A Framework for Bimanual Inter-Device Interactions

Ali Roudaki  Jun Kong  Gursimran Walia
Department of Computer Science
North Dakota State University
Fargo, USA
{ali.roudaki, jun.kong, gurisimran.walia}@ndsu.edu

Abstract—A shared interactive display (e.g., a tabletop) provides a large space for collaborative interactions. However, a public display lacks a private space for accessing sensitive information. On the other hand, a mobile device offers a private display and a variety of modalities for personal applications, but it is limited by a small screen. We have developed a framework that supports fluid and seamless interactions among a tabletop and multiple mobile devices. This framework can continuously track each user’s action (e.g., hand movements or gestures) on top of a tabletop and then automatically generate a unique personal interface on an associated mobile device. This type of inter-device interactions integrates a collaborative workspace (i.e., a tabletop) and a private area (i.e., a mobile device) with multimodal feedback. To support this interaction style, an event-driven architecture is applied to implement the framework on the Microsoft PixelSense tabletop. This framework hides the details of user tracking and inter-device communications. Thus, interface designers can focus on the development of domain-specific interactions by mapping user’s actions on a tabletop to a personal interface on his/her mobile device. A user study compared our interaction style with a standard tabletop interface and justified the usability of the proposed interaction.

Keywords—bimanual interaction; multimodal interface; tangible interface; human computer interaction

I. INTRODUCTION

The benefits of tabletop-based applications have been investigated in different scenarios, such as collaboration [3] or pedestrian navigation [21]. However, it is hard to protect personal information on a tabletop. In addition, a public environment limits the usage of some modalities. For example, the usability of auditory feedback through a speaker is reduced in a noisy public environment. While a mobile device provides diverse multimodal feedback, it is limited by its small screen which makes it frustrating to browse a large amount of information.

Therefore, a synergistic interaction with mobiles and tabletops integrates their merits. While previous studies have explored the benefits of combining mobile devices and a large display together (e.g., a fluent switch between individual and group work [32]), one challenge in inter-device interactions is to minimize distractions when switching between devices. Another challenge is to develop a generic solution that is suitable for various scenarios and applications. Although different techniques have been developed to support interactions in a multi-device ecology [10, 19], few researchers have focused on developing a generic platform that supports a variety of applications. Recently, based on PhoneTouch [29], Schmidt et al. [31] developed a generic platform supporting a novel interaction style that fits different applications. This interaction style is featured by pairing a phone touch event with the identity of the phone through an accelerometer.

In contrast to the PhoneTouch interaction style [31], our generic framework (hereinafter referred to as MobiSurf) supports a bimanual interaction style using a tangible object. Our approach uses a passive tangible object to perform coarse-grained selections on a tabletop while a mobile device is held by the dominant hand for fine-grained interactions. Based on the previous studies on the tangible interfaces [13, 34], we proposed various gestures (e.g., “Pointer Rotate”, “Pointer Move”, and “Pointer Share”) that enable a natural interaction with a tabletop device. The gesture of each user on the tabletop is detected and accordingly produces a unique personal interface with multimodal feedback on the associated mobile device. The MobiSurf framework is featured by a thin client on the mobile device, which is application-independent. In other words, during the development process, the client side, which encapsulates the functions of the interface generation and inter-device communications, is generic for different applications and does not need any revision. Such an implementation allows developers to focus only on application-specific developments by translating user gestures on a tabletop to appropriate messages on an associated mobile device (See Section III). To evaluate the usability of the proposed interaction style, we conducted a controlled empirical study. Participants were asked to complete two tasks using both a standard tabletop interface and the MobiSurf interface. The results from this study showed significant improvements in the usability when using the MobiSurf interaction style as compared to the standard interface.

II. RELATED WORK

Researchers have explored the combination of mobile devices and tabletops to improve the usability for collaborative tasks, such as augmenting a computer with PDAs in the single display groupware [22] or exchanging information between a personal device and a public display [9]. This section reviews different techniques for user tracking and inter-device interactions.

