ForcePhone: Software Lets Smartphones Sense Touch Force

Yu-Chih Tung and Kang G. Shin, The University of Michigan

As smartphones become an essential part of our daily lives, human–phone interactions have become the norm. To enhance the input capability on the severely limited space of a phone's screen, *touch force* was introduced to expand user–phone interactions. It has been more than a year since the launch of Apple 3D Touch—the commercialized force-sensing technology based on augmenting proprietary touch force sensors (www.apple.com/iphone-6s/3d-touch)—yet the percentage of mobile apps using this new feature is less than 0.01 percent.[1]

One of the main reasons why app developers are reluctant to adopt this advanced user interface is the small number of devices equipped with it. This function is absent from not only all Android devices but also the latest lower-end iPhone model because of the additional cost.[2] Many solutions require additional specialized hardware, but we propose a pure software-based solution, called *ForcePhone*, that introduces a new force-sensing technology at little to no additional cost.

FORCEPHONE

ForcePhone isn't the first software-based solution. A force-sensitive interface can be implemented by checking the flashlight source blocked by a human hand,[3] monitoring the reduction of sound volume by covering the microphone reception hole,[4] or estimating the damped motor vibration with accelerometers.[5] However, systems based only on built-in sensors usually impose unnatural or inconvenient usage restrictions, because it's challenging to recognize those interactions without additional sensors. For example, users must touch the microphone reception hole, block the camera flashlight source, or tolerate the motor-driven vibrations for sensing a touch interaction, thus limiting the usability of this additional sensing.

Structure-Borne Sound Propagation

Unlike these prior solutions, ForcePhone provides a force-sensitive input interface to the touchscreen and also to the body of commodity phones. It estimates the user-applied force by using the *structure-borne sound propagation*—that is, the sound transmitted through subtle vibrations of the device body. Sound is a mechanical wave broadcasted by compressions and rarefactions. The most common material for sound to propagate is the air, which is known as *airborne propagation*, but when the sound is generated and received by the same device, its body becomes another pathway for the sound to travel.

In most designs, such as headphones or pipe-work, this type of propagation is considered mechanical noise, but ForcePhone uses it in a novel way to estimate the force applied to commodity phones. As Figure 1 shows, when the phone is left free to vibrate (the user isn't touching or squeezing the phone), the sound sent from the phone's speakers can easily travel through its body to its microphone. However, when force is applied to the phone, it restricts the phone body's vibration with the sound, thus degrading the sound traveling through this structure-borne pathway. ForcePhone estimates the amount of force applied to the phone by monitoring the degree of this degradation.

Validating the Theory

To validate the relationship of sound vibrations and touch force, we used the Polytec OFV-303 laser vibrometer, measuring the nm-level vibration of a phone and capturing the change of structure-borne propagation caused by touching the phone with a hand. A thumb applied force at the middle of an Apple iPhone 6s, and we installed an external force sensor on the phone to acquire the ground truth. Figure 2 shows our experimental settings and measurement results.

As shown in Figure 2b, the vibration amplitude decreases approximately 10 percent when a 1 kg force is applied to the phone, and almost 25 percent of the vibration amplitude is damped when more force is applied. The most important property observed in this experiment is the high correlation between the applied force and the decreased vibration amplitude. Based on this property, ForcePhone enables useful force-sensitive applications.

Note this experiment was only used to validate our hypothesis that the sound played by the phone speaker would propagate via a subtle vibration, which would change with the applied touch force. In a real system, we used the phone's microphone to sense this structure-borne propagation, because a laser vibrometer isn't installed in commodity phones.

