MobiRing: A Finger-Worn Wireless Motion Tracker

Yi-Lin Chen*, Yi-Lung Tsai*, Kailing Huang*, Pai H. Chou†,*
*Dept. of Computer Science, National Tsing Hua Univ., Taiwan
†Center for Embedded Computer Systems, University of Irvine, CA USA
{iverlon,gugigugi323}@gmail.com, phchou@uci.edu

***Abstract*—**

MobiRing is a compact, finger-worn wireless motion-tracking platform that can be reprogrammed for a wide variety of features to augmented mobile computing. First, it works as a wireless mouse in the air without requiring a surface for a mouse or a trackpad, making it suitable for augmented-reality or virtual-reality systems. Second, it supports collection of stroke-in-the-air data that can be interpreted as gestures for commands, text input, or raw strokes. An advanced version can combine input from additional units on multiple fingers for richer possibilities. Third, the ability to quickly switch pairing among multiple host computing systems enables MobiRing to support drag-and-drop-like operations across systems such as PCs, smartphones, tablets, display walls, and even displayless embedded systems, where a wired connection may be inconvenient if not infeasible. MobiRing has been prototyped with an ultra-compact wireless sensing platform with an on-board triaxial accelerometer and a short-range wireless transceiver with an optional second accelerometer for enhanced accuracy. The prototype has been evaluated quantitatively for accuracy and qualitatively for usability by asking ten volunteer novice users to try a variety of the proposed operations. The results show that the MobiRing system to be intuitive for mouse-like functions and the recognition accuracy of each gesture is over 90%.

I. Introduction

Pointing can be an intuitive input modality for post-PC computing systems. By pointing, the user can indicate the focus or select the desired options and operation. Most pointing devices to date require a flat surface, such as the desktop for a mouse or a trackball, the input area for a stylus, or the surface of the trackpad or touchscreen itself. However, for the Internet of Things (IoT), finding an available a surface is not always feasible. For instance, with augmented-reality devices such as Google Glass, the users are expected to interact normally with real-world objects that are annotated with computing objects on the display.

In addition to pointing, users often also need to input data or commands without being able to use a pointing device or a keyboard, both of which also require a surface. Although voice could be used, it is not always socially acceptable. In these cases, gesture-based input can be a good option. Gestures are motion patterns that are mapped to symbols that can be further interpreted as command or data. Motion patterns can be captured using video, accelerometers, and possibly strain and bending-angle sensors. Ideally, if the gesture-input device and the pointing device can be integrated into one unit, then it will be minimal burden to the user.

We propose such a miniature, integrated input device in the shape of a finger-worn ring, called MobiRing, that is capable of *both* surfaceless pointing and motion tracking for finger-gesture input using an accelerometer. Our proposed MobiRing uses short-range wireless communication for maximum mobility and minimum intrusion. Being wireless opens up new opportunities for interacting with not just a single but multiple computing devices. In the simple case, our MobiRing should be able to very quickly pair with the selected device in the most intuitive way, with latencies much shorter than today's Bluetooth devices that often take a few seconds. We propose to take advantage of the motion-tracking feature for quick pairing by simply tapping one's finger with the MobiRing on the target master, which is also assumed to be equipped with an accelerometer. The simultaneously detected complementary motion is what triggers and authenticates the pairing.

With the ability to switch pairing between different devices, MobiRing can support cross-device operations that are natural and intuitive but currently difficult to do with today's systems. For example, one might want to do *cut-and-paste* but across devices, such as from one cell phone to a PC, even though it is a common operation between different running programs on a given PC. Currently there is no easy way to do so, even with wireless mice on the market. Another related operation is *drag-and-drop*, which allows either a data object (such as a graphic), a highlighted text string, or a file to be "moved" by direct manipulation using the pointing device to the desired location. Similarly, such an operation is well supported on modern graphical user interface (GUI) systems but not so smoothly, consistently, or at all across computing devices, mobile or stationary. Instead, users often resort to using a USB stick to copy data manually as files. This may work for PCs but not for most mobile or IoT devices.

The contribution of this work is a compact, finger-wearable motion-sensing platform, called MobiRing, that can address these problems in mobile computing as mentioned above. As a platform, it can be reprogrammed for customization and new features. We have implemented features including mouse-cursor control by finger tilting (without requiring a surface) while holding the thumb button, gesture input while not holding the button, tapping to pair with multiple devices, and software support for cross-device data-transfer operations. We evaluate our prototype both quantitatively and qualitatively by asking ten volunteers to try these functions. Results show that our gesture input achieves over 90% accuracy with high user satisfaction. Although some of the motion tracking features may be achieved by other systems, no other system comes close in the level of integration and controllability offered by our proposed MobiRing system.

The rest of this paper is organized as follows. Section II surveys related work, and Section III describes the principle of operation for the mouse-cursor control. Section IV defines

gestures in terms of basic and compound ones with a technical description for the motion processing techniques and the tapping gesture. Section V discusses communication issues. Section VI describes our system prototypes and discusses their trade-offs. Section VII presents an evaluation of MobiRing by volunteer users. Finally, Section VIII concludes with directions for future work.