A. User Identification and Tracking

Because a tabletop represents a public space, it is necessary to identify the users to protect their personal data. Various

This work is in part supported by NSF under grant #CNS-1126570.
approaches have been proposed to pair a user’s interactions on a tabletop with a unique ID, such as hand biometrics [30], utilizing the back of a user’s hand as an identifier [26], infrared light pulses through a ring-like device [27], or using a tangible interface to authenticate users [34]. However, these approaches did not support direct data sharing between a public device and a personal one. Since most tabletop/mobile devices are equipped with a camera, the computer vision technique has been commonly used to associate a mobile device with a large display. For example, BlueTable [35] implemented a vision-based handshaking procedure by blinking an infrared light or flashing the display of a mobile device to establish a connection between a mobile device and a tabletop. Similarly, Schoning et al. [28] used the flashlight and Bluetooth unit of a mobile phone as a response channel to authenticate users. Ackad et al. [1] used the color detection to implement a handshaking protocol to identify a registered mobile device placed above a tabletop. This system also used a depth camera so that a user can continuously be tracked even if the device was removed from the tabletop. In addition, cameras were used in inter-device communications to replace radio-based techniques (e.g., WiFi or Bluetooth) for information exchange, such as FlashLight [11] or C-Blink [20].

Instead of using the computer vision technique, various approaches used gestures to associate a mobile device with a public display based on built-in sensors. Tilt correlation [12] compared the touch-derived tilt angle on a public display with the tilt sensor information from a mobile device to distinguish different mobile devices. Patel et al. [24] proposed a gesture-based authentication by shaking a device according to a required pattern. PhoneTouch [29] correlated the phone touch events detected by an interactive surface and by a mobile phone through an accelerometer to identify multiple mobile devices. The above sensor-based approaches used a mobile device for both the user identification and interaction. Consequently, a user cannot interact with the mobile device during the identification process. Instead, our approach introduces a tangible object for the user tracking so that the user tracking and interaction can be performed simultaneously.

Interacting with the computers using tangible objects has been widely studied. The pioneering work by Ishii and Ullmer [13] bridged the gap between a physical environment and a cyberspace through a tangible interface. Our approach uses a tangible object for tracing users’ actions on a tabletop.

B. Inter-Device Interaction

A mobile device provided gesture-based interactions and multimodal feedback; thereby making it suitable for being a remote controller for inter-device interactions [18, 19]. Several approaches leveraged a built-in camera to manipulate a remote object by directly touching or moving a mobile device, such as point & shoot for remote selection [2], camera-based pose estimation for remote operation [25], a privacy-respectful input method [16], snap and grab for sharing contextual multimedia contents [17], and touch projector for interacting with surrounding displays [4]. Alternatively, by using the accelerometer, movement-based gestures were developed for interacting with a distant display [6, 33].

Instead of remotely manipulating visual objects in a distant display through a mobile device, some approaches required a direct contact between a mobile device and a public display for inter-device interactions, such as placing a mobile device above a tabletop during the entire duration of interactions [7, 23], or freely moving a mobile device on top of a public display. Hardy and Rukzio [10] used an NFC mobile device as a stylus for interacting with an NFC-tagged display.

Although the above approaches supported mobile-tabletop interactions, they were not generic for supporting a variety of scenarios. Recently, Schmidt et al. [31] developed a generic platform for the synergistic usage of mobile devices and tabletops. Built on the PhoneTouch [29] technique, this platform used a mobile device as a stylus to select objects on a tabletop and analyzed touch events to recognize the identity of a mobile phone. Our framework is different with the above approach from the following perspectives. First, our approach implements a bimanual interaction, in which the non-dominant hand performed a coarse-grained selection through a tangible object while the dominant hand held the mobile device for a fine-grained interaction. According to Buxton et al. [5], the bimanual input outperformed the one-handed input for selection, positioning, and navigation tasks. Furthermore, the bimanual interaction is capable of tracking the path of hand’s movements on the tabletop while accordingly producing continuous feedback to the mobile device. Secondly, our framework implements a thin client which is suitable for different applications without modification. The implementation of a thin client allowed developers to focus on mapping user’s actions on the tabletop to interaction commands on the mobile device, while the framework itself can automatically produce a personal interface on the mobile device according to the mapping.

III. SYSTEM DESIGN AND ARCHITECTURE

A. System Design

The MobiSurf framework was built on three types of hardware components, i.e., a tabletop device, passive tangible objects (pointers), and mobile devices. Without losing generality, we implemented our framework on the Microsoft PixelSense tabletop which supported multi-touch interactions and was equipped with infrared sensors. Each passive tangible object served as a pointer to make a course-grained selection and was associated with a distinct mobile device for passing the interaction events from the tabletop to the associated mobile device, which was used as a personal area for accessing sensitive information with multimodal feedback. The pointer can be constructed with various shapes and different materials based on the users’ needs.

