Interacting with Smartphones on Tabletop Computers
Combining Smartphones and Tabletops: Interaction Design

Master Thesis by Leo Sicard
lnsi@itu.dk

March 2012

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem. Investigationes demonstraverunt lectores legere me lius quod ii legunt saepius. Claritas est etiam processus dynamicus, qui sequitur mutationem consuetudium lectorum. Mirum est notare quam littera gothica, quam nunc putamus parum claram, anteposuerit litterarum formas humanitatis per seacula quarta decima et quinta decima. Eodem modo typi, qui nunc nobis videntur parum clari, fiant sollemnes in futurum.

Acknowledgements

Juan David Hincapie Ramos, Aurlien Tabard, Jakob Bardram, Sebastian Bttrich, PitLab, experiment volunteers. Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem. Investigationes demonstraverunt lectores legere me lius quod ii legunt saepius. Claritas est etiam processus dynamicus, qui sequitur mutationem consuetudium lectorum. Mirum est notare quam littera gothica, quam nunc putamus parum claram, anteposuerit litterarum formas humanitatis per seacula quarta decima et quinta decima. Eodem modo typi, qui nunc nobis videntur parum clari, fiant sollemnes in futurum.

Contents

Chapter 1

Introduction

1.1 Research context

Modern smartphones are able to support most users' daily computing tasks. They fit in a pocket, which makes them ultra mobile, and they offer good storage capacities as well as all-around connectivity. This tendency implies that users have access to personal data and applications at all times. Smartphones give rise to a new type of computer interaction which is unplanned, spontaneous, on-the-go. They bring computing to situations where laptops don't fit, such as standing in a crowded train, or walking in the street. Furthermore, they make it possible to get the most out of unforeseen opportunities, and in particular, they seem to be the ideal tool to support these chance meetings that suddenly turn into constructive collaboration. However, the size of the device can be a limitation to this form of improvised computer interaction, especially in situations with simultaneous users.

Tabletop computers, on the other hand, are made for collaboration. They are cutting-edge devices that merge input and output spaces into one horizontal interactive surface. They have been the focus of extensive research since the 1990s, with early systems such as the DigitalDesk [Wellner 1993] and DiamondTouch [Dietz and Leigh 2001], that used overhead projection. In recent years, tabletops are being commercialized, with interactive displays mostly based on computer vision and capacitive technologies. Capacitive screens are marketed since the iPhone [Apple 2012], but are now being produced in larger sizes [Displax 2012], [3M 2012]. Though its predecessor uses camera-based vision, the latest Microsoft Surface [Microsoft 2012a] is based on PixelSense technology [Microsoft 2012b]. Other computer vision based solutions include MultiTaction [MultiTouch 2012] and side vision overlays [PQ Labs 2012]. Thus, tabletop displays are gradually becoming part of the infrastructure in shared environments such as meeting rooms, public lobbies, bars, restaurants, etc... They are an ideal platform for spontaneous use, and multi-user interactions. Furthermore, they support a multi-touch based experience that is in many ways similar to the one on most smartphones.

The latest smartphones boast 720p HD screen resolutions (1280 by 720 pixels) that exceed the naked eye's ability to distinguish separate pixels. However, sometimes it is the bigger picture that matters. Reading an article and consulting a map are examples of situations in which small displays present limitations. To be able to view the data at a convenient scale, the user must zoom in on a portion at a time. This implies using repetitive zoom and pan gestures, which makes the whole interaction somewhat cumbersome. In such a situation, a smartphone could benefit from the additional screen space provided by a tabletop.

Device composition is a research area that can be traced back to the early work on smart space technologies [REFERENCE]. It has been steadily gaining importance due to the growing multiplicity of computing devices and mobility of users. Nowadays, a typical user owns mobile computers (handheld, tablet, laptop, etc...) and interacts with other devices in an ad hoc way, as s/he comes upon them throughout the day (public desktops, printers, displays, etc...). Enabling efficient communication and collaboration between various devices is therefore an essential issue, with the overall goal of improving the user experience.

Device composition focuses on the following challenges:

Connection: this includes device detection, identification and connection.

Communication: different types of devices (hardware, OS) must use a common language if they are to collaborate.

