DopLink: Using the Doppler Effect for Multi-Device Interaction

Md Tanvir Islam Aumi[1], Sidhant Gupta[1], Mayank Goel[1], Eric Larson[2], Shwetak Patel[1,2]
[1]Computer Science & Engineering, [2]Electrical Engineering
UbiComp Lab, DUB Group, University of Washington, Seattle, WA 98195
{tanvir, sidhant, mayankg, eclarson, shwetak}@uw.edu

ABSTRACT
Mobile and embedded electronics are pervasive in today's environment. As such, it is necessary to have a natural and intuitive way for users to indicate the intent to connect to these devices from a distance. We present DopLink, an ultrasonic-based device selection approach. It utilizes the already embedded audio hardware in smart devices to determine if a particular device is being pointed at by another device (*i.e.*, the user waves their mobile phone at a target in a pointing motion). We evaluate the accuracy of DopLink in a controlled user study, showing that, within 3 meters, it has an average accuracy of 95% for device selection and 97% for finding relative device position. Finally, we show three applications of DopLink: rapid device pairing, home automation, and multi-display synchronization.

Author Keywords
Doppler effect; multi-device Interaction; pointing; pairing.

ACM Classification Keywords
H5.m. Information interfaces and presentation (*e.g.*, HCI): Miscellaneous.

INTRODUCTION
With the proliferation of multiple smart computing devices in the environment—personal computers, smartphones, smart TVs, and other network-enabled devices (i.e., the smart home)—it is becoming increasingly important for these personal mobile devices to connect to each other and share information. However, there are no easy approaches for selecting which devices to connect to, nor is there a way to select a subset of devices easily at a distance. In particular, selecting, controlling, and sharing information between devices still requires a significant effort from a user. For example, to select one or more co-located wireless devices, a user needs to know each device address or have physical access to each device. As the number of devices increases, this becomes tedious and unmanageable. This is in contrast to accomplishing selection in a more natural way – such as simply pointing at the devices.

Figure 1: ***Top:*** **Scanning gesture;** ***Bottom:*** **Pointing gestures:** ***(Left)*** **Flick gesture,** ***(Right)*** **Push gesture**

This is a well-known problem. Numerous techniques have been proposed for selecting objects in the physical environment. Many solutions require extra sensors to identify devices [2], or they require simultaneous movement of both devices [3,5] which is not feasible for larger devices (*e.g.*, TVs, printers) or devices embedded into the environment (*e.g.*, furniture). There are also many camera-based solutions for household device selection [6], but these are sensitive to lighting conditions and occlusion. More importantly, camera-based solutions such as blinking LEDs or QR code based systems do not blend into or *disappear* into the environment. To address these limitations, we present *DopLink*, an ultrasonic based device selection approach. It utilizes low-cost, commodity audio hardware, often already embedded in smart devices, to determine if a particular device is being pointed at by another device. With DopLink, the user moves their mobile phone at another device in a pointing motion. This technique uses a well-understood phenomenon known as the Doppler effect, which characterizes the change in observed frequency of a sound wave as a source moves towards or away from the receiver. Contrary to vision based approaches, DopLink is much less susceptible to occlusions

UbiComp '13, September 8–12, 2013, Zurich, Switzerland.

http://dx.doi.org/10.1145/2493432.2493515

between the controller and receiver, and is invisible to the user. Our technique is simple, robust, and easy to implement—it can be developed on cheap hardware at low cost or on top of existing commodity hardware.

DopLink is also able to find the relative position of each device by implementing a "scanning gesture" (Figure 1, *Top*). We evaluate DopLink in both quiet and noisy environments and at varying distances to devices being selected. DopLink performs at-a-distance target selection with an average accuracy of 95% and extracts the relative position between devices with an average accuracy of 97%, both at distances up to 3m. Finally, we have developed example applications to showcase the utility of the approach. In particular, our video figure shows example applications of how the system can be used to (1) rapidly pair and un-pair devices for grouping or sharing information, (2) control home appliances with minimal instrumentation, and (3) ascertain relative position of separate monitors to stitch together a multi-display system.

THE DOPLINK SYSTEM

DopLink leverages the Doppler shift of an inaudible tone to initiate a connection between multiple devices. When the user wants to connect to a particular device, they hold a button to initiate an inaudible tone from their mobile phone and then make a pointing gesture with their phone towards a target (Figure 1, *Bottom*). Since the velocity of the sound source changes, a Doppler shift can be observed by all the target devices in the vicinity. DopLink works on the hypothesis that the intended target device will receive the maximum frequency shift compared to the other potential target devices. By comparing the peak frequency shift in all the devices, we infer the intended recipient.