B. User Tracking

In a collaborative environment with multiple users, it is necessary to identify and track user’s actions to provide the personalized information and protect privacy. Mobile devices have been used to identify and track a user, such as PhoneTouch [29] or Tilt correlation [12]. However, the above approaches constrained the usage of the mobile device for interactions during the process of user identification and tracking. In order to overcome the above limitation, MobiSurf
implemented bimanual interactions based on a tangible interface. More specifically, each tangible object (i.e., a pointer) has a unique ID defined by an infrared tag which consists of a geometric arrangement of infrared reflective and absorbing areas. In the beginning, a user needs to type in the ID of a pointer on a mobile device to pair the pointer with the mobile device. The pairing process is only performed once at the beginning of an interaction session. When a particular user terminates the connection with the tabletop, the pointer is automatically released and can be used by another user.

C. Software Design

MobiSurf applies an event-driven architecture and serves as a middleware to set up two-way communications between a tabletop and a mobile device. On one side, the MobiSurf API accepts tabletop UI events, produces a set of commands, and forwards them to a mobile device. Those commands specify the actions performed on the mobile device (such as displaying a text box or generating a vibration). On the other hand, the MobiSurf API receives responses from the mobile device and accordingly notifies the tabletop application. Based on the above communication mechanism, the mobile application has two functionalities. First, based on the received messages, it either performs proper actions or renders proper UI elements on the mobile device. Second, it generates responding messages based on the user’s actions on the mobile device and sends them back to the MobiSurf API. Such an event-driven design results in a thin client which makes the mobile side application-independent. Because MobiSurf is completely compatible with standard UI elements, developers can extend a standard tabletop interface with the feature of inter-device bimanual interactions by defining the semantics of user actions on a tabletop. In summary, MobiSurf hides the details of user identification and inter-device communications and minimizes the coupling between the mobile device and tabletop through an event-driven architecture.

D. Event Handling

Different events and messages are supported in MobiSurf.

User Connection and Disconnection. Once a user requests to connect to the MobiSurf API, the API raises a “NewClient” event and sets up a virtual connection with the mobile device through TCP/IP. Pairing a pointer with the mobile device is also completed as a part of the user connection. When a user terminates the connection, the “ClientRemoved” event is raised, which releases the pairing between the pointer and the mobile client.

Actions on the Tabletop. After the connection and pairing steps, the MobiSurf API continuously tracks user’s actions on the tabletop through the paired pointer and raises corresponding internal events, i.e., “PointerOver”, “PointerRotated”, “PointerShare”, and “PointerMove”. For instance, PointerShare is triggered when two or more pointers are placed in proximity. This event is designed for content sharing among mobile devices. Table I summarizes all internal events handled by the MobiSurf API.

Table I. MobiSurf API Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewClient</td>
<td>New mobile client connected</td>
</tr>
<tr>
<td>ClientRemoved</td>
<td>Mobile client disconnected</td>
</tr>
<tr>
<td>MessageReceived</td>
<td>Acknowledgement from the mobile device</td>
</tr>
<tr>
<td>PointerOver</td>
<td>Pointer is over a UI Element</td>
</tr>
<tr>
<td>PointerRotate</td>
<td>Pointer twisted over a UI Element</td>
</tr>
<tr>
<td>PointerShare</td>
<td>Multiple pointers are in proximity</td>
</tr>
<tr>
<td>PointerMove</td>
<td>Continuous movements</td>
</tr>
</tbody>
</table>

Mobile Interaction. Based on an internal event, MobiSurf allows interface developers to specify proper action messages (Table II) and sends them to the paired mobile client. For example, when a user moves his/her pointer to a text box that invites a password, MobiSurf triggers the “PointerOver” event. Based on the developer’s specification, an action message is sent to the paired mobile client, which produces a text field on the mobile device for inputting the password. Therefore, the user can apply the mobile device as a private channel to input a password with improved privacy. In summary, the actions on the mobile device are classified into two groups. The first-group actions can dynamically generate UI elements on the mobile device, and the second-group ones can produce various multimodal feedback. Any combination of actions in Table II can be defined and sent to the mobile device.