Sharing: collaboration often requires sharing information. Besides technical problems, this raises the privacy issue of protecting the user's personal data.

Interaction: the user must be able to interact with the system if s/he is to benefit from it.

This project focuses on the human computer interaction aspect of combining smartphones and tabletops. The slightly broader issue of combining smartphones and larger interactive displays has been approached in various ways, each focusing on a specific interaction metaphor.

Streaming is a one way approach where only the visual output of the smartphone is forwarded to a remote display.

Replication goes one step further, by allowing the user to interact with the replicated UI.

Projection is a metaphor similar to replication, in which the smartphone is used as a projector, allowing the user to "drop" the UI onto an available display [Winkler et al. 2011].

Adaptation refers to an improved UI transfer where the UI is modified to make full use of the additional resources offered by the remote display [Arthur and Olsen 2011].

Extension provides the possibility of transferring single applications/processes to a remote machine.

This project focuses on UI replication because it improves the user experience while keeping the interaction natural, and it can be implemented with the available resources. On a technical level, the advantage of UI replication is that it uses the application logic of the personal device, requiring of the remote display only to forward graphical output, and touch-based input. This allows the development of software that is easily adaptable to various programming platforms. On a human-computer interaction level, it reduces the learning curve for the user, by providing an intuitive experience that is similar to the one s/he is used to. By comparison, the streaming metaphor is too limited, and the other paradigms all introduce new interaction dimensions that require user adaptation. A final argument in favor of UI replication is that it allows the implementation of an engaging prototype without requiring any additional graphical design.

1.2 Problem statement

problem is actually the user-centered design of a composite device between smartphone and tabletop (with extra focus on interaction techniques)

Intuition is the ability to understand something immediately, without the need for conscious reasoning.

This project attempts to find out *how to design intuitive systems that integrate smartphones and tabletops.*

Particularly, the following questions are asked:

- is it feasible to build a system that supports the UI replication of any smartphone to the Microsoft Surface tabletop computer?
- which interaction techniques should be used to design for an intuitive user experience?

1.3 Research methods

To answer these questions, the following methods are used:

- a literary review and analysis of the research background,
- a requirements analysis of the system,
- a solution design produced via a user-centered approach [Benyon 2010],
- the implementation of an application prototype,
- an evaluation of the solution by way of a usability study.

1.4 Results

This report shows that it is possible to develop software that supports novel interactions, without requiring any conscious learning effort of the user, by designing intuitive systems. *TIDE* (Tabletop Interactive Display Extension) is a prototype that makes using a tabletop to interact with a smartphone as natural as interacting with the smartphone itself. The report presents design guidelines with a list of the interaction techniques that are most intuitive for the user in this application context.

The TIDE prototype shows that it is possible to implement an application on the Microsoft Surface that replicates the UI of any type of smartphone. It currently supports iOS and Android smartphones, and can easily be extended to other devices. The system consists of two main components. The *Remote UI* is a window on the tabletop that replicates the UI of a smartphone, and allows remote interaction. It is based on the VNC protocol [Richardson et al. 1998]. The *Surface UI* is the tabletop UI that contains the Remote UI, and provides controls to manipulate it.

1.5 Thesis overview

Chapter 2 presents a literary review of the research that constitutes the background to this work, and the theoretical work on which the design approach is based.
Chapter 3 describes the process that lead to the design of the TIDE prototype. The system itself is presented in Chapter 4, and its evaluation by way of a usability study in Chapter 5.
Chapter 6 is a discussion that addresses the results and lessons learned throughout this process, and brings suggestions for future work.
Chapter 7 concludes the report.

Chapter 2

Related Work / Theory

start with context/history - smart rooms - digital desk - metadesk / tangible interaction

then device composition (use a matrix to show all possibilities)

pairing = technologies

UI distribution = technologies, approaches

tabletops tracking (objects, users) identification

INTERACTION DESIGN / THEORY

- intro / theory = Norman, Buxton (sketching user experiences) - methods / approaches - gestures - interactions

Chapter 3

Design

The thesis investigated here is that using tabletops as basic IO peripherals for other computing devices will help their adoption by the public. A user should be able to transfer the display of his/her device to a tabletop, and interact with it in a natural way. The success of such an interaction model depends on the usability of the system.

