence. When using device-induced vibrations, the devices were simply placed on the table and did not require any user intervention. All the data was recorded in the application. In post processing, we checked if the system was able to classify the user's phone into respective clusters for each of the three tables.

Figure 7 shows the performance of SurfaceLink over varying distances between the client and host device. The accuracy of the system expectedly decreases as the inter-device distance increases. When using user-induced vibrations, the accuracy drops significantly around 8 feet, whereas in the device-induced vibration case, the accuracy remains above 94% even at distances more than 12 feet. In order to alleviate the accuracy drop after 8 feet while using user-induced vibrations, we used vibration daisy chaining as described previously. Daisy chaining improves accuracy from 71.9% to 93.9% at 12 feet. Overall, SurfaceLink detects devices on the same surface with 97.7% when the devices are up to 8 feet apart. The performance difference was not significant for any of the materials as the magnitude of vibrations was high in all cases.

### Detecting Surface Gestures

The data collection application for gesture classification recorded the vibrations observed by the contact microphone. We did not formally evaluate the performance with the on-device microphone because the noise cancellation capabilities of modern mobile devices are part of the hardware firmware and are not accessible to the developer. Four phones were placed on a surface in an arbitrary square with approximately 0.5 m between them.

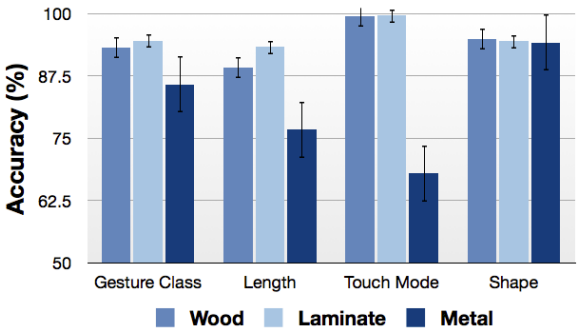


**Figure 8.** Performance of gestures on different devices (5-fold cross validation). Wood and laminate performed the best. Errorbars are standard errors.

All possible combinations of the gestures were not evaluated because it would have led to 180 different input gestures and impractically long evaluation sessions. We tested all combinations of *Gesture Shape* and *Touch Modes*. In case of *Gesture Class* and *Gesture Length,* the gestures were only performed with the fingertip. Thereby creating a vocabulary of 5*3+6+3=24 different gestures. Each gesture was repeated 10 times, and the order was randomized. Again, in order to evaluate the performance of SurfaceLink on a variety of surfaces, the data was collected on three different surfaces: wood, laminate, and metal.

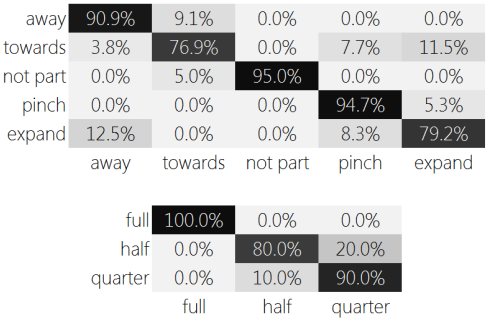


**Figure 9.** Confusion matrices for different (*Top*) Gesture Class and *(Bottom)* Gesture Length. (Note: Non-participating devices are abbreviated as *not-part*)

Figure 8 shows the performance of SurfaceLink in classifying different surface gestures using a 5-fold cross validation. The performances of wood and laminate were

similar to each other (94.2% and 94.9%, respectively) and were both better than metal (81.2%). This is expected as the metal surface is not a textured surface and was used to only to test the lower bounds of SurfaceLink's performance. The mean accuracy across gesture class, length, touch mode, and shape detection was 90.3%.

The confusion matrices for *Gesture Class* and *Gesture Length* are shown in the Figure 9. It can be seen that there is no pair of gesture classes that often confused. The disambiguation of participating and non-participating devices is also very robust.