II. Related Work

Related work includes ring devices, surfaceless input devices and wireless systems that perform targeted pairing dynamically. Input devices are further divided into pointing devices, motion trackers, gesture input, and other surfaceless input devices. The devices may be applied to but are not limited to virtual reality and augmented reality systems.

A. Ring Devices

A number of ring devices have been announced and will soon enter the market, including the Logbar Ring, Fin, and Nod. Logbar Ring [11] and Nod [14] are similar in that both are worn on the index finger and can convert motion-based gestures to either text symbols or actions for controlling other IoT devices such as lights, remote controlled cars, etc. Fin [18], on the other hand, is a thumb ring with a tongue piece that can distinguish the different divions of fingers being touched to address different IoT device. This is in contrast to drawing a gesture in the case of Ring. Our MobiRing is also a platform that can offer similar capabilities, but we support mainly a tilt-based mouse device that can also transport data as the feature by default.

B. Surfaceless Mouse Devices

Surfaceless mouse devices are primarily in the form of *gyro mouse*, also known as rotational mouse. These devices use a tilt sensor such as a gyroscope to measure deviation from the original orientation. *Roll* (i.e., tilting left or right) causes horizontal movement of the cursor, while *yaw* (i.e., rotating away or towards oneself) causes vertical movement. These devices are relatively large: they are the size of typical TV remote controls and are meant to be grabbed by one hand.

Since gyroscopes are active sensing elements based on either spinning or vibrating structures, they consume nontrivial power, at least until recently. In contrast, our MobiRing has the form of a ring worn on the index finger. All five fingers are still available for typing on a keyboard, grabbing an object, or other uses. We use accelerometers for tracking the gravity vector. It measures absolute deviation and consumes much lower power.

A data glove is a 3D input device with embedded sensors that can sense a variety of finger and hand actions. For example, P5 Glove [2] uses bending sensors to enable user interactions with 3D and virtual environments. Pinch Glove [3] is a pair of stretch-fabric gloves including sensors in each fingertip, which can detect gestures and can be programmed for special meaning such as on/off commands. Data gloves can be an intuitive and interactive way for input in virtual-reality environments. However, they may be somewhat intrusive to the user when the actual touch matters.

C. Gesture-Based Computing

Gestures can be captured by vision or by motion analysis. We assume motion analysis as it is easiest to measure with accelerometers and other types of miniature sensing devices (including but not limited to gyroscopes, compass, etc). Motion analysis is also of low complexity, and here we assume the finger-worn MobiRing can measure acceleration when worn on the user's primary pointing finger, which is usually the right index finger.

Sawada [19] uses a triaxial accelerometer to capture the changes of acceleration and rotational force. These sequential acceleration data are matched to standard patterns to discriminate 10 different kinds of gestures used in a real-time music control system. Benbasat [5], [6] used an inertial gesture recognition framework to measure human motions. Data produced by a hexaxial inertial measurement unit (IMU) are analyzed by a recognition algorithm that considers data on a per-axis basis and are categorized as simple motions. Ubi-Finger [22] uses an accelerometer along with touch and bending sensors to realize sensing operations for information appliances. Integrating sensors with infrared transmitters have been used as a general-purpose remote control. Acceleration Sensing Glove (ASG) [16] is a wearable input device that can interpret human gestures from raw acceleration data. ASG analyzes static data produced by multiple accelerometers and uses the idea of gravity-induced acceleration bias to develop a pointing device.

D. Gesture Modeling

Besides static data analysis and pattern matching, Hidden Markov Model [17] (HMM) has been applied to gesture recognition for modeling time-series with spatial and temporal variability. Acceleration-based gesture recognition using HMM has been studied in works of Mäntyjärvi [9], [10], [12], [13]. They discuss the control for design environments and information appliances. Hoffman [8] used a glove-based accelerometer system with HMM for the recognition of a subset of the German sign language.

E. Dynamically Paired Wireless Connection

Bump [1] is an app for accelerometer-equipped smartphones. As long as it is installed on two devices and the GPS is functioning, the users can transmit data from one device to the other by bumping them together. The bump event can be detected by the triaxial accelerometer. Once the event is detected, the smartphone will connect to the Bump server, which will determine which two devices should establish a connection. However, it is useful for data transfer only; it is neither a pointing device nor a gesture-based input system.

EcoRaft [21] is a multimedia environment for users to interact with animated characters in a virtual outdoors setting. The room has several computers projecting parts of this virtual environment on different screens, and a user can take a tablet computer as a "raft" and approach one of these stationary computers. One of the animated characters, also called agents, can then "hop" from the stationary computer to the tablet. The user can then take this raft with the agent and approach another stationary computer, and this agent can then "jump off" the raft onto the destination shore. The EcoRaft mechanism can be

applied to drag-and-drop of data across computers. However, the user has no direct control as to which piece of the data to grab from the source system or where the data should go on the target system.

III. Mouse-Cursor Control

MobiRing operates in mouse-tracking mode when the button is held down with a thumb. The triaxial accelerometer on the MobiRing is used for moving the mouse cursor. This section describes the principle of operation.