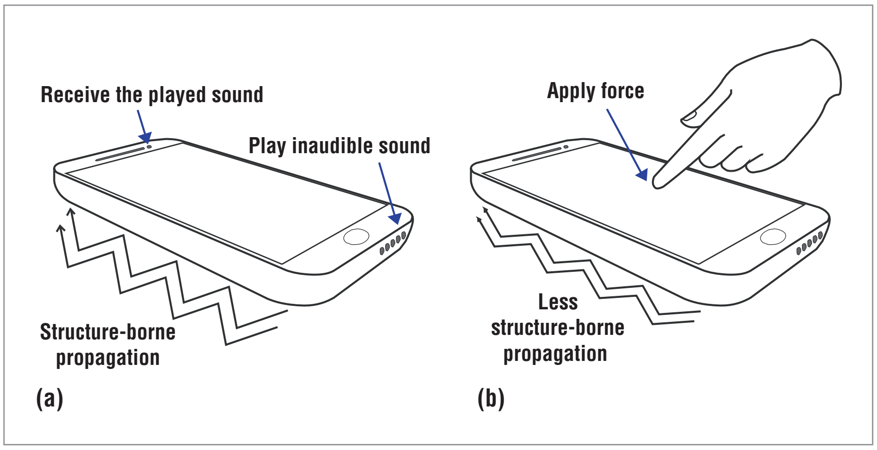

Figure 1. Structure-borne propagation and the applied force. (a) When no force is applied to the phone, the frame and internal components of the phone can vibrate freely, and hence the played inaudible sound can easily propagate through the phone's body. (b) When force is applied, it restricts the phone body's vibration with the sound, thus degrading the sound traveling through this structure-borne pathway.

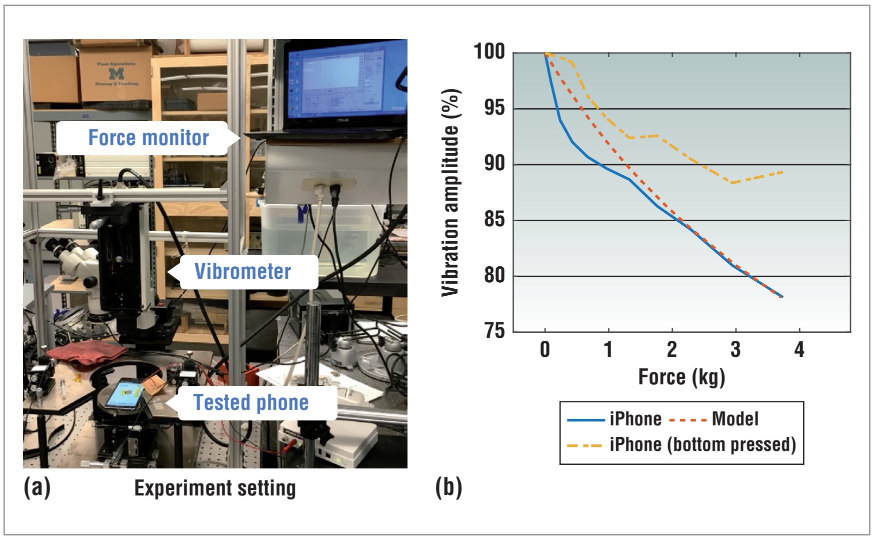

Figure 2. Phone vibration damped by force: (a) experimental setup and (b) the measurement results. The correlation between the damped vibration and the applied force enables ForcePhone's force-sensitive and squeezable interfaces.

SYSTEM DESIGN

As shown in Figure 3, ForcePhone actively plays an inaudible sound with the phone speaker and then picks up this sound with the phone's microphone. The touchscreen input and the data from the other motion sensors are also recorded. This sensor data is then used to improve the force estimation and reduce the number of false detections. When force is applied to the touchscreen or other parts of the phone, the action analyzer triggers the predesigned feedback/behavior based on the monitored (inaudible) sounds and user actions.