In the evaluation, each user was asked to perform each gesture 10 times, and then we used a 5-fold cross-validation in order to obtain results. In a real world setting the user would need to perform a training phase before using the system. We therefore did an analysis to see how much training data is needed to obtain reasonable performance, as shown in Figure 10. The *x*-axis shows the percentage split in training and test data. Larger values on *x*-axis signify more training data. It is clear from the figure that in the case of *Gesture Class*, *Touch Mode*, and *Shape*, SurfaceLink does not require much training data, as just a few examples would suffice. The accuracy is lowest for *Gesture Length,* but even then only 5 examples are needed to increase the accuracy to 85%.

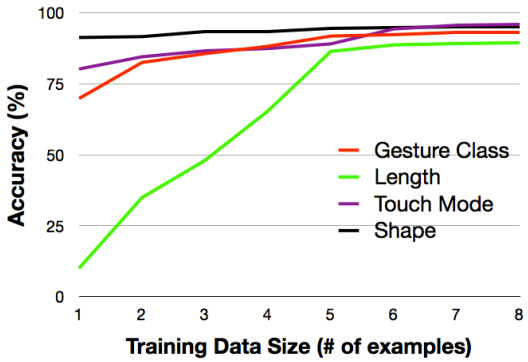


**Figure 10.** Improvement in average accuracy of gesture detection on the three surfaces with increasing size of training data.

### Detecting Device Arrangement

This evaluation was divided into two parts. The first part was to evaluate the performance of acoustic stereo positioning. A mobile device was kept at different relative locations to the host device (Motorola Xoom). The aim of this evaluation was to check the bounds of distance between the two devices where SurfaceLink is able to determine effectively determine the arrangement.

Figure 11 shows the performance of acoustic stereo positioning. The average accuracy over inter-device distances between 4 and 36 inches is 89.4%. The accuracy drops to approximately 66% when the devices are 36 inches apart. This is because the distance between the two speakers on the host device of less than 12 inches. As the inter-device distance increases, the difference in the observed tone amplitude will become negligible.

The acoustic stereo positioning detects which devices are to the left and right of each other. But in order to get the accurate arrangement of all the devices on a surface, a combination of stereo positioning and user gesture is required. The participants were asked to draw a gesture between all the devices on the table with their fist. They were presented with 10 unique device arrangements including devices in a straight line, square, triangle, *etc*. The inter-device distance was limited between 4 to 24 inches as larger distances meant devices were too far away for a single user. The participants were not informed which gesture shape to perform. They selected the gesture shape on their own from a set of triangle, square, circle, line, and arc. They were informed to select the shape keeping in mind that their gesture should try to go near all the devices. For example, when the devices were kept in a triangle arrangement, a circular or triangular gesture would be appropriate.

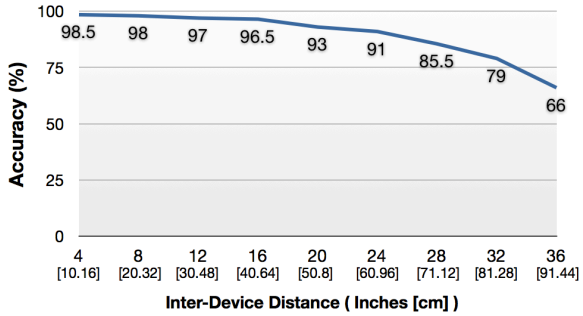


**Figure 11.** Performance of acoustic stereo positioning.

The system tried to determine the order and whether the devices were in a linear, circular, triangular, or square arrangement. The system accurately detected the device arrangement 88% of the times.

### Qualitative System Evaluation

We developed a multi-device photo-sharing application (Figure 1) in order to qualitatively evaluate the performance of interaction techniques introduced by SurfaceLink. In this application, the system first detects the devices that are present on the same surface. Then it allows the users to interact with devices through surface gestures.