To avoid any confusion, a few concepts must be defined. The system involves a *person*, a *personal device* and a *tabletop*. Human computer interaction takes place on the screen of the tabletop. The application UI is divided in two main parts. The *display extension*, or *remote UI*, is a window that replicates the display of the personal device on the tabletop, relaying touch input and graphical output dynamically. The *surface UI* consists of UI elements on the tabletop that allows the manipulation and control of the remote UI on the interactive surface.

This chapter presents the design process of such a system. Section 1 focuses on understanding the context of use and section 2 is an analysis of the requirements. Together they present the conceptual design. Section 3, 4, and 5 describe the process of realizing a physical design for the surface UI element. They respectively present the generation of ideas, their evaluation through a participatory usability study, and the design decisions.

3.1 Enhancing mobile computing with tabletops

It is critical to understand fully the context in which the system will be used in order to achieve a good design. Most users own a computing device with personal data and applications that are tailored to their needs. Those personal devices are becoming smaller and more mobile, with devices such as tablet and handheld computers. In many cases, the display size of the personal device is a limitation in terms of graphical input and output, and has a negative influence on the user experience. One of the main characteristics of tabletops, however, is that they have superior graphical I/O capabilities. This project focuses on situations where a tabletop can be used as a display extension to the personal device, thus enhancing the user experience.

Figure 3.1 describes the primary use case. The system should basically be a graphical peripheral unit for other computing devices. Its main functions are to forward user input to the device, and to display graphical output from the device.

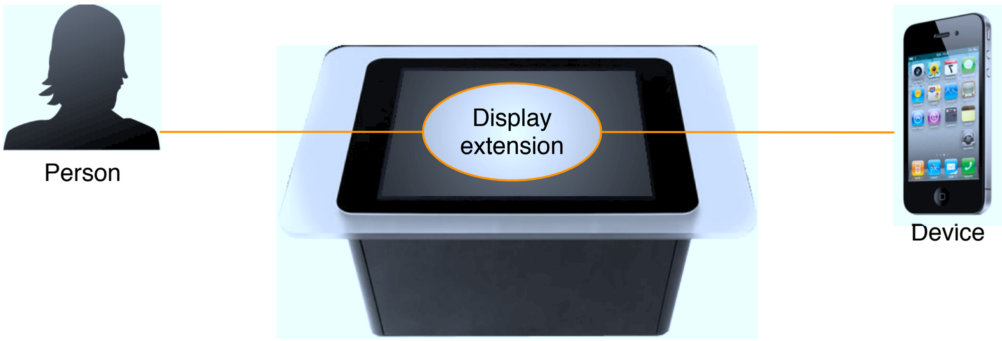

Figure 3.1: Main use case.

3.1.1 The devices

Tabletops

A tabletop presents a range of characteristics that have concrete implications on the system design. A tabletop is . . .

. . . a computer. This project investigates a strategy where a tabletop is used as a smart IO peripheral. The system should allow a tabletop to function as a relay between a personal device and users, forwarding both input and output as required.

. . . a table. Its working surface is horizontal by nature, with the implication that it cannot support any prolonged interaction, because of the bad ergonomics of the "hunched over" working position. Furthermore, a table gives naturally rise to various activities, such as eating, and it supports all sorts of objects, often in a collocated collaborative atmosphere. The system should therefore support simultaneous users, and handle limited space availability. Finally, a table offers no specific orientation, implying that the orientation of the system UI will have to adapt to the user's position around the tabletop.

. . . a situated device. A tabletop is not mobile. It usually sits at a specific location, and users come to it in order to use it. The main implication is that tabletops seem to naturally fit in public spaces, where they are shared among multiple users. This tendency is accentuated by the price factor, that makes a private person not likely to buy such a device for private use. The system should handle public use, characterized by short anonymous sessions and an often interrupted interaction flow. Other scenarios should be considered as well, such as a tabletop in an individual office, or in a family home. In any case, it can be expected to have dedicated power supply and network connection, removing such concerns from the developer's mind.

. . . a shared device. In many cases the tabletop will be shared by multiple users, raising concerns of user identification and data protection.