Sound Selection

The design of sound is critical to system performance. Although there are many other possible options, ForcePhone's current design uses a 1,200-sample linear chirp from 18 to 24 kHz. A hamming window is multiplied to the first and last of 300 samples to eliminate the audible noise caused by spectral leakage. The main frequency range of sound signals used for sensing is approximately 20 to 22 kHz, while the remaining signals (close to 18 and 24 kHz) are played with a minimal volume and are "stuffed" to avoid signal loss due to the windowing. ForcePhone samples this chirp at 48 kHz and replays it every 50 ms. This sound is designed to achieve minimal user annoyance, a high signal-to-noise ratio (SNR), and adequate force-sensing delay.

ForcePhone uses the signal correlation (also known as the *matched filter*) to estimate the reception of the played sound. The SNR of this correlation in the chirp is proportional to the signal length and sweeping frequency.[6] We selected the sweeping frequency above 18 kHz to not annoy users and cope with the hardware limitation while setting the signal duration to 25 ms. Even though a longer chirp can achieve a higher SNR, it also increases the sensing delay because ForcePhone must wait for the completion of the sound being played. To strike a balance between SNR and the sensing delay, ForcePhone sets the chirp duration to 25 ms. There is another 25-ms stop time after playing each chirp to avoid inter-chirp interference, which makes the total delay in sensing each chirp equal to 50 ms (that is, 20 force estimations can be made every second). This setting provides sufficient SNR to estimate the applied force

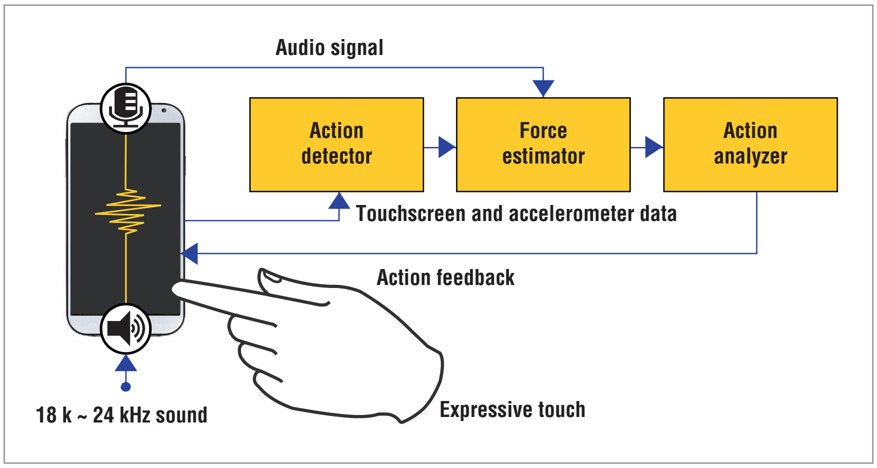

Figure 3. System overview. Force applied to the phone damps the inaudible sound sent from the phone's speaker to its microphone. Accelerometer and gyroscope readings are used to avoid other audio signal noises caused by movements.

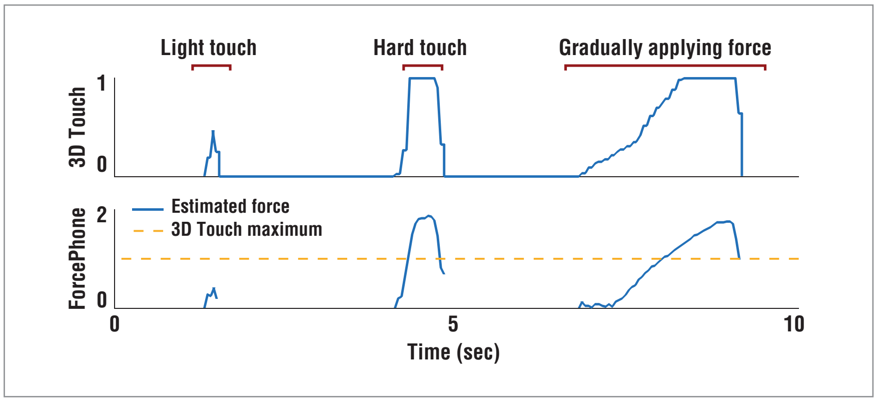

Figure 4. Responses of different amounts of applied force.