A. Mouse-Cursor Movement by Tilting

We use tilting to control the moving direction of the mouse cursor. That is, *roll* and *yaw* indicate horizontal and vertical cursor motion, respectively. Although this appears similar to how a gyro mouse works, the principle of operation of our system is based on gravity pull. When perfectly horizontal, the Z axis of the accelerometer senses the 1 g gravity pull, while the X and Y axes experiences 0 g. When the accelerometer on MobiRing is tilted about the X or Y axis, the gravity causes a component force on the other axis. Right-tilting causes the cursor to move in $+$X direction, left-tilting for $-$X, down-tilting for $+$Y, and up-tilting for $-$Y. We can use the degree of inclination to determine the velocity of the cursor. The larger the angle of inclination, the more gravity exerts the component force on that axis.

B. Discussion

Our accelerometer-based mouse-cursor control can work quite well and mimic gyro-based ones. One difference though is that gyro ones can continue to work when the user is mobile and moving or even in zero-gravity settings, since the gyro outputs deviation from the original orientation. However, if the mobile user is also rotating, then the gyro mouse can get confused. On the other hand, ours assumes that the user is on earth, upright, and not moving while using MobiRing in mouse mode. This assumption reduces complexity of implementation. Our system will not work in zero-gravity settings or when the user may be lying down horizontally while pointing upright by default, but we believe the trade-offs are reasonable. Extensions are possible by dynamic compensation.

Another consideration is the use of threshold or hysteresis. The user is unlikely to be perfectly horizontal with their fingers at all times. As a result, the mouse cursor may be moved unintentionally. Fortunately, as soon as the user releases the thumb button, all mouse motion stops, so they are able to keep the mouse cursor in place. However, it is still be useful to suppress motion until the tilt exceeds a threshold.

IV. Hand Gestures

With the thumb button released, MobiRing operates in gesture mode. In this section, we describe the basic gestures based on which compound gestures can be defined along with the corresponding commands. One special gesture is called a tap, which involves tapping the finger wearing MobiRing on a targeted computing host device for pairing. Tapping requires cooperation between the tapped host and MobiRing with several possible ways of partitioning of responsibilities.

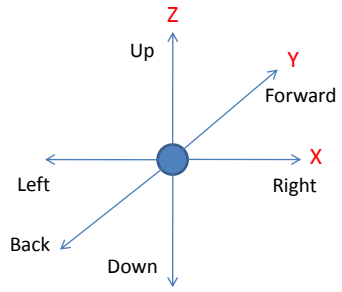


Fig. 1: The directions of the six basic gestures.

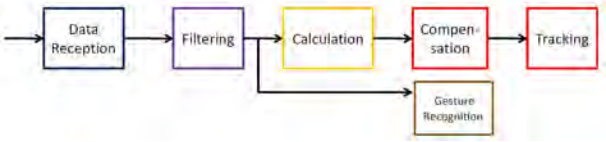


Fig. 2: Gesture Recognition flow diagram

A. Basic Gestures

We define six basic gestures: UP, DOWN, LEFT, RIGHT, BACK, and FORWARD, as shown in Fig. 1. UP and DOWN are vertical, straight-line movements of the finger going upwards and downwards, respectively (not to be confused with tilting to move the cursor). LEFT and RIGHT are horizontal, straight-line movements of the finger in left-to-right and right-to-left directions, respectively. BACK and FORWARD are also horizontal, straight-line movement of the finger but in forward-to-back and back-to-forward directions. Note that these are gestures that have no intrinsic meaning, and in fact they can be viewed as mapped to a subset of keys on the keyboard.

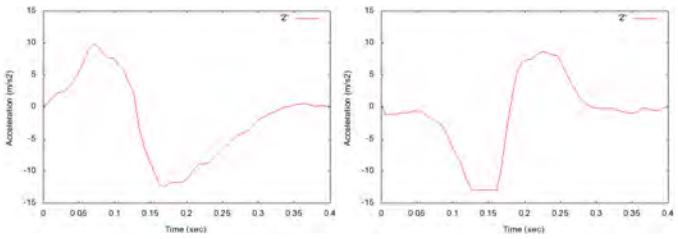
B. Recognition of Hand Gestures

The process of gesture recognition is shown in Fig. 2. It can be executed on a PC, smartphone, tablet, or the ring.

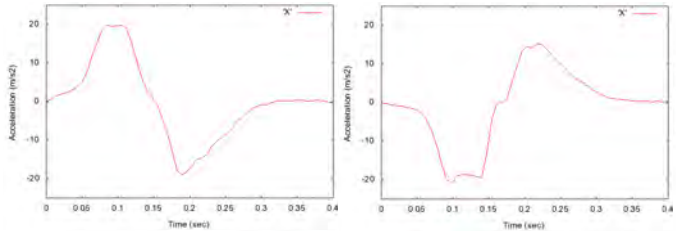
1) Filtering: Filtering is needed to eliminate noise or compensate for other effects on the sensors. The two most relevant filters are for the offset of the accelerometers' output and the effect of gravity, both of which will have a great impact on tracking or recognition results. Hence, we must devise a mechanism to filter the signals either by hardware or software approaches. Below describes how we remove the offset by software methods.