The performance of this system is compared with Bump[1] and Cooperative-Stitching [14]. The purpose of this evaluation was to test how SurfaceLink performs with increasing number of users and devices. We recruited 10 users to evaluate a system in which they transferred pictures between four devices using all the three techniques (*i.e.*, Cooperative-Stitching, Bump, SurfaceLink). The participants were given a randomly generated list of 15 photo transfer tasks. An example of a task will be "Transfer

[1] Bump: http://www.bu.mp

photo from A to B". In 5 of the 15 tasks for each participant, there were multiple receiving devices. For example, "Transfer photo from A to B and C". These tasks were added to mimic a scenario where users might want to share files with multiple devices. In Cooperative-Stitching the users collaboratively drew a line with their finger from sending device to receiving device. For SurfaceLink the users used the swipe gesture to share photos between devices and drew circles with fist to create a subgroup of devices (in the case of multiple receiving devices).

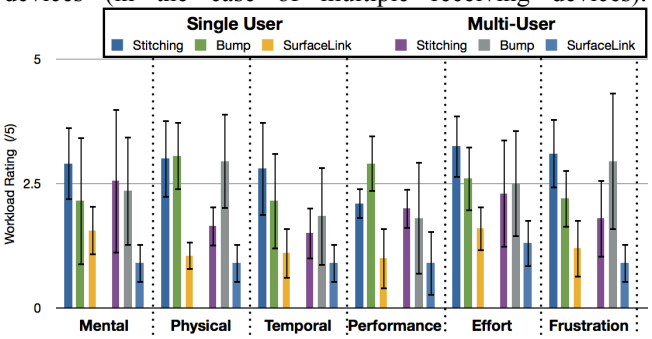


**Figure 12.** Perceived workload ratings show SurfaceLink results in significantly lower workload in the single-user scenario. Lower ratings are better.

Two experiments were conducted to test the three techniques in single- and multi-user environments. In the first experiment, a single user interacted with 4 devices. In the second experiment, a set of four users controlled one device each. We overlapped our users and each user was part of 2 separate sets. Therefore we had 5 sets of four users, each.

The participants were taught the functionality of each app and were given time to get familiarized with the interface. Once they were comfortable they were presented with the three techniques in a random order. After completion of tasks with each technique, we asked participants to make Likert-scale ratings (Figure 12) based on NASA TLX perceived workload index [12]. We also recorded the time taken by each participant to complete the tasks with each technique.

A repeated measures ANOVA revealed that, regardless of number of devices per user, participants spent significantly less time in completing the tasks while using SurfaceLink as compared to the other two techniques ($F_{2,18}$=10.94, $p$ < .05). A Friedman test showed that in the single user scenario, SurfaceLink resulted in significantly less workload in all dimensions as compared to Bump and Cooperative-Stitching. In the case of multiple users, SurfaceLink resulted in significantly less mental, temporal, and physical workload, and frustration. All Wilcoxon pairwise comparisons were protected against Type I error using a Bonferroni adjustment.

When asked which of the techniques they preferred, and all participants preferred using SurfaceLink over the other two techniques. P4 said, "The surface gestures feel very natural." P2 said, "The fact that I do not need to interact with all of the devices' touchscreens to share files can be very useful." However, 6 out of 10 participants preferred Bump or Cooperative-Stitching over SurfaceLink if they just had to send one single batch of files to a single paired device.

## DISCUSSION AND LIMITATIONS

Our results show that SurfaceLink performs better than Bump and Cooperative-Stitching when there is more than one device per user. There is no established baseline for the interaction techniques for multiple devices. We selected Bump and Cooperative-Stitching because they are very lightweight and do not require any hardware adjustment to the devices. Moreover they are good representatives of "co-occurrence" based multi-device systems. We do not aim to hypothesize that SurfaceLink is better than *all* the multi-device interaction approaches. We solely wanted to test how SurfaceLink performs with increasing number of users and devices in comparison to some of the proven techniques. The fact that all the participants preferred SurfaceLink suggests that it is an attractive interaction system as the number of devices as well as our interaction with these devices increases.