Ultrasound Doppler shift technique has been used before to detect gestures and motions. In [1], we used Doppler shifts to detect gestures. DopLink, builds upon that method, but goes beyond gesturing. Instead of detecting hand gestures in front of a laptop, we compare Doppler shifts in *multiple devices* and from the comparison, infer the relative position of all devices and the target device.

Similarly, Peng et al. proposed a system that used a pointing gesture to select a particular smartphone or tablet using audible chirping sounds [4], where each recipient calculates the interval between two sequential "chirp" sounds emitted by the source. Our system is similar in spirit to this work, but has some distinct advantages: DopLink is less reliant on the mechanics and timing of the gesture; and allows continuous interaction, like scanning devices in an in-air sweeping motion while producing no audible sound.

Theory of Operation

In the remainder of this work we refer to the device used to gesture as the "source" and the devices sensing the Doppler shift as the "receivers." The observed frequency shift sensed by a receiver is proportional to the source frequency (f_S) and to the relative velocity at which the source moves. In our approach, initially the source and receiver are stationary, and there is no change in the receiving

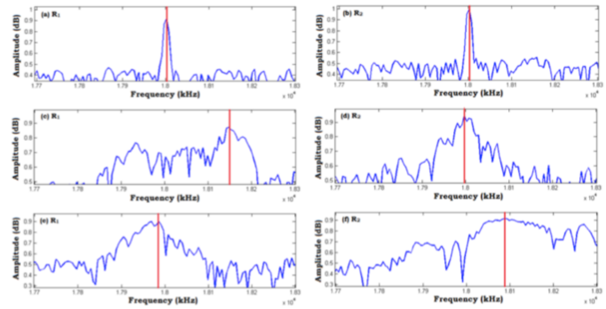

Figure 2: (Top) Peak signal observed by both devices is similar; (Middle) shift in frequency observed by R1 higher; (Bottom) shift in observed frequency greatest for R2.

frequency. When a user moves the source towards the receiver, it causes a shift in observed frequency. The amount of this shift is measured by the receiver (f_R) and can be described by the equation: $f_R = f_S.(\frac{c+v_R}{c-v_S})$ where c, v_S, and v_R are the velocity of sound in air, of the source, and of the receiver, respectively. In our system, the receiver is stationary ($v_R = 0$). Thus, as the source velocity (v_S) increases, the received frequency increases proportionally.

Algorithm and Implementation Details

In DopLink, a source device generates a continuous tone, played through the device's speakers at 18 kHz. Although we have verified that it can operate down to 6 kHz, we favor 18 kHz since it is generally inaudible but detectable by nearly all commodity hardware. Additionally, higher source frequencies result in greater Doppler shifts, making it computationally easier to estimate motion at a given frequency resolution.

All receivers continuously sample their microphone at 44.1kHz. On each receiver, we buffer the incoming audio signal into 4096-point, non-overlapping frames and compute the $|FFT|$ of each frame. This yields a frequency resolution of 10.8 Hz per bin and frame rate of about 10Hz.

Device Selection

The device selection algorithm can be broken down into four steps: (1) initialization, (2) signal conditioning, (3) peak shift calculation, and (4) final selection.

Initialization: When the system starts, the receiver scans the frequency range between 17-19kHz and finds the bin with the largest magnitude, *i.e.*, a sharp peak as a result of our source tone. This bin should remain consistent across multiple FFT vectors. For our system, we define this as when the same bin is selected for 5 consecutive frames. We save this index, N_{source} and wait for detecting shifts. Figure 2 (*Top*) shows these peaks when there is no Doppler shift.

Signal Conditioning: If we conservatively bound the gestures of interest to a maximum speed of 6m/sec ([1] reports 3.9m/sec maximum speed of users moving hands), then the maximum shift of the system can be ~31 frequency bins. Thus we limit the signal of interest to 31 bins.

Peak Shift Calculation: For each FFT frame we store the frequency bin with the largest magnitude, N_{peak} and compare it to N_{source}. The difference between these peaks captures the observed Doppler shift, ΔN. However, each user may perform a pointing gesture with slightly different velocities and durations. Moreover, because each frame contains only 100ms of data, a consistent shift will not always be seen in consecutive frames. As such, we devised an approach that estimates the start and ending of the gesture, stores the Doppler shift over all frames during the gesture, and sums the observed Doppler shifts over the entire gesture.