Table II. MobiSurf Actions

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lighting</td>
<td>Flash the LED light</td>
</tr>
<tr>
<td>Vibrate</td>
<td>Vibrate the mobile device</td>
</tr>
<tr>
<td>Beep</td>
<td>Generate a beep sound</td>
</tr>
<tr>
<td>TextMode</td>
<td>Display a textbox on the mobile device</td>
</tr>
<tr>
<td>ListMode</td>
<td>Display a listbox on the mobile device</td>
</tr>
<tr>
<td>Button</td>
<td>Display a button on the mobile device</td>
</tr>
<tr>
<td>WebLink</td>
<td>Open a web page on the mobile device</td>
</tr>
<tr>
<td>Text</td>
<td>Display textual contents on the mobile device</td>
</tr>
<tr>
<td>Speech</td>
<td>Speak a text on mobile device</td>
</tr>
<tr>
<td>Image</td>
<td>Show an image on the mobile device</td>
</tr>
<tr>
<td>Media</td>
<td>Play a voice or video file</td>
</tr>
<tr>
<td>AlertDialogue</td>
<td>Show a text alert message on screen</td>
</tr>
<tr>
<td>DataRequest</td>
<td>Request data stored on a mobile device</td>
</tr>
</tbody>
</table>

Acknowledgement. After the mobile client completes the required interaction, it sends an acknowledgement to the MobiSurf API. Accordingly, MobiSurf raises the “MessageReceived” event which includes the actions completed on the mobile device (such as free-style typing or selection from a list) and the information being input along with the user ID. The user’s actions include “ButtonClick” (i.e., the user tapped a button on the mobile device), “ItemSelect” (i.e., the user selected an item from a list box on the mobile device) and “TextChanged” (i.e., the user inputted texts to a text box on the mobile device).

Users’ gestures on a tabletop in general have a metaphoric basis. However, due to different application domains and users’ backgrounds, the same gesture may intend to different commands under different interaction scenarios. Therefore, MobiSurf supports an open framework, which provides the flexibility for interface developers to determine the action of a
gesture on a tabletop by mapping the tabletop-based gesture (See Table I) to commands on a mobile device (See Table II).

IV. MobiSurf Interaction Style

Schmidt et al. [31] summarized six challenging issues in the use of multi-touch tabletops. We introduce the MobiSurf interaction style to address those issues.

A. Data Transfer

Users can transfer data from a tabletop application to a mobile device and vice versa. For example, a user can store his/her bookmark on the mobile device and then copy it to a tabletop application. In addition, multiple users (i.e., two or more users) in the same group can share information.

Tabletop to Mobile. A user first moves his/her pointer to a tabletop UI element that includes the information of interest; then, he/she twists the pointer which triggers the “PointerRotate” event. By mapping the “PointerRotate” event on the above UI element (e.g., a text field) to a specific action (i.e., the Text action) on the mobile device, the requested information is transferred to the mobile device.

Mobile to Tabletop. Automatically transferring personal information from a mobile device to a tabletop can facilitate data entry and avoid redundant inputs. A user places the pointer over a UI element that invites inputs from the user, which triggers the “PointerOver” event. A message including the “DataRequest” action is sent to the mobile device. The user can choose the corresponding information on his/her mobile device for sharing with the tabletop.

Mobile to Mobile sharing. A user (i.e., the source) selects the information being shared (i.e., image or text) on his/her mobile phone and moves his/her pointer in proximity to the pointer of the target user. The above action triggers the “PointerShare” event on a tabletop. Based on the type of information being shared, the tabletop application sends proper action messages (i.e., “Text”, “Image”, or “Media”) to the target mobile device. In order to avoid an accidental sharing, the “PointerShare” event can also produce an “AlertDialogue” action on the source device to confirm the sharing. The information sharing can be easily extended to three or more users by placing their pointers in proximity.

B. Personalization

A mobile device is ideal to supplement tabletop interactions with personal information, such as copying personal information from a mobile device to a tabletop to automatically fill in a form or defining a personal area on a public display.

Auto Fill. Auto fill avoids redundant data entry by automatically copying personal data from a paired mobile device to the tabletop, which is essentially implemented as data transfer from a mobile device to a tabletop.