The accelerometer's values as output by the ADC must be offset by an amount that reflects the specific hardware configuration. Otherwise, it will become a drift problem. That is, if the sensor is under 0 g acceleration, the output is a constant and non-zero value initially. When using an analog accelerometer, the offset value of the 12-bit ADC is 2680, which encodes 0 g gravity. The translation equation is

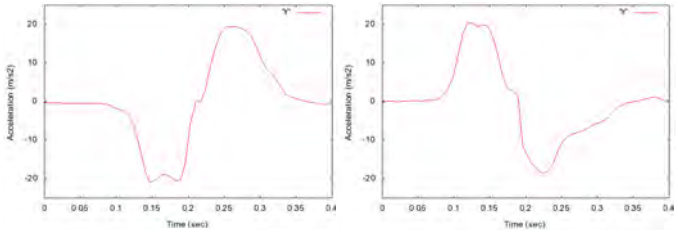
$$\begin{aligned} a &= \frac{(\text{ADC Output} - 2680)}{4095} \times \frac{2200\text{mV}}{333\text{mV/g}} \times 9.8\,\text{m/s}^2/\text{g} \\ &= 0.015810(\text{ADC Output} - 2680)\,\text{m/s}^2 \end{aligned} \quad (1)$$



(a) Up (b) Down



(c) Left (d) Right



(e) Forward (f) Back

Fig. 3: Corresponding patterns of the six basic gestures

where the operation voltage is 2.2 V and the output voltage at 1 g acceleration is about 333 mV, since the output voltage is linearly proportional to the acceleration.

2) Compensation for Motion Tracking: One commonly known problem with accelerometer-based trajectory trackers is error accumulation with unbounded growth due to many integration steps involved. Usually, an accelerometer-based system can be aided by secondary sensors such as gyro or compass along with Kalman filter to reduce the error accumulation. Some may also use external measurement to compensate for the errors.

For resource-constrained platforms, several techniques recalibration can address the problems of either high complexity of high error. The first is the tilt, as described before, as it incrementally steers the mouse cursor without double integration. The second technique is Zero Velocity Compensation (ZVC) [4], which resets the error accumulation between phases of motion sequences. That is, when the sensor detects static acceleration in a period of time continuously, it is interpreted as a pause in motion, and we can then assign the current acceleration and calculated velocity to zero directly to reduce the error accumulation. This condition can be described by the following equation:

$$\Delta a(t) = \frac{V(t_2) - V(t_1)}{t_2 - t_1} = \text{Constant} \qquad (2)$$

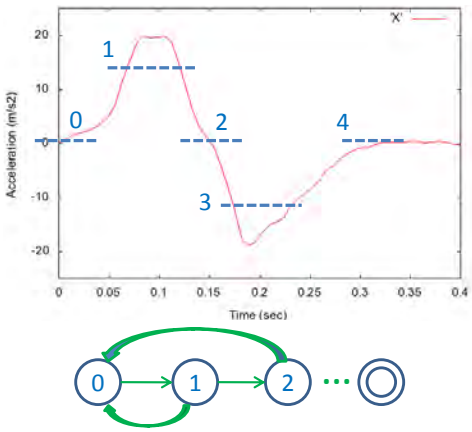


Fig. 4: Time history and corresponding FSM for Recognition Process

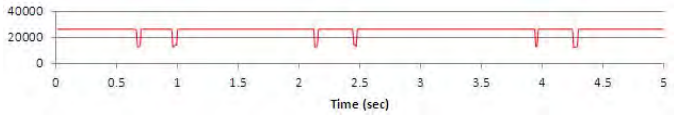
3) Gesture Recognition: Strictly speaking, gesture recognition does not need to be done in terms of position and velocity, but can be done all in terms of acceleration. This approach, called *non-reference tracking*, is not only simpler by eliminating the integration step but also avoids the error accumulation problem. We decompose every complex gesture into six basic gestures. They move along each of the $\pm$X, $\pm$Y, and $\pm$Z axes in 3D space as shown in Fig. 3. For every gesture, we use a simple finite state machine (FSM) with a set of thresholds to track the recognition state. Assume that the ring is stationary initially, the accelerations are constant values. After the ring starts moving, the accelerations will reach the first threshold, and the FSM will go to the next state. While encountering some variations of the gesture, it will reach another threshold and go to another state, and so on. If some failure conditions are met, then the FSM will go back to the initial state and resume the recognition process. Finally, when reaching the final state, it will trigger the action corresponding to this gesture. The process is shown in Fig. 4.

We set the thresholds to $\pm$1.5 g relative to the initial state to strike a balance between sensitivity and specificity over different gesture classes. In addition, we set a time interval for each gesture class. Complex gestures, such as handwriting, have a longer time interval in the recognition process, while simpler ones have a shorter interval.

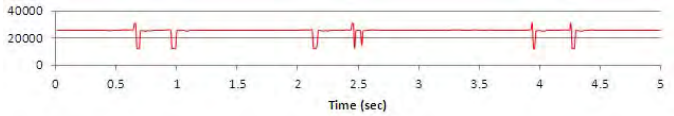
Fig. 2 shows the flow of our FSM-based gesture recognition method. It uses the angles of yaw, roll, and pitch to cancel g effect. One advantage is that our recognition and sensing processes can be performed in parallel, whereas many other recognition algorithms require all the data samples for the entire gesture interval to be matched against the entire database. Not only does our method speed up the recognition rate, but developers can also implement new gestures more easily based on the basic gestures that we recognize.