To detect the start of the gesture we look for a shift greater than ΔN_{thresh}=4 bins. This threshold was chosen based on an experiment where the tone was played for a long duration in front of a receiver without any gestures being performed. The maximum observed value of ΔN was 2. We chose 4 to be conservative. The end of the gesture is determined when ΔN<4 for 4 consecutive frames (about 400ms). We then take the sum of ΔN's for each frame during the detected gesture. This sum, ΔN_{Σ}, is transmitted to a central server.

As an example, assume we have one source S and two receivers R_1 and R_2. Initially both the receivers are listening to the peak signal (Figure 2, *Top*). As the user points their device to receiver R_1, both R_1 and R_2 will observe a frequency shift. However, the shift received by R_1 would be larger than R_2 (Figure 2, *Middle*), or pointing at R_2 results in shift greater for R_2 (Figure 2, *bottom*). Thus, by comparing these two shifts, we can infer which of the two devices the user pointed towards.

Final Selection: In this step, all the receivers report the value of peak frequency shift to a central server[1]. The server calculates the maximum value among all those shifts and sends a message back to the receiver whose frequency shift was largest. Now, a connection can be set up between the source and the intended receiver. Once the first device connects, the user can quickly point towards another device and connect to more devices in rapid succession. Hence, multi-pairing can be easily accomplished. Moreover, once a group of devices is established, all devices are continually listening for a tone from an outside device. A new device can quickly be added to the group simply by gesturing at any device in the group.

Relative Device Positions

In this implementation, a single device is being used to ascertain the relative position of other devices. Here, the user initiates a scanning gesture across the devices of interest (Figure 1, *Top)*. All the devices in front of the source will sense a frequency shift and report it to the server, as before. However, the time stamp of when they detected the gesture will be different. If scanning is performed from left to right, then the leftmost device will observe the shift first (and vice-versa for scanning right to left), then the second device, and so on. The server in this case, instead of calculating the maximum, organizes the devices in a sequence based on their arrival time (in all of our implementations network delay is much less than the time it takes to perform a scanning gesture). The server then sends the sequence to all the clients so that they are aware of their position relative to other devices. We employed this technique to ascertain the relative position of multiple displays and stitch them into a single display.

[1] We chose a central server for rapid prototyping, the algorithm itself could be adapted to distributed and decentralized architectures as well.

EVALUATIONS AND RESULTS

We evaluated the performance of DopLink with 6 participants (3 males, 3 females) ranging in age from 24 to 30. Initially we showed the users two gestures: a flick gesture, like an overhand throw and a push gesture to "select" a target using the phone in their hand (Figure 1, *Bottom*). We then asked the users to perform whichever of the two gestures they preferred for device selection.

The performance of our system depends on the distances and the angles between the source and the receivers. At farther distances, the reduced sound intensity at high frequency can adversely affect the system. If the angle between the two targets is too small, the system cannot reliably distinguish the target devices. We evaluated the effect of these factors on the performance of the system via four experiments.

Inter-device distance fixed, participant-to-receiver distance changed: In this experiment, three laptops were positioned 1m apart. Each participant performed 30 trials towards the middle device at distances of 1, 2, 3, 4, and 5 meters. As a result, angle between the receivers and participant also changes. Any detection from the left or right laptop was recorded as an error. Table 1 shows the percentage of the gestures correctly identified across all users.

Participant-to-receiver distance fixed, Inter-device distance changed: In this experiment, we control the angle between the target devices by arranging them in an arc of constant radius 2 meters. Users again gestured 30 times at the middle receiver. Table 2 shows the percentage of the gestures correctly identified as gestures towards the middle laptop for all users.

Target Distance (m)	Angle (°)	Accuracy (%)
1	45	100
2	26.6	96.1
3	18.4	89.4
4	14.0	77.2
5	11.3	57.2

Table 1. Pointing gesture detection accuracy as a function of participant-to-receiver distance.

Target Distance (m)	Angle (°)	Accuracy (%)
2	26.6	95.0
2	21.8	90.6
2	16.7	84.4
2	11.3	72.8
2	5.7	53.9

Table 2. Pointing gesture detection accuracy as a function of angle between target devices.

Implication: These two experiments show that DopLink performs robustly for distances up to 3 meters and angles down to 18.4°, which are reasonable tolerances for our example applications such as device control and device sequencing. Three meters spans the distance of most living spaces and, with an angular resolution of 18.4°, one can fit up to 5 adjacent devices within a 90° span (*i.e.*, up to 20 devices in a circle). Hence, these two experiments demonstrate the practicability and limitations of our intended applications.