Region Selection. The PhoneTouch based approach [31] only tracks separate phone touch events, while MobiSurf supports tracking the continuous movement of a pointer that corresponds to the “PointerMove” event. A tabletop application can record a moving path that includes a sequence of coordinate data from the “PointerMove” events. The information of a moving path is useful in various scenarios. For example, we can define a personal area according to the moving path (i.e., a close area where the first point in the moving path is identical to the last one). The personalized information or adaptation (based on the mobile-to-tabletop sharing) can then be applied to this personal area.

C. User Interface Composition

In a collaborative environment, moving interaction commands (such as a menu) from a tabletop to a mobile device can utilize the screen more efficiently and allow multiple users to simultaneously operate commands displayed in proximity. Based on the location of a pointer on the tabletop, MobiSurf can use the mobile device as a tangible controller by displaying contextual menus.

Expanded Screen. During the interaction on a tabletop, a user’s mobile phone displays contextual menu items in response to the user’s action. For example, a user moves his/her pointer to an image on the tabletop which triggers the “PointerOver” event. Then, appropriate contextual menu items (e.g., save, edit, email) are sent to the mobile device through corresponding action messages (e.g., Button or ListMode).

UI elements to Phone. A user can move his/her pointer over a UI element and rotate the pointer (i.e., the PointerRotate) to transfer the UI element along with its content to his/her mobile phone (i.e., TextMode or ListMode). A user can manipulate the UI element on his/her mobile device (i.e., changing the content) and send it back to the tabletop.

D. Authentication

Authentication on shared interfaces has always been challenging [15]. MobiSurf uses the mobile device as a private channel to authenticate the identity of the user.

Password entry. Traditional username-password based authentication is still popular in many existing systems. However, it is not secure to input a password or other sensitive information on a shared display. By mapping a “PointerOver” event on a UI element (e.g., a text field for a password) to a “TextMode” action on the mobile device, the MobiSurf interaction style uses a mobile device as a personal device to input sensitive information. Once a user moves his/her pointer over the username/password entry panel, the login interface will be generated on his/her mobile device to protect privacy.

Advanced Authentication. Some applications that require advanced security need both the username/password and hardware authentication. Since a mobile device is a personal device, its International Mobile Equipment Identity (i.e., IMEI) can be used for the hardware authentication. Based on MobiSurf, an advanced authentication can be implemented by sending both the TextMode (i.e., inputting the password) and DataRequest (i.e., sending the IMEI information) actions to a mobile device.

E. Localized and Private Feedback

Since a mobile device supports different output modalities (e.g., vibration, flash or sound), it is ideal to produce personalized and private feedback.

Multimodal Feedback. MobiSurf supports various types of feedback on the mobile device, such as a flashing light,
beep, vibration, or speech. For example, when a user twists his/her pointer over a text block on the tabletop, the text-to-speech service can be utilized (based on the “Speech” action) through a speaker or earphone on the mobile device. With an open event-driven framework, any combination of output modalities can be defined through action messages.

**Accessibility.** Researchers have investigated on improving the accessibility of a public interface for blind users [14]. MobiSurf enables visual-impaired users to interact with a tabletop by providing personalized interactions and feedback through a mobile device. More specifically, MobiSurf tracks user’s actions on a tabletop (e.g., “PointerOver” or “PointerMove” events) and accordingly generates vibration and voice feedback (through the “Vibrate” and “Speech” actions) to guide the user to access various parts in an interface.

**F. Input Expressiveness**

In addition to the traditional interaction methods, MobiSurf exploits tangible objects as an additional input method.

**Movement Tracking.** The “PointerMove” event supports the movement-based gesture to control a tabletop application, such as a region selection or accessibility.

**Multiple Pointers.** MobiSurf supports pointers with different shapes as long as a proper infrared tag is attached to each pointer. Furthermore, multiple pointers can be registered to a single user. The shape, the number of tags associated with a user, and gestures performed on a pointer define a design space for tangible interactions. For example, in a chess game application, each player has 16 pointers with different shapes and the movement of a pointer produces the corresponding feedback on both the tabletop and the mobile device.

In summary, MobiSurf seamlessly integrates multiple mobile devices with a tabletop and is potentially useful in different applications. For example, MobiSurf can provide a user-friendly interface for both visually impaired users and normal users. More specifically, vibration and speech feedback on a mobile device allows disabled users to access information while normal users read visual information on a tabletop.

**V. EVALUATION**

A controlled empirical study was conducted to investigate the user experience.