C. Tap to Pair

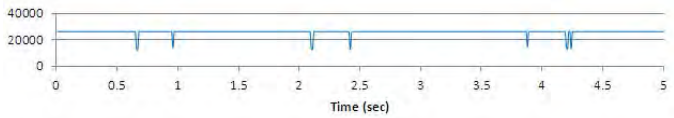
A gesture that requires special treatment is called a Tap, which is used when selecting the device to pair with. It differs from all other gestures in that the motion is sensed simultaneously by both the ring and the device being selected.



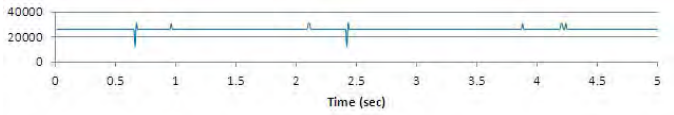
(a) Y-axis of MobiRing



(b) Z-axis of MobiRing



(c) Y-axis of Tapped Host



(d) Z-axis of Tapped Host

Fig. 5: The tapping Pattern of MobiRing and tapped host

The time-history plots of the Y, Z acceleration experienced on both the MobiRing and the tapped host are shown in Fig. 5. As can be seen, pairing is expressed as three taps in a row on the designated host, which is assumed to be stationary. We find that the variation of the difference between the times of detection by the two accelerometers is within 0.1 second.

D. Processing

We provide the options of performing data processing on either the ring or the host system. Due to resource constraints, our initial implementation performs most processing on the host. Ultimately, this may be negotiated dynamically between the MobiRing and the application, since some applications may want the raw data while others want processed data. Besides the computing capability of the MobiRing, another important consideration is the power: local computation can consume some power but can be worth it if it can significantly reduce the wireless communication (e.g., by transmitting the recognized letter as an ASCII code rather than all the data points along the trajectory). In the case of BLE version of MobiRing, the human interface device (HID) profile can make available both the mouse-motion stream and the recognized key codes, so that the application can select the modality at run time.

V. Communication Protocol

This section describes the wireless communication protocol that enables MobiRing and a target system to establish and terminate connection and to transfer bulk data.

A. Pairing

Pairing means logical association of a MobiRing with a host computing device such as a PC or a tablet. Our

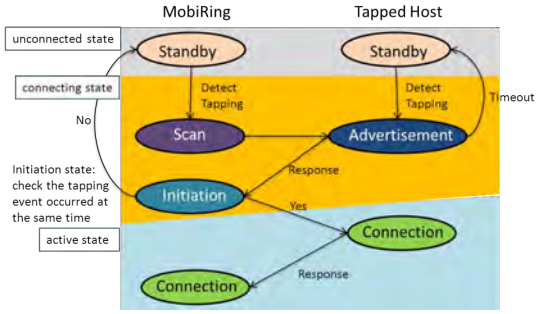


Fig. 6: The Procedure of the Communication Protocol

initial prototype has been built for the ShockBurst protocol by Nordic. It does not perform pairing per se but just relies on ID matching. To achieve the effect of pairing with different hosts, we add a session layer on top of the link layer in a way that is compatible with the concepts defined for BLE.

As shown in Fig. 6, upon powering-up, a MobiRing is initially unpaired. Upon tapping, the tapped host enters scanning mode while the tapping ring enters advertising mode. In BLE terminology, a scanner is a potential master looking for available slave to connect, while an advertiser is a slave looking for a master to pair with. The scanner and advertiser exchange basic information. The scanner then switches to initiation mode to connect with the advertiser. If successful, the initiator becomes master and the advertiser becomes slave.

In an implementation using the Enhanced ShockBurst protocol, all nodes can share a frequency channel for control (i.e., advertising, scanning, and initiating), and then they can switch to a mutually agreed-upon channel for data communication. The advertiser would broadcast its ID to a broadcast address, while a scanner (only the tapped one is possible) listens on the broadcast packet, grabs the ID, and replies to the advertiser with its own ID. Then, they both switch to the data channel.

In an implementation using BLE, the pairing process is specified by the BLE protocol in a similar but more general, secure way. An advertiser and a scanner work on three of the frequency channels in turn, and the scanner initiates the connection and becomes the master, whereas the scanner accepts the connection and becomes the slave. The two exchange additional information including security key and frequency-hopping sequence and work on a secure link.

B. Connection

To save power, one effective way is to suppress transmission when there is no motion. To do so, we take advantage of the threshold detection feature supported by the accelerometer hardware so that it can remain silent until the user's gesture causes acceleration to exceed the pre-programmed threshold. Then, the accelerometer generates an interrupt signal to notify the microcontroller unit (MCU) to start transmission of either raw data or processed data. The same mechanism also enables the MCU to sleep. After a period of inactivity, the ring can go to sleep mode, but it can also disconnect from the host until the user presses the button to wake it up.

(a) (b)

Fig. 7: (a) Prototype of MobiRing (b) System Prototype.