. . . an interactive surface. As such, it typically offers large graphical output, as well as a range of input techniques to allow for user interaction. Most tabletops support multitouch-based input. This introduces a new kind of interaction model, more intuitive, that the system should be based upon. In some cases, such as with camera-based devices, it is possible to add tangible objects to the tabletop experience. This project reports on the possibilities to use a personal device as a tangible control integrated to the tabletop.

Mobile computing devices

A mobile computing device is . . .

. . . a computer. It offers enough computing power, storage space and connectivity to support most users' daily tasks. The system should allow the user to access his/her applications and personal data.

. . . a small device. Hence the mobility. Such a device is carried around by users. However, the small form factor implies a suboptimal graphical user experience. The system should try to improve this, by offering superior IO resources.

. . . a mobile device. With the implications that its power supply is limited by battery life, and its connectivity is unstable. The system should be developed with those concerns in mind.

. . . an interactive device. Most mobile devices are now equipped with touch-based displays, making them naturally suitable for display extension on a tabletop. However, all devices present physical buttons to the user as well. The physical buttons implement strategic functions that the system should support.

3.1.2 The situations

There exists different types of everyday situations in which the system can be put to use. They vary depending on whether they involve one or more users, and whether the tabletop is a public or private device. A tabletop is considered a public device as soon as there are more than one user that have access to it. As a consequence, the scenario of multiple users on a private tabletop is not considered. Each situation implies a slightly different set of application features. The original scenarios are included in appendix A.

Single user on a public tabletop

It should be possible for the user to wirelessly pair his/her personal device with a public tabletop computer. This implies that the devices are both connected to the local wireless networks, that they are able to detect each other and discover each other's identity on the network. It would not be safe to establish this connection automatically in a public space. Therefore, dialogs should be used both on the mobile device and on the tabletop to gather user input. The UI of the mobile device should be transferred to the interactive surface as graphical output, and this transferred display should be able to accept touch input to be forwarded back to the device. The transferred display should be contained in an application window, and this window should be manipulable (drag, resize, rotate, minimize, hide, ..).

The application window should react to the state of the interactive surface. An example of this is that the application window should turn inactive if it is obstructed by an object on the table.

Finally, the mobile device could be a participant in the interaction model, i.e. listing it off the table should interrupt the connection and exit the application.

Multi users on a public tabletop

In a collocated collaborative context, it should be possible for more than one personal device to simultaneously have their display extended to the interactive surface. This implies that the implementation should support parallel connections and simultaneous use.

Mobile computing devices come in many forms, and ideally the system should support all of them. Devices vary in terms of software and hardware specifications. Some parameters that are especially important here are the programming platform, as well as the display resolution.

When a tabletop is used simultaneously by multiple users, there is a very concrete risk of lack of space on the surface. This fact introduces a new need for the system, to allow a user to remove his/her personal device from the table, while keeping the display extension active.

Single user on a private tabletop

If the tabletop is private, such as a home computer or a working desk, the system should offer extended functionalities to the user. He/she should have the option to configure the tabletop in order to allow/initiate those extended functionalities. Some suggested functionalities are:

- automatic launch of the display extension application
- push application widgets from the extended display to the tabletop
- share data between the personal device and the tabletop

3.2 Solution requirements

The general focus is on a seamless user experience.

3.2.1 Pairing

Connecting a personal device to a tabletop should be a quick and easy process. The system should include a detection mechanism that would allow the devices to become aware of each other, as well as a discovery protocol to gather the information necessary to the pairing, such as a Network IP. The connection should be wireless to guarantee a smooth experience. However, a public tabletop should not be allowed to gather data from, let alone connect to, a personal device without the explicit consent of its owner. In the case of a trusted setup, it should be possible to bypass any explicit user input. Exiting the application and closing the connection should also be easy, allowing the system to handle short successive sessions.

3.2.2 Remote UI

At the core of the system is the display extension. The UI of the personal device should be transferred to the tabletop, allowing the user to interact with it in a natural way. Graphical output from the personal device should be forwarded to the tabletop, and touch-based input from the tabletop should be forwarded to the personal device.