**A. Research Hypotheses**

The following hypotheses were formulated for this study:

H1: MobiSurf interface is better than a standard tabletop interface in terms of effectiveness, ease of use, user satisfaction, privacy, and comfort.

H2: MobiSurf interface is at least as easy to learn as the standard tabletop interface.

H3: There are no distractions when switching between the mobile and tabletop devices for the MobiSurf interface.

**B. Participating Subjects and Apparatus**

Forty-four undergraduate students participated in the study. None of the participants had any prior experience with tabletop devices. The study design required each participant to perform two tasks (i.e., user authentication and content sharing) using both the standard tabletop interface and the MobiSurf interface. A complete counterbalancing of the order of the tasks and the interfaces was performed to avoid the pitfalls of the “carryover” effect in a standard repeated measures design.

MobiSurf can be implemented on any kind of mobile devices. Without losing generality, a prototype was implemented on a Microsoft PixelSense Samsung SUR40 tabletop and Android mobile devices.

**C. Experiment Design**

The study began with a pre-study questionnaire followed by a training session on the standard tabletop and MobiSurf interfaces. Next, the participants were asked to perform two tasks, using each interface respectively. The experimenter recorded the completion time of each subject for each task on each method. After performing each task, the participants filled out a post-study questionnaire to provide feedback on their reaction to each interface.

Step 1 – Prestudy Questionnaire: During this step, we collected information from the participants about their reading skills, the prior knowledge of touch screen interfaces, the experience (i.e., whether they own a smartphone or not) and the comfort level with smartphones.

Step 2 – Training Session: The experimenter trained the participants on using a standard tabletop interface as well as a MobiSurf interface. The training session included the general description of both interfaces. Then, participants practiced a text entry task and a content sharing task, designed only for the training purpose, using both interfaces.

Step 3 – Performing Tasks: Researchers have designed and evaluated the authentication and content sharing tasks on tabletop devices [15, 17, 24, 25, 28]. These tasks indicated the common interaction applications on tabletops; thus, they were selected in our study to compare the usability of the MobiSurf interface against the standard tabletop interface. The participants were not allowed to ask for any help during the study. They performed the tasks based on written instructions to accomplish each task.

Task #1- For this task, we provided a standard login interface and assigned each subject with a random username-password combination that was of the same length for all the subjects.

Task #2- Content Sharing. In this task, each user downloaded a company’s mission statement, revised it, and finally sent it back to a coordinator. In a standard tabletop interface, email was used to transfer information between different devices. Each participant was asked to email a text block from the tabletop to a designated email address. Then, the participant opened the email on his/her mobile device to download the mission statement, edited it, and emailed the revised version to the experimenter’s email. In order to make a fair comparison, in the tabletop-only interaction, a user clicks a button to open an email interface, which was embedded and displayed in the same tabletop application, for transferring data. Alternatively, the MobiSurf interface used gestures to support the email function. First, each participant was asked to use the...
PointerRotate gesture, which mimics the disk rotation for data exchange, to transfer the mission statement from a tabletop to his/her mobile device. After revising the statement, the updated version could be shared with the coordinator through the PointerShare gesture.

Step 4 – Survey Questionnaire: At the end of each task using each interface, participants were asked to fill an online survey questionnaire to rate his/her experience on using the interface.

**D. Analysis and Results**

An alpha value of 0.05 is selected to judge the significance of the results.

**H1 – Usability of MobiSurf and Tabletop:** To compare the usability of the treatments, each interface for each task was rated using a 5-point scale on five relevant characteristics, i.e., effectiveness, ease of use, user satisfaction, privacy, and comfort. Using the individual score, we calculated the median score for each treatment method on each characteristic separately for the user authentication task (See Table III) and the content sharing task (See Table IV).