and provides an adequate sensing delay to meet most users' needs. None of the participants in our user study noticed or heard the sound used in ForcePhone. The other detailed settings can be found in our related technical paper.[7]

Applied Force Estimation

In addition to structure-borne propagation, there are other factors that affect the received sound strength. For example, the airborne propagation might be blocked by hand, and the overall sound signal strength might be enhanced by reflections from the environment or the internal resonant. In ForcePhone, these noises are identified and removed by timestamping the received audio signal. For example, the reflection from an object 10 cm away will be received 28 samples later than the airborne propagation, because it travels 10 cm farther. Moreover, sound usually travels 100 times faster in a solid phone body.[8] Thus, the structure-borne propagation will be received 21 samples ahead of the airborne propagation when the microphone and the speaker are 15 cm apart. Based on these observations, ForcePhone uses the signal that's 20 samples ahead of the airborne propagation as the indicator of the structure-borne propagation, thus removing the most undesirable noise.

Note that the reference of airborne propagation is assumed to be the strongest audio correlation, because the sound energy decays faster through the solid phone body and is absorbed more on the reflection paths than air. Ideally, the signals detected before the airborne sound should represent only the structure-borne sound propagation. However, in our measurement, the signals from multiple paths were mixed together due to the audio distortion and the adoption of windowing. The windowing process suppressed the frequency-domain signal leakage but incurred the time-domain signal leakage. This 20-sample-ahead sampling heuristic thus includes both air- and structure-borne propagations.

To get a reliable estimation of the applied force, ForcePhone uses the sound strength when the touch begins as a reference to estimate the subsequent change caused by the force applied later. That is, the signal components from the other paths are removed by subtracting the current estimation from the reference signal. This heuristic is designed based on the assumption that the signal of the other paths won't change as significantly as the change of structure-borne signals when the user applies force to the phone.

Figure 4 shows a real-world example of applying force to an iPhone 6s placed on a wooden table. In this measurement, there are three different types of touch, each with a different applied force: light touch, hard touch, and touch with a gradually increasing force. The ground truth of the applied force can be read from the Apple 3D Touch sensors. As shown in the figure, our heuristic captures most of the force-changing characteristics even when the applied force exceeds the maximum sensing range of Apple 3D Touch.

TESTING APPLICATIONS

Figure 5 shows three applications that have been implemented on ForcePhone. The first one is a copy of Apple 3D Touch functionality, where users can get the option menu by hard-pressing the app icon.

The second app is a generalization of the first app. Users move a ball continuously by applying different amounts of force to a "soft" button and then stop the ball by releasing the button. Users must stop the ball at the red-marked box to succeed, and the number of boxes increases when users pass the current level; the highest box is reached by applying a 500-g force. This app follows a design principle similar to that of Gonzalo Ramos, Matthew Boulos, and Ravin Balakrishnan[9] in an attempt to understand if the user can effectively control the force at different levels.

The last app lets users surf previously viewed pages/screens by double-squeezing the phone body (instead of clicking the back button in the left-top corner). This app simulates a useful control alternative to users, especially when they are operating the phone with only one hand.

ForcePhone isn't limited to these three applications; we've also implemented many other applications, such as controlling the speed of a racing car game by hard-pressing the gas pedal icon or compressing a virtual hand trainer by squeezing the phone body. See our demo video to learn more about the usage of ForcePhone (https://youtu.be/cYxr2wnQVMU).