C. Reliable Transmission

Data transfer between the MobiRing and the receiver can be divided into key-event data and streaming data. Key-event data is the output after the MobiRing recognizes gestures and outputs key codes that encode them, similar to a keyboard's output. On the other hand, streaming data consists of raw or processed acceleration values without gesture recognition. The difference between them are that the key-event data is transmitted in a reliable way (i.e., with acknowledgment), whereas streaming data can be transmitted without ack. The chosen radio may support acknowledged or unacknowledged communication:

Protocol	ShockBurst	Enh. SB	BLE
Acknowledged		✓	✓
Unacknowledged	✓	✓	

To support reliable transmission on a transceiver such as Shockburst, we have implemented a simple protocol based on RMST [20]. Data objects are divided into small code segments, each of which contains at most 16 packets of data, and each packet is 25 bytes in size. For each segment, the receiver keeps a 16-bit bitmap to record the receipt of packets in a segment. Each bit is set to 1 when the corresponding packet is received. The packets are reassembled and checked if the segment has been received in its entirety. If the segment is not completed within a time interval, a negative acknowledgment (NAK) will be sent along with the 16-bit bitmap to the sender to request for retransmission of lost packets.

VI. System Prototype

We have prototyped MobiRing to demonstrate the feasibility of the idea. Fig. 7a shows one of the first implementations of MobiRing worn on a finger, and it can connect to a PC, smartphone, or tablet as shown in Fig. 7b. This section describes the hardware and software of the MobiRing, the base station, and the target computer in detail.

A. MobiRing

Three generations of the MobiRing have been prototyped. They represent the range of technologies that are all potentially applicable to the MobiRing concept. This subsection describes each and discusses their trade-offs.

The first was constructed using an ultra-compact wireless sensor platform named Eco [15]. It is 1 cm^3 in volume and under 2 grams in weight, including an analog triaxial accelerometer (Hitachi Metals H34C) with ± 3 g range, a Nordic nRF24E1 MCU with an integrated radio transceiver (ShockBurst) at 1 Mbps, 4 KB EEPROM, a chip antenna (Rainsun), and a 90 mAh Li-polymer rechargeable battery.

TABLE I: Supported Host-Transceiver Combinations

Host	Protocol	Transceiver
PC	ShockBurst and Enhanced SB	USB dongle, IP base station (Rev. A, B)
	BLE	USB dongle
iPhone, iPad	ShockBurst and Enhanced SB	MFi dongle, IP base station (Rev. C)
	BLE	built-in

The second generation uses an improved RF+MCU, the Nordic nRF24LE1 at 2 Mbps and 16 KB program flash, and a digital triaxial accelerometer (ST LIS331DLH) [7]. The improvements over the first generation include the doubled data rate, hardware-supported reliable communication (auto-ack and auto-retransmit features of the Enhanced ShockBurst) with hardware-supported, programmable threshold and free-fall detection by the accelerometer. This feature enables suppression of transmission during the absence of motion.

The third generation uses the TI CC2540 RF+MCU, which supports the full BLE master/slave protocol stack and has 256 KB of program flash. The BLE stack takes up slightly less than half of the firmware. It also uses the same digital accelerometer. Most importantly, the entire PCB is only 8×8 mm^2 in area including the chip antenna [23]. This makes it possible to construct a minimally intrusive MobiRing. More importantly, the use of HID profile means it can work without dongles or separate driver installation on systems with built-in BLE support.

B. Target Hosts

The target host can be a PC, a tablet, or a smartphone, as shown in Fig. 7b. For the first and second generations of MobiRing, we need to connect via either a USB dongle or an IP base station, due to the lack of built-in transceivers for ShockBurst and Enhanced ShockBurst protocols. The third generation is based on BLE, which is supported since iPad 3, iPhone 4S, Mac OS X computers since mid-2011, Android 4.3, Windows 8, and newer devices. However, for the purpose of this test on an older PC, we created a USB dongle for BLE compatibility. For older iPhones, we created an MFi (Made for iPod) dongle to work with these hosts. Although the iPad and iPhone do not make use of a mouse, pointing is still a useful feature. One scenario that is particularly useful is during text editing. It can be difficult to precisely position the editing cursor right at the right place between the letters, even with the help of the pop-up modifying-glass feature on the screen. The mouse movement is remapped to controlling the text-insertion blinking bar on these devices.

VII. Evaluation

We evaluated the usability of MobiRing quantitatively and qualitatively by asking ten volunteer novice users to perform a number of tasks. This section first describes the test cases, including both finger-gesture input and mouse control using

MobiRing, tap-to-switch pairing, and cut-and-paste across hosts. We report the assessment results by these users.

A. Experimental Setup

We configure the transceiver power level to 0 dBm and the accelerometer to collect 12-bit samples at 100 Hz in all three axes. We conducted a series of trials with ten volunteer novice users over a variety of the proposed operations. These users were mainly Computer Science majors, including three undergraduate and seven graduate students. None of them had prior knowledge or experience with gesture recognition systems or MobiRing. Each user received 10 minutes of tutorial and introduction to MobiRing prior to a series of trials individually, and we interviewed each user afterwards. The user are given the following MobiRing commands:

- with button released, tap to pair with a host PC.
- hold the thumb button to move the mouse pointer on screen by tilting (i.e., rotating and hold) the ring up, down, left, or right.
- release button, (gesture) to "grab, copy, cut" data/file
- tap to pair with another host PC.
- hold the thumb button to navigate to the desired folder
- release the button, (gesture) to "put, paste" data/file
- tap to pair with a DVD player that has been modified to work with MobiRing
- use gesture to navigate the DVD's menu and playback control.