3.2.3 Surface UI

The user experience should be improved by using the display extension on the surface. Therefore, it is important to provide for a rich interaction. The transferred UI should be contained within a manipulable window on the tabletop. Specifically, the user should be able to move, rotate and resize the window; as well as minimize, hide and restore it. The system should include UI elements that implement the functions supported by the physical controls present on the personal device. Those elements and their function should be obvious to the user.

Modern smartphones include sensors that allow to switch the display orientation by tilting the device. This feature is strategic to certain applications, and should be implemented by the system. Obviously, tilting the tabletop is unfeasible, so another solution is necessary.

As mentioned earlier, a tabletop's screen space would typically be shared among different applications and/or objects. Therefore, the system should handle limited screen space, and obstruction of the display extension.

In a trusted setting, the user should have the possibility to push application widgets from the personal device to the tabletop, outside of the display extension, thus saving space on the latter.

3.2.4 Tangible UI

It is a natural thing to place an object on a table, and tabletops are designed to allow for the integration of physical artifacts. Therefore, it should be possible to use the personal device as a tangible UI. For example, it would seem obvious that placing the device on the tabletop would launch the pairing process, or that lifting it off would interrupt the connection. Furthermore, it should be possible to control the position of the display extension by sliding the personal device on the surface.

3.3 Interaction design: the surface UI

The solution requirements are divided into 4 elements: pairing procedure, remote UI, surface UI and tangible UI. The rest of this chapter investigates the physical design of the surface UI only, for the following reasons.

The pairing procedure does not introduce anything new to the field. There are known solutions to this challenge, which are described in section **??**. The remote UI replicates the UI of the personal device on the tabletop, and does not require any supplementary design. Using the personal device as a tangible UI for the display extension raises a series of design and implementation challenges. It is the opinion of the author that this research angle is promising, but it was decided to

leave it out of the project for reasons of time constraints.
The surface UI is the part of the system that allows the user to manipulate the remote UI on the tabletop using touch-based input. Touch-based interaction removes the necessity of having input peripherals such as keyboard and mouse, and provides the user with a more direct and sensual experience. The aim is to achieve a design that allow the user interaction to be intuitive.

3.3.1 Generating ideas

The generation of ideas is an important part of the design process. David Benyon refers to it as envisionment [Benyon 2010], and defines it as the process of externalizing design thoughts. The techniques that were used to permit this process are brainstorming, sketching, storyboarding and prototyping.

Storyboards

The scenarios mentioned in section 3.1.2 are used as a base for the making of storyboards. Storyboarding helps getting a feeling for the general flow of the interaction with the system. At the same time, it gives a visual dimension to the definition of the different system features, and raises new design issues.

Several system features were described in the scenario as being the effect of a tap on a button. When storyboarding, it became obvious that having too many UI buttons would be cumbersome, which lead to the consideration of other interaction techniques. Similarly, the challenge of the location and size of the display extension on application launch became apparent with the first storyboard.

Low fidelity prototypes

Paper prototypes were used to aid the process of generating and evaluating as many possible design solutions as possible. Screenshots of the iPhone UI [Apple 2012] were printed out in various sizes, and used on a normal table to simulate interaction with the surface UI. Figure 3.2 shows some prototypes and a working session.

3.3.2 Defining interaction stragegies

To support the interaction design process, the concepts of *actions* and *commands* are used. They are inspired by the work done by Wobbrock et al. on hand gestures for interactive surfaces, [Wobbrock et al. 2009].

Human computer interaction can be modeled as a simple cause-effect relationship. The user wishes the computer to execute a command. To achieve that,

Figure 3.2: Working with low fidelity prototypes.

he/she performs an action to provide input. In the case of a touch-based interactive surface, the action is typically a hand gesture. The action is the cause, the command is the effect, and together they form a single interaction between user and machine.

Commands

The following six basic commands are identified as interaction primitives for the surface UI.

1. *Dragging* the application window across the interactive surface.
2. *Rotating* the application window across the interactive surface.
3. *Resizing* the application window across the interactive surface.
4. *Minimizing* the application window, making it possible to restore it easily.
5. *Hiding* the content of the application window.
6. *Exiting* the application, thus closing the application window.