We deployed the system on a work desk for a week with almost no false positives (4 in whole week). SurfaceLink is able to use the duration of gesture to easily remove false positives. Additionally, considering it is aware of device arrangement and also classifies devices into "non-participating devices" category, accidental touches are almost always classified as "non-participating devices" for all devices.

Extended use of the microphones and accelerometer can have significant power implications. The devices do not need to be in permanent listening mode. The devices can do opportunistic sensing and would only listen when they infer they are on a flat surface [7].

Surface gestures are also useful in "around-the-device" interactions. Such interactions are becoming very popular. SideSight [3] is one example of such a system. The rich set of gestures introduced by SurfaceLink is demonstrated to be useful in a multi-device interaction scenario. But these gestures can be very useful for a single device interaction as well, and aid in "around-the-device" interaction. This paper uses the photo sharing as an example scenario but the demonstrated gesture language can be used for a variety of purposes in both single- and multi-device environments.

## CONCLUSION

It is common to have multiple computing devices in the same environment. It is also becoming increasingly common for a single user to have multiple devices. Current multi-device interaction technologies do not scale well to increasing numbers of devices per user. In this paper, we presented *SurfaceLink*, a system that uses the shared surface between devices as a bounded communication medium. It provides an end-to-end solution for impromptu, scalable

interaction between devices using on-device accelerometer, vibration motor, speakers as well as an off-device contact microphone. SurfaceLink detects devices on the same surface (*Acc.*=97.7%), enables a variety of continuous multi-touch gestures (*Acc.*=90.3%), and infers relative device arrangement (*Acc.*=89.4%). Our evaluation also demonstrates that SurfaceLink scales well to increasing numbers of devices and users and can be an attractive interaction system as the number of devices or our interaction with these devices increases.

**ACKNOWLEDGEMENTS**

We thank Lilian de Greef for her help in prototyping.