We asked the user to perform the four tasks below:

Task 1: Do each gesture ten times: We recorded the accuracy of these gestures of each user. We demonstrated each gesture first and asked them to repeat each gesture for ten times. Due to the acceleration will be too small to detect, each gesture should be finished in 3 seconds as the lower bound.

Task 2: Mouse-like function for cursor control: We told these users to hold the button down on MobiRing to make it act as a wireless mouse, as shown in Fig. 8a.

Task 3: Tap to Pair: We asked the user to pair with a DVD player augmented with our RF module as shown in Fig. 8b and control the cursor of the player.

Task 4: Drag-and-Drop Data from one PC to another: The process of this task is shown in Fig. 8c-8e. The user first pressed the button on the MobiRing to make it act as a wireless mouse and tilted it to control the direction of the cursor. After choosing the file, the user made an gesture of "Change Host" to change the control to another computer. Subsequently, the user made a "Paste" gesture to transport the file to the destination system.

B. Results

The experimental results from these four tasks are evaluated quantitatively and qualitatively for the usability. We recorded the experimental results and report the statistics. We also asked these users individually for their feedback.

TABLE II: The Average Accuracy of Each Gesture Recognition

Action	Gesture	Avg Accuracy
left click		93%
right click		95%
double click		92%
back		95%

Action	Gesture	Avg Accuracy
close window		90%
paste		91%
grab		92%
change host		92%

1) Recognition Accuracy: The result of Task 1 is presented in Table II. As can be seen, all of the gestures can be recognized correctly and the accuracy of each gesture recognition is over 90%.

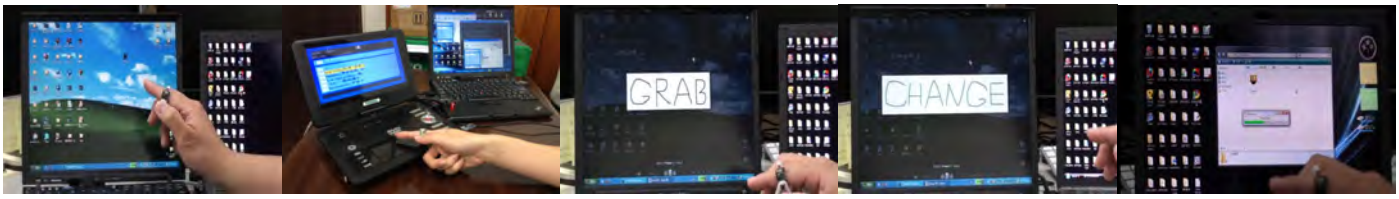
2) Convenience and Intuition: The results of Task 2 and Task 3 are satisfactory. In Task 2, every user could quickly pair with the DVD player and start controlling it by gestures such as power on, next page or volume control. In Task 3, even though we did not tell them that the tilting direction determined the movement of the cursor on the computer, they could control it well after minimal trial and error. This validates that our MobiRing can be intuitive enough for novice users. In Task 4, we recorded the time spent on the transporting process (excluding the file transmission time). Fig. 9 shows the time spent by each user. A great majority of students could finish it within 10 seconds.

3) Feedback From Users: After these experiments, we collected feedback from these users on their experience with using our MobiRing. The vast majority said MobiRing was an intuitive way for pointing and gesturing, and that it was convenient for transporting data among electronic devices. One of the users commented that the mouse-like function was more convenient than using the trackpad on the laptop.

Our next question was what they saw the potential utility of MobiRing for daily life might be. One said it could be convenient to use gesture recognition to control smartphones while driving. One of the users skilled at 3D modeling said MobiRing may make a useful input device for working with 3D models such as Sketchup or navigating in Google Earth. Another found MobiRing to be convenient while his smartphone was far away from him. One user suggested the use of two MobiRings, one on the index finger of each hand, to enable multi-touch-like operations similar to that on smartphones but without surface contact.

VIII. Conclusions

MobiRing provides a direct, intuitive way to handle data between electronic systems. Cut-and-paste and drag-and-drop operations are intuitive ideas users come to expect when dealing with electronic systems, and MobiRing brings this idea from the virtual world into the real world. This can be very convenient especially where graphical user interface is not available. Gesture recognition and reliable transmission issues are vital to the operation of the proposed MobiRing system. Improving gesture recognition accuracy and lowering retransmission rate are two important design tasks to ensure smooth operation of the MobiRing. MobiRing is shown to be



(a) (b) (c) (d) (e)

Fig. 8: (a) Using "tilt" to control the cursor of the laptop. (b) Using "Tap" to select the device to pair with. (c)-(e) Data transporting between two computers: (c) Grab the file on the source computer (d) Change the control to the destination computer (e) Paste the file to the destination computer

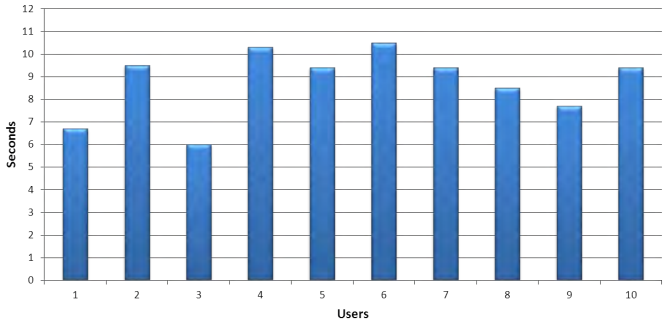


Fig. 9: The Time Spent on Transporting Data of Each User

the most direct way for users to transfer any type of data between computers.