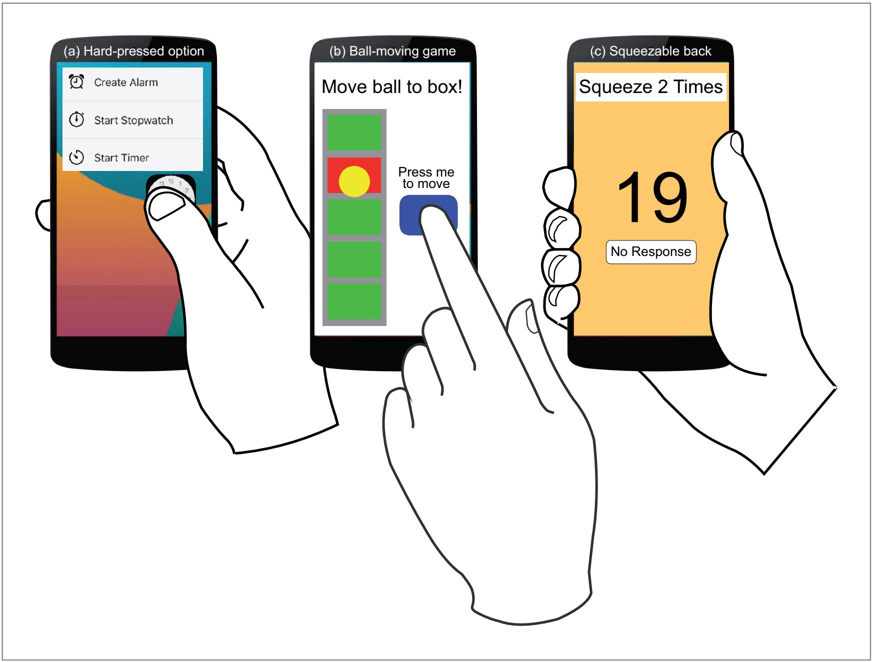

Figure 5. Experimental apps. Users can trigger option menu, control ball movement, or switch to the previous page by hard-pressing a button or squeezing the phone body.

PERFORMANCE

In our measurements with having a user hold an iPhone 6s in his or her right hand, the mean square error of this in-hand example is 205 g, and the correlation coefficient to the force estimated by external sensors is 0.87. This estimation error is reduced to 54 g when the phone is placed on a wooden table. Note that the error in estimating the exact value of the applied force could change if the force were applied in a different way from our calibration (because we don't consider the damping coefficient to change in our current model). For example, there might be an estimation drift, so applying a varying force from 500 g to 1,000 g might be estimated incorrectly as changing from 400 g to 900 g, plus the previously mentioned errors.

However, this feature doesn't hurt the experience of using ForcePhone, because users aren't aware of the exact value of the force applied to a phone unless this value is shown in the user interface.[9] With a proper user interface design, even though the estimated force is 100 g less than the real force applied to the phone, users can easily learn to adjust the applied force for getting a correct response. As shown in our usability study, this setting is adequate for building useful force-sensitive and squeezable apps.

ForcePhone is resistant to most external noises, such as chat or background music, because human noise doesn't overlap with the frequency used by ForcePhone (that is, higher than 18 kHz), and because human noise usually has different structures from the sound used by ForcePhone, so the noise can be filtered out during our sample data processing. Likewise, ForcePhone also allows other sounds (such as the game background music) to be played by the same sensing device unless the speaker isn't saturated. For example, in the current setting, the inaudible sound is played at 50 percent of the speaker's volume to estimate touch force, thus allowing the other sounds to be played by the other 50 percent of volume. Last, ForcePhone can be used simultaneously by multiple devices because each device only uses a short period (that is, only 20 samples ahead of airborne sound) of signals to estimate the touch force, so multiple devices will unlikely play and use the same period of sound at the same time.