**REFERENCES**

1. Arentz, W.A. and Bandara, U. Near ultrasonic directional data transfer for modern smartphones. *Proc. UbiComp'11*.
2. Balfanz, D., Smetters, D.K., Stewart, P., and Wong, H.C. Talking To Strangers : Authentication in Ad-Hoc Wireless Networks.
3. Butler, A., Izadi, S., and Hodges, S. SideSight : Multi-"touch" interaction around small devices. *Proc. UIST'08*.
4. Crevoisier, A. and Polotti, P. Tangible acoustic Interfaces and their applications for the design of new musical instruments. *Proc. Nime'05*.
5. Cuypers, T., Francken, Y., Vanaken, C., Reeth, F.V., and Bekaert, P. Smartphone localization on interactive surfaces using the built-in camera. *Proc. Procams'09*.
6. Dearman, D., Guy, R.T., and Truong, K.N. Determining the orientation of proximate mobile devices using their back facing camera. *Proc. CHI'12*.
7. Goel, M., Wobbrock, J.O., and Patel, S.N. GripSense: Using built-in sensors to detect hand posture and pressure on commodity mobile phones. *Proc. UIST'12*.
8. Harrison, C. and Hudson, S.E. Scratch Input : Creating Large, Inexpensive, Unpowered and Mobile Finger Input Surfaces. *Proc. UIST'08*.
9. Harrison, C. and Hudson, S.E. Lightweight Material Detection for Placement-Aware Mobile Computing. *Proc. UIST'08*.
10. Harrison, C., Schwarz, J., and Hudson, S.E. TapSense : Enhancing finger interaction on touch surfaces. *Proc. UIST'11*.
11. Harrison, C., Xiao, R., and Hudson, S.E. Acoustic Barcodes : Passive , Durable and Inexpensive Notched Identification Tags. *Proc. UIST'12*.
12. Hart, S.G. Nasa-Task Load Index (NASA-TLX); 20 Years Later. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting 50*, 9 (2006).
13. Hinckley, K., Ramos, G., Guimbretiere, F., Baudisch, P., and Smith, M. Stitching : Pen Gestures that Span Multiple Displays. *Proc. AVI'04*.
14. Hinckley, K. and Ramos, G. Cooperative Stitching : Spontaneous Wireless Connections for Small Co-Located Groups (TECHNOTE).
15. Holmquist, L.E., Mattern, F., Schiele, B., Alahuhta, P., Beigl, M., and Gellersen, H.-w. Smart-Its Friends : A Technique for Users to Easily Establish Connections between Smart Artefacts. *Proc. UbiComp'01*.
16. Hutama, W., Song, P., Fu, C.-W., and Goh, W.B. Distinguishing multiple smart-phone interactions on a multi-touch wall display using tilt correlation. *Proc. CHI'11*.
17. Kane, S.K., Avrahami, D., Wobbrock, J.O., et al. Bonfire : A Nomadic System for Hybrid Laptop-Tabletop Interaction. *Proc. UIST'09*.
18. Kortuem, G., Kray, C., and Gellersen, H. Sensing and visualizing spatial relations of mobile devices. *Proc. UIST'05*.
19. Kunze, K. and Lukowicz, P. Symbolic Object Localization Through Active Sampling of Acceleration and Sound Signatures. *Proc. UbiComp'07*.
20. Lopes, P., Jota, R., and Jorge, J.A. Augmenting touch interaction through acoustic sensing. *Proc. ITS'11*.
21. Lucero, A., Holopainen, J., and Jokela, T. Pass-Them-Around : Collaborative Use of Mobile Phones for Photo Sharing. *Proc. CHI'11*.
22. Lucero, A., Jokela, T., Palin, A., Aaltonen, V., and Nikara, J. EasyGroups : Binding Mobile Devices for Collaborative Interactions. *Proc. CHI'12 (WIP)*.
23. Matsushita, N. and Rekimoto, J. HoloWall: Designing a Finger; Hand, Body, and Object Sensitive Wall. *Proc. UIST'97*.
24. Mayrhofer, R. and Gellersen, H. Shake Well Before Use : Authentication Based on Accelerometer Data. *Proc. Pervasive'07*.
25. Murray-Smith, R., Williamson, J., Hughes, S., and Quaade, T. Stane : Synthesized Surfaces for Tactile Input. *Proc. CHI'08*.
26. Paradiso, J.A., Leo, C.K., Checka, N., and Hsiao, K. Passive acoustic sensing for tracking knocks atop large interactive displays. *Proc. IEEE Sensors 2002*.
27. Ramos, G., Hinckley, K., Wilson, A., and Sarin, R. Synchronous Gestures in Multi-Display Environments. *Human-Computer Interaction, Special Issue: Ubiquitous Multi-Display Environments 24*, 1-2 (2009).
28. Rekimoto, J. and Saitoh, M. Augmented Surfaces : A Spatially Continuous Work Space for Hybrid Computing Environments. *Proc. CHI'99*.
29. Schmidt, D., Chehimi, F., Rukzio, E., and Gellersen, H. PhoneTouch : A Technique for Direct Phone Interaction on Surfaces. *Proc. UIST'10*.
30. Seniuk, A. and Blostein, D. Pen Acoustic Emissions for Text and Gesture Recognition. *10th Int'l Conference on Document Analysis and Recognition* (2009).
31. Thiel, B., Kloch, K., and Lukowicz, P. Sound-based proximity detection with mobile phones. *Proc. PhoneSense'12*.
32. Wilson, A.D. and Sarin, R. BlueTable : Connecting Wireless Mobile Devices on Interactive Surfaces Using Vision-Based Handshaking. *Proc. GI'07*.