It is also agreed that the surface UI should offer an implementation for any supplementary command that is controlled by a physical button on the personal device.

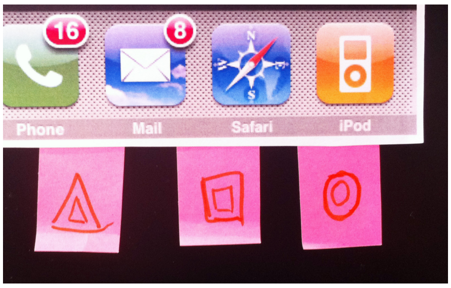

Figure 3.3: action tabs prototype

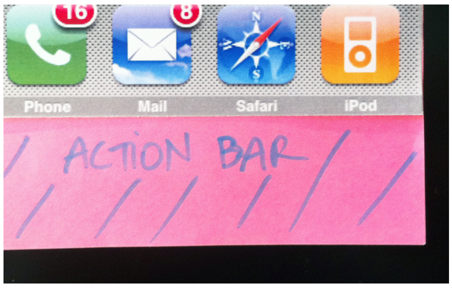

Figure 3.4: action bar prototype

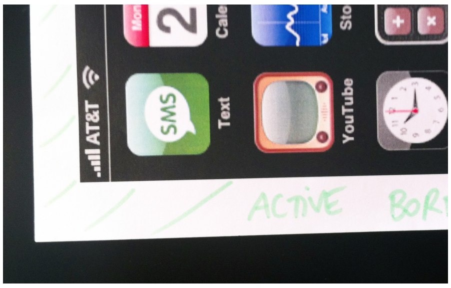

Figure 3.5: active border prototype

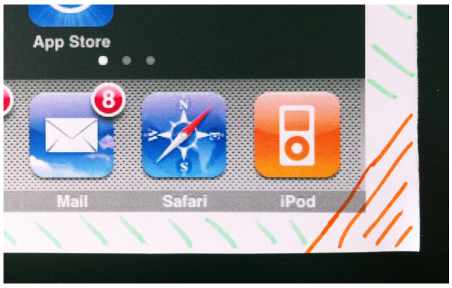

Figure 3.6: active corners prototype

Actions

Various interaction techniques can be used to invoke application level commands, as shown in figures 3.3 to 3.6. Working with paper prototypes to generate ideas lead to the definition of a set of five interaction strategies. Each strategy can be consistently implemented for each previously defined command. There is a sixth category, that regroups design suggestions that are not part of a consistent strategy.

1. *Action Tabs* are traditional buttons/tabs that implement functionalities.
2. The *Action Bar* can be compared to a virtual touchpad, it includes a manipulation area and buttons.
3. *Window Toggle* refers to using a switch to toggle the window between inactive and active states. In its inactive state, the window is made manipulable as a common digital picture.
4. The *Active Border* is a digital frame around the application window used for manipulation.
5. *Active Corners* is a strategy similar to Active Border, with the difference that the border's corners implement specific functionalities.

6. *Other* regroups suggestions that do not fit with any specific strategy.

3.4 Preliminary usability study

In order to design an interactive system that is both usable and engaging, it is important to keep the process human-centred. This can be done by taking a participatory approach, where users explore and evaluate ideas. This study was run with this goal in mind. It is based on an experiment where users engage with low fidelity prototypes of the system, and are asked to evaluate the interaction strategies that are defined in section 3.3.2.

3.4.1 Method

Parameters

The parameters of the experiment are the commands, referred to as *interaction primitives*, and the actions, referred to as *interaction strategies*, defined in section 3.3.2. The six interaction primitives represent each a system feature that is deemed critical to the overall design. For each primitive, there are six interaction strategies. The aim of the study is to have real users compare and evaluate those strategies, and to determine which ones are best fitted for each primitive.

Figure 3.7: Participant and designer during an experiment session.

Experiment

Twelve participants were recruited on a voluntary basis. An experiment session involves one participant and one designer. The participant sits next to the Microsoft Surface tabletop [Microsoft 2012a], and is presented with an iPhone [Apple 2012], but both devices are off. On the tabletop are paper prototypes, that are to be used as representations of UI elements throughout the session. The designer leads the experiment by reading instructions from a script (included in appendix B) and answering the participant's questions.