USABILITY STUDY

We conducted a two-part experiment to study ForcePhone's usability. In the first part, we asked users to finish certain tasks and analyzed their reactions, including the average time needed to finish the test. Users repeated the tasks many times under different situations for comparison. The second part of experiment aimed to study users' perception, such as whether users viewed the accuracy or delay as acceptable. In this test, we asked users to use the three testing apps shown in Figure 5 for 20 minutes total and then fill out a survey form. We recruited six participants (two female and four male) to join the first part of experiments and 21 par-

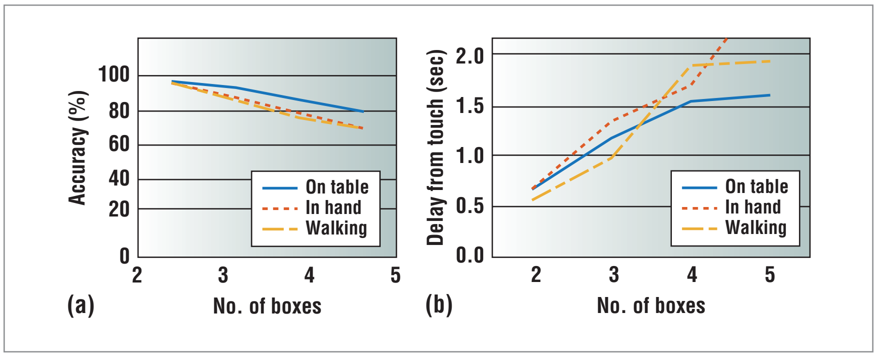

Figure 6. The result of controlling a ball with ForcePhone. Results are averaged over six participants. Delay is estimated as the time between the user presses/releases the button.

TABLE 1
ForcePhone study results.

Statements	Strongly disagree/disagree	No opinion	Strongly agree/agree
Hard-pressed option is helpful	0	0	21
Hard-pressing (3D Touch*) is responsive	1	1	19
Hard-pressing is responsive	0	1	20
Ball-moving game is interesting	0	3	18
Moving ball is responsive	2	4	15
Squeezable back is helpful	1	2	18
Squeezing is responsive	2	1	18
False detection is acceptable	4	7	10

* This statement refers to the hard-pressed option implemented by Apple 3D Touch; the others refer to ForcePhone.

ticipants (seven female and fourteen male) to test the second part.

Figure 6 shows the accuracy (the success rate of moving the ball into the selected box) and the delay of using our ball-moving app. The plots show that most participants could effectively control ball movement with ForcePhone when there were only two to three boxes. This result supports our hard-pressed option app, because the users could easily control the applied force at two levels with higher than 97 percent accuracy. When more than three boxes (levels) are provided, users could still achieve higher than 90 percent accuracy when the phone was stationary on the table.

On average, users required only 0.7 second to stop the ball at the correct position when there were two boxes. This delay increased to 2 seconds when the users attempted to move the ball among five boxes while walking. The average accuracy of repeating the same task on the Galaxy Note 4 increased by 3 percent. We think this minimal improvement is due to the users' familiarity with this task rather than the force estimation accuracy.

In summary, users could effectively use ForcePhone to control different levels of applied force for various scenarios. We also showed that users had more than 90 percent accuracy in controlling our squeezable-back app, and the squeeze behavior could be detected even when the phone was in the user's pocket (see elsewhere[7] for details).

The results of our survey are summarized in Table 1. Most users had a positive view of the proposed apps and found them helpful. For the hard-press option, most users thought ForcePhone had a comparable performance to Apple 3D Touch. One user said ForcePhone is better than 3D Touch because the vibration in Android is much clearer (stronger) than iPhone's, which isn't related to the force detection. Only three users thought iPhone's performance was better, but they still acknowledged ForcePhone was responsive enough for the hard-pressed option app. This indicates that ForcePhone can handle simple tasks with a comparable performance as adding proprietary hardware sensors. Moreover, most users feel the squeezable back app is helpful, which is a unique capability of ForcePhone.

Most users thought that controlling the ball based on the applied force was relatively difficult, but they still were able to control the ball. Two users thought our test setting was too sensitive, making it difficult to move the ball. We also discovered some errors caused by applying a large initial force and releasing the button immediately, which wasn't the intended case for ForcePhone. After the users were instructed to move the ball by gradually applying force, they were able to control the ball with ForcePhone. The squeezable back app received a similar rating as the ball-moving game.