Scanning Gesture and Relative Device Position Accuracy: In this experiment, we placed three laptops 50cm apart, and instructed users to perform a scanning gesture at five different distances, 30 gestures at each distance. The users were asked to randomly perform either the left-to-right or the right-to-left scanning gesture. Table 3 shows the percentage of gestures that resulted in correct sequencing by the system. DopLink robustly allows devices to be grouped and informed of their relative position within 3 meters, but again has trouble distinguishing devices at greater distances. **Implication**: we again conclude that DopLink will work sufficiently for sequencing devices in most applications. We also believe that, with a 100% within 1 meter, having users step closer to screens for a one time sequencing gesture is sufficiently simple and non-invasive.

Distance (m)	Accuracy (%)
1	100
2	99.4
3	92.2
4	83.9
5	77.8

Table 3. Relative position accuracy with respect to distance between the source and the receivers.

Different conditions: To ensure that DopLink works irrespective of background noise, we also evaluated gesturing in different environments of varying sound levels with a subset of the users (N=2). At each location, we placed three receivers separated by 100cm. Users stood at 2m and performed the pointing gesture 50 times towards the middle receiver. Table 4 shows the percentage of correctly identified gestures. **Result**: In the café, the decreased accuracy can be attributed to movement in the surroundings, as people walking by can also create Doppler shifts in the reflected tone. We note, however, that there were not any false positives. This is because the generated tone is short lived. The user holds a button to generate the tone just before gesturing and while gesturing. DopLink identifies and syncs to the tone in under 400ms and finds the shift during the gesture. **Implication**: the system is slightly sensitive to busy areas where others may be walking. Even so, the movement does not result in false selections, but may require users to point more than once at a device to select it.

Lastly, we evaluated DopLink when obstacles exist between the source and receivers. We again used a subset of the participants (N=2) The participants performed the pointing gesture from behind a monitor to a laptop 3 meters away; and from behind partitioning furniture (*e.g.*, whiteboards). We found that DopLink works sufficiently (detecting 77.2% of gestures), though multipath analysis in the future could potentially improve this. Traditional vision based techniques [6] fail in this situation as they need a clear line of sight. In contrast, DopLink might require users to perform a gesture more than once when obstacles are present.

DISCUSSION AND CONCLUSION

In this paper, we described how Doppler shift could be used to enable at-a-distance device selection and device position scanning. We detailed a robust algorithm for detecting which target device a user selects and evaluated the robustness across different devices, users, and locations.

Location	Sound Level (dB SPL)	Accuracy (%)
Apt. near Highway	64	97
Computer Science Lab	60	94
Busy Cafe	74	84

Table 4. Accuracy in different environments

DopLink is a promising approach for enabling impromptu, natural device selection, pairing, and content transfer. On most computing devices, it requires no additional hardware; and only a microphone and simple phase shift sensing hardware on embedded devices. Though we make use of the FFT in the current prototype, a hardware solution could use a local oscillator coupled with an analog phase detector (like those commonly found in low cost FM demodulators) to sense Doppler shift with an 8-bit microcontroller using a low sampling rate ADC.

In the future, the gesture vocabulary and robustness can be improved even further by leveraging multiple tones and more sophisticated gesture recognition. For example, combining our work from [1] to not only select a device, but also transmit control logic with a single gesture.

REFERENCES

1. Gupta, S., Morris, D., Patel, S., and Tan, D. SoundWave: using the doppler effect to sense gestures. In *Proc. CHI 2012*.
2. Hinckley, K. Synchronous gestures for multiple persons and computers. In *Proc. UIST 2003*, 154–158.
3. Patel, S. and Abowd, G. A 2-way laser-assisted selection scheme for handhelds in a physical environment. In *Proc. UbiComp 2003*, 200–207.
4. Peng, C., Shen, G., Zhang, Y., and Lu, S. Point&Connect : Intention-based Device Pairing for Mobile Phone Users. In *Proc. Mobisys 2009*, 137–149.
5. Rekimoto, J., Ayatsuka, Y., and Kohno, M. SyncTap : An Interaction Technique for Mobile networking. In *Proc. MobileHCI 2003*, 104–115.
6. Schmidt, D., Molyneaux, D., and Cao, X. PICOntrol: Using a Handheld Projector for Direct Control of Physical Devices through Visible Light. In *Proc. UIST 2012*, 379–388.