During the introduction phase, the following things are explained to the participant:

- the purpose of the study
- the purpose of the TIDE application
- the tasks that the participant will perform
- the principles of working with paper prototypes

During the main experiment, the user is asked to perform a task using the prototyped application. The task is to write an email, and it requires the user to go through six phases. Each phase is dedicated to an interaction primitive, and they have the same structure, which is as follows:

1. the primitive is explained to the participant in terms of a command to the application
2. the user is asked to suggest an action that he/she would perform to obtain the desired effect, and to demonstrate the action using the prototypes
3. the designer gives three action suggestions, the user is asked to demonstrate the actions, then rank them by order of preference.

There is a seventh phase focusing on the pairing procedure. This phase occurs first and is meant as an example to the participant, describing the common structure. At the end of this phase, a slide animation is used to describe all six primitives to the user.

For each command, there are six possible actions. However, it was decided that asking a user to choose between six choices would be overwhelming. Therefore the volunteers are split into 2 groups, each evaluating a subset of the interaction strategies. The repartition is shown in table 3.8.

	Group 1	Group 2
Dragging	1,2,3	4,5,6
Rotating	2,3,4	5,6,1
Resizing	3,4,5	6,1,2
Minimizing	4,5,6	1,2,3
Hiding	5,6,1	2,3,4
Exiting	6,1,2	3,4,5

Figure 3.8: The repartition of the evaluated interaction strategies between the two groups of participants. The strategies are 1) Action Tabs 2) Action Bar 3) Window Toggle 4) Active Border 5) Active Corners 6) Other.

3.4.2 Results

Participant answers were gathered by the designer in a form such as the one included in appendix C. The form is a matrix where an entry corresponds to a pair (primitive,strategy). Those entries contains the position from 1 (highest) to 3 (lowest), that was given by the user for using the suggested strategy for implementing the primitive. After processing all answers, each entry contains 6 positions. In order to obtain a numeric score for each entry, a weighted average was calculated, giving a weight of 3 to a first position, a weight of 1 to a second position, and a weight of 0 to a third position. Finally, the results were normalized to a [0-1] interval, where a 1 signifies that the entry was awarded a first position by all participants, and a 0 signifies that all participants ranked this entry third. Figure 3.9 summarizes the normalized scores, with colored cells containing values above 0.6. This is considered a superior score, because it can only be obtained if half of the participants awarded the first position. The same results are presented in the form of charts in figure 3.10.

	Action Tabs	Action Bar	Window Toggle	Active Border	Active Corners	Other
Dragging	0.50	0.61	0.22	0.89	0.44	0.00
Rotating	0.06	0.56	0.17	0.61	0.78	0.50
Resizing	0.56	0.22	0.17	0.50	0.67	0.56
Minimizing	0.44	0.72	0.00	0.67	0.00	0.67
Hiding	0.33	0.17	0.39	0.78	0.44	0.56
Exiting	0.28	0.06	0.50	0.50	0.33	1.00
Avg	0.36	0.39	0.24	0.66	0.44	0.55

Figure 3.9: Normalized weighted average of the ranks given to each pair (primitive, strategy).