Several directions for future work remain. One is to embed compatible wireless interfaces into household appliances for not only grabbing data snapshots such as sensor readings but also establishing live links among them. Another is to explore multi-touch-like dual-finger operation and other richer gestures, which will further enhance the expressiveness of the MobiRing. As a programmer platform, MobiRing can be programmed for many other purposes beyond our original vision. It is our goal to open up the MobiRing hardware and software platform to developers in general to make possible even more uses of this novel platform.

REFERENCES

[1] Bump. http://www.vrealities.com/pinch.html.

[2] P5 glove. https://bu.mp/.

[3] Pinch glove. http://www.vrealities.com/pinch.html.

[4] W. Bang, W. Chang, K. Kang, E. Choi, A. Potanin, and D. Kim. Self-contained Spatial Input Device for Wearable Computers. In *Proceedings of the 7th IEEE International Symposium on Wearable Computers*, page 26. IEEE Computer Society, 2003.

[5] A. Benbasat and J. Paradiso. Compact, configurable inertial gesture recognition. In *CHI '01: CHI '01 extended abstracts on Human factors in computing systems*, pages 183–184, 2001.

[6] A. Benbasat and J. Paradiso. An inertial measurement framework for gesture recognition and applications. In *GW '01: Revised Papers from the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction*, pages 9–20, London, UK, 2002. Springer-Verlag.

[7] C.-Y. Chen, Y.-T. Chen, Y.-H. Tu, S.-Y. Yang, and P. H. Chou. EcoSpire: An application development kit for an ultra-compact wireless sensing system. *IEEE Embedded Systems Letters*, 1(3):73–76, October 2009.

[8] F. G. Hofmann, P. Heyer, and G. Hommel. Velocity profile based recognition of dynamic gestures with discrete Hidden Markov Models. In *Proceedings of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction*, pages 81–95, September 1998.

[9] S. Kallio, J. Kela, and J. Mäntyjärvi. Online gesture recognition system for mobile interaction. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 3, pages 2070–2076, Washington, D.C., USA, 2003.

[10] J. Kela, P. Korpipää, J. Mäntyjärvi, S. Kallio, G. Savino, L. Jozzo, and S. Di Marca. Accelerometer-based gesture control for a design environment. *Personal Ubiquitous Comput.*, 10(5):285–299, 2006.

[11] LogBar, Inc. Ring: Shortcut everything. http://logbar.jp/ring/, 2014.

[12] J. Mäntyjärvi, J. Kela, P. Korpipää, and S. Kallio. Enabling fast and effortless customisation in accelerometer based gesture interaction. In *MUM '04: Proceedings of the 3rd international conference on Mobile and ubiquitous multimedia*, pages 25–31, New York, NY, USA, 2004. ACM.

[13] V.-M. Mäntylä, J. Mäntyjärvi, T. Seppänen, and E. Tuulari. Hand gesture recognition of a mobile device user. *Multimedia and Expo, 2000. ICME 2000. 2000 IEEE International Conference on*, 1:281–284, 2000.

[14] Nod, Inc. Wave hello to Nod. https://www.hellonod.com/, 2014.

[15] C. Park and P. H. Chou. Eco: Ultra-wearable and expandable wireless sensor platform. In *Proc. Third International Workshop on Body Sensor Networks*, pages 162–165, April 2006.

[16] J. K. Perng, B. Fisher, S. Hollar, and K. S. J. Pister. Acceleration sensing glove (asg). In *Proc. of the IEEE International Symposium on Wearable Computers*, pages 178–180, 1999.

[17] L. R. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[18] RHLvision Technologies Pvt. Ltd. Fin: Wearable ring make your palm as numeric keypad and gesture interface. http://www.wearfin.com/, 2014.

[19] H. Sawada and S. Hashimoto. Gesture recognition using an accelerometer sensor and its application to musical performance control. In *Electronics and Communications in Japan Part 3*, pages 9–17, October 1997.

[20] F. Stann and J. Heidemann. RMST: Reliable data transport in sensor networks. In *IEEE Workshop on Sensor Net Protocols and Applications (SNPA)*, May 2003.

[21] B. Tomlinson and L. Carpenter. The EcoRaft project. http://orchid.calit2.uci.edu/EcoRaft/.

[22] K. Tsukada and M. Yasumura. Ubi-Finger: Gesture input device for mobile use. In *Proceedings of APCHI*, volume 1, pages 388–400, 2002.

[23] A. Wang, Y.-T. Huang, C.-T. Lee, H.-P. Hsu, and P. H. Chou. EcoBT: Miniature, versatile mote platform based on Bluetooth Low Energy Technology. In *Proceedings of the IEEE International Conference on Green Computing and Communications*, Taipei, Taiwan, September 1-3 2014.