In another survey, 16 users indicated difficulty in clicking the app back button when operating the phone with one hand, which favors the design of our squeezable back app. Most users thought our current parameter setting tuned by the previous six participants was responsive and acceptable. Half of the users experienced false detections when they moved the phone from one hand to the other. During the test, none of the users heard the sound used in ForcePhone, so no user annoyance was reported. The test locations were near a cafe crowded with students, but ForcePhone was robust to human noises. Most common comments from the users were that it was a "cool idea" and "useful."

We plan to have extensive and large-scale testing before releasing ForcePhone to the public. Since our university's press release of this technology, we have received many useful comments from industry companies about its commercialization. The most important comments related to calibrating the touch force among different device touchscreens and identifying the proper force sensitivity setting for general users. Other open issues include understanding the side effect of the emitted sounds to humans or animals and the energy consumption of audio hardware. We've already solved some of these issues, such as only playing the sound when users begin touching the screen and stopping it when the touch is released, mainly to reduce energy consumption. P

ACKNOWLEDGEMENTS

We thank Karl Grosh and Chuming Zhao for the help of vibration measurements and Kassem Fawaz, Lung-Pan Cheng, Chuan-Che Huang, Ming-Yuan Yu, Chen-Ming Chang, and Meng-Ting Chung for constructive comments.

REFERENCES

1. B. Lovejoy, "Will Limited Device and App Support Lead 3D Touch to Wither and Die?" *9 to 5 Mac,* 25 Mar. 2016; http://9to5mac.com/2016/03/25/3d-touch-future-opinion.
2. M. Campbell, "Why Apple's iPhone SE Lacks 3D Touch Technology," *Apple Insider*, 23 Mar. 2016; http://appleinsider.com/articles/16/03/23/why-apples-iphone-se-lacks-3d-touch-technology.
3. S. Low et al., "Pressure Detection on Mobile Phone by Camera and Flash," *Proc. 5th Augmented Human Int'l Conf.*, 2014, pp. 11:1–11:4.
4. S. Hwang and K.-y. Wohn, "Pseudobutton: Enabling Pressure-Sensitive Interaction by Repurposing Microphone on Mobile Device," *Proc. Extended Abstracts on Human Factors in Computing Systems* (CHI), 2012, pp. 1565–1570.
5. S. Hwang, A. Bianchi, and K.-y. Wohn, "Vibpress: Estimating Pressure Input Using Vibration Absorption on Mobile Devices," *Proc. 15th Int'l Conf. Human-Computer Interaction with Mobile Devices and Services* (Mobile-HCI), 2013, pp. 31–34.
6. S. Salemian, M. Jamshihi, and A. Rafiee, "Radar Pulse Compression Techniques," *Proc. World Scientific and Engineering Academy and Society* (WSEAS), 2005, pp. 203–209.
7. Y.-C. Tung and K.G. Shin, "Expansion of Human-Phone Interface by Sensing Structure-Borne Sound Propagation," *Proc. 14th Ann. Int'l Conf. Mobile Systems, Applications, and Services* (MobiSys), 2016, pp. 277–289.
8. Y.-H. Kim, *Sound Propagation: An Impedance Based Approach*, Wiley, 2010.
9. G. Ramos, M. Boulos, and R. Balakrishnan, "Pressure Widgets," *Proc. SIGCHI Conf. Human Factors in Computing Systems* (CHI), 2004, pp. 487–494.

Kang G. Shin is the Kevin and Nancy O'Connor Professor of Computer Science and Founding Director of the Real-Time Computing Laboratory in the Department of Electrical Engineering and Computer Science, at the University of Michigan, Ann Arbor, Michigan. Contact him at kgshin@umich.edu.

Tung Yu-Chih is a PhD candidate in computer science and engineering at the University of Michigan, Ann Arbor. Contact him at yctung@umich.edu.