<table>
<thead>
<tr>
<th>TABLE III. RATINGS ON FIVE CHARACTERISTICS FOR TASK 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1 – User Authentication</td>
</tr>
<tr>
<td>Effectiveness</td>
</tr>
<tr>
<td>MobiSurf</td>
</tr>
<tr>
<td>Tabletop</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE IV. RATINGS ON FIVE CHARACTERISTICS FOR TASK 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 2 – Content Sharing</td>
</tr>
<tr>
<td>Effectiveness</td>
</tr>
<tr>
<td>MobiSurf</td>
</tr>
<tr>
<td>Tabletop</td>
</tr>
</tbody>
</table>

In addition to the noticeable positive ratings for the MobiSurf interface, we also noticed the differences between the ratings on the two tasks. Specifically, the difference on the privacy characteristic between the two interfaces on task 1 is larger than that on task 2, which can be attributed to the nature of the tasks. That is, task 1 (user authentication) is more privacy concentrated compared with task 2 (content sharing) which implies that the MobiSurf interface is preferred by users with the privacy needs. In addition, we noticed that for all five characteristics, the median ratings for the MobiSurf interface were significantly higher than that for the tabletop interface. A Paired-sample Wilcoxon Signed-Rank test on each pair of rating values revealed significant differences between two methods for each characteristic (i.e., p<0.001)

**H2 – Ease of learning on MobiSurf and tabletop interfaces:** MobiSurf introduces a new interaction style, which should not significantly increase the learning time. A Paired-sample Wilcoxon Signed-Rank test compared the median ratings for each method on each task. As presented in Table V, in task 1, the results showed that the participants rated the ease of learning on both interfaces equally positive, and the difference was not statistically significant (i.e., p=0.323). In task 2, the participants rated the MobiSurf interface significantly easier to learn than the tabletop interface (i.e., p=0.012). These results verify our hypothesis that the MobiSurf interface is at least as easy to learn as the standard tabletop interface.

<table>
<thead>
<tr>
<th>TABLE V. RATINGS ON EASE OF LEARNING</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ease of learning</td>
</tr>
<tr>
<td>MobiSurf</td>
</tr>
<tr>
<td>Tabletop</td>
</tr>
</tbody>
</table>

**H3 – Distraction:** The MobiSurf interaction style requires a user to switch his/her focus between a mobile device and a tabletop. In the study, we specifically asked participants to rate the level of distraction they felt due to the device switch during an interaction session. In Table VI, which presents the median score, score 5 indicates “the user strongly agrees that the combination of a mobile device and a tabletop does NOT distract his/her attention while performing a task”. Based on the participants’ ratings, 39 of the 44 participants for tasks 1 and 36 participants for task 2 rated “agree” (with a score of 4) or “strongly agree” (with a score of 5) that they did not feel distraction using the MobiSurf interface. This result verified our hypothesis that the switching between devices in the MobiSurf interaction style does not distract users.

<table>
<thead>
<tr>
<th>TABLE VI. RATINGS ON USER DISTRACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Distraction</td>
</tr>
<tr>
<td>MobiSurf</td>
</tr>
</tbody>
</table>

Fig. 1 provides an overview of participants’ efficiency results. It shows the mean and standard deviation (SD) of the task completion time for each task and interface combination. For both tasks, the SD values for both interfaces are relatively small and indicate consistency across the data. Regarding task 1, the subjects spent an average of 21 seconds using the tabletop interface vs. an average of 24 seconds using MobiSurf. Based on the researcher’s observations during the user study, the device switch in MobiSurf may result in longer time to complete task 1. Regarding task 2, there is a visible difference and significant improvement in the task completion time when using the MobiSurf interface (an average of 33 seconds) vs. tabletop interface (an average of 115 seconds).

![Figure 1. The mean values and SD of task completion time](image)

We then used multiple regression [8] to find any statistically significant correlations between their background data and their efficiency when using the MobiSurf and tabletop interfaces. The results showed that the subjects’ experience of using smartphones (i.e., owning a smartphone) had a significant correlation to the efficiency using the MobiSurf
interface for both tasks 1 (p=0.0098) and 2 (p=0.0462). That is, subjects familiar with smartphones spent significantly less time to complete tasks 1 and 2 using the MobiSurf interface. In the MobiSurf interface, a smartphone was used as an external controller to supplement a tabletop interface. Therefore, the previous user experience for smartphones affects the efficiency for the MobiSurf interface.

In summary, MobiSurf provides a balanced interaction style that considers both usability and privacy for collaborative applications. Our results showed that the MobiSurf interface provided a seamless interaction between a personal device and a shared tabletop with minimal distraction. We identified two main limitations during the user study. First, the proposed approach utilizes tangible objects to track users over the display, but it is only applicable to horizontal displays where a user can comfortably place the tangible object on top of a tabletop. Modifications are necessary to extend our approach to vertical or wall mounted devices. Second, the use of infrared sensors to identify pointers’ infrared tags limits our system to indoor environments.

VI. APPLICATIONS

Based on the MobiSurf framework, interface designers can design inter-device bimanual interactions through the following steps.

1) Design a standard tabletop interface. Interface developers elicit requirements, and design a tabletop interface according to existing guidelines.

2) Identify inter-device user interface (UI) elements. After designing the standard tabletop interface, developers select UI elements that require inter-device interaction. For example, a textbox for a password within a standard tabletop interface is extended with an inter-device interaction by using a mobile device for a private input while some text blocks or images can be given with the sharing capacity across different devices.

3) Define the intended action of a tabletop gesture on a UI element. Developers must define the action of a gesture (e.g., “PointerOver” or “PointerRotate”) on each inter-device UI element. According to the intended action, corresponding action message(s) are transferred to and performed on an associated mobile device. The mapping between a tabletop gesture on a UI element and its action provides the flexibility for interface developers to define the application-dependent semantics for user’s actions.

Based on the above design process, we have designed three applications, discussed as follows.

Although tabletops are getting more and more popular as public interfaces, they lack accessibility features for blind users. Though commercial tools (e.g., Apple’s Voice Over, and Google’s Eyes-Free) have been proposed to support accessibility features on tabletop devices, they in general combine the gesture-based input with the speech-based output for blind users, which may be potentially limited in a public environment due to ambient noises. In addition, the voice input on a tabletop is only useful for single-user interactions because different sources of sound can interfere with each other. Furthermore, most approaches require significant modifications on the hardware or software, which makes the new system inaccessible for non-blind users. Without compromising the usability for non-blind users, we developed a prototype (See Figure 2) to facilitate blind users to access information in a public environment. The prototype gives vibration and voice feedback to a visually impaired user through his/her personal mobile device. Therefore, one user’s interaction does not interfere with others. More specifically, non-blind users interact with the tabletop application in a normal way. On the other hand, the prototype provides accessibility features for visually impaired users. The boundary in the following screenshot (Figure 2) includes a serial of red blocks that guide blind users to browse different parts of the interface. In the beginning, a blind user places his/her pointer on the top-left corner of the screen. Then, speech feedback, which directs the user to the destination, is delivered to the user’s mobile device when the user moves the pointer along red blocks. The user is alerted through vibration if he/she accidently moved the pointer off the track. When the user’s pointer arrives at the destination, the application generates an action message “Speech” on the user’s mobile to request the corresponding information. The user can input the required information through speech on his/her mobile, which does not interfere with other users.

In addition to applications in a public environment for both disabled and normal users, MobiSurf is useful for collaboration, such as brainstorming or class discussion. Taking advantage of the large screen of a tabletop, people can gather around a digital surface and interact with the system simultaneously. The MobiSurf interaction technique is especially useful to protect privacy and support data sharing in collaborative tasks. For example, a meeting coordinator can easily issue a private poll on a particular idea and others can privately send their responses to the coordinator through the “PointerShare” gesture. Furthermore, during the discussion, a participant can access or input sensitive information through his/her mobile device while he/she uses the tabletop as a public space for discussion and information sharing. Another potential application is to support class discussion in educational applications. For example, an instructor can use a public display to share contents with students (i.e., the mobile-to-tabletop sharing) while students can use private sharing to send back their answers to the instructor (i.e., the mobile-to-mobile sharing). Voice, vibration and other multimodal feedback can add more interactivity.
VII. CONCLUSION AND FUTURE WORK

This paper presents the MobiSurf framework that supports inter-device bimanual interactions in different scenarios through tangible objects and mobile devices. MobiSurf is featured with an event-driven architecture, which maps user’s actions on a tabletop to corresponding interaction commands on a paired mobile device. We investigated different scenarios, to which the MobiSurf interaction style is applied. A user study evaluated the usability of the MobiSurf interaction style. The results indicated that the proposed interaction style was as usable as the standard multi-touch interaction while offering improved privacy and efficient data sharing. In the user study, some advanced features, such as “Region selection” or “Multiple pointers”, were not evaluated. We will evaluate these features in future experiments. The future work also includes developing complex real-world applications, especially focusing on improving the accessibility for vision-impaired users on tabletop interfaces, and evaluating their usability. In addition, we plan to evaluate the ease of instantiating the framework in applications.

REFERENCES