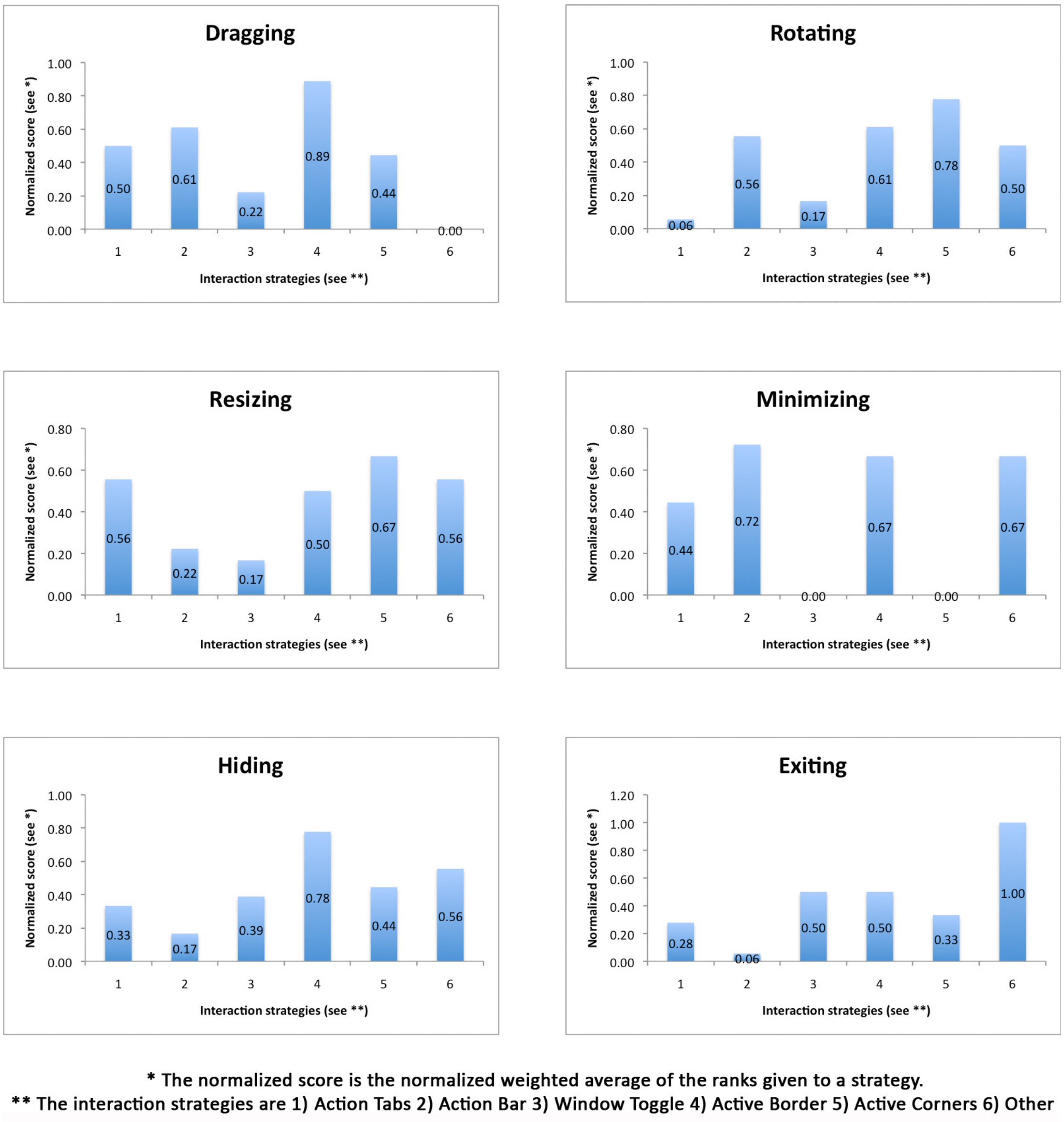

Figure 3.10: The score of the different interaction strategies for each primitive.

Analysis

It is possible to divide the primitives into two groups. The first half -dragging, rotating, resizing- have a concrete visual signification, where the second half -minimizing, hiding, exiting- are more abstract.

For the first three primitives, there is a strong coherence in the choice of the participants. The favored strategies are the active border, the action bar and active corners. All three require the user to interact with an area directly around the window in order to manipulate it and modify its position, orientation or size. This interaction strategy is similar to the current standard for manipulating pictures on interactive screens, with the difference that in the case of the display extension, the window containing the remote UI is logically avoided because of its role as IO bridge between the tabletop and the personal device.

For the last three primitives, the action bar and active border continue to score high, even though there is no apparent relation between the visual aspect of the strategy and the effect implied by the command. To understand this, it is necessary to look at the relevant entries in more detail. The favored strategies for minimizing were double tapping on the action bar and double tapping on the active border. For hiding, the favored strategy was double tapping on the active border. It is thus obvious that it is the double tap that participants have a preference for, possibly because it is a common technique in many other application contexts, as well as a quick and easy one.

It is not possible to analyze the scores of the sixth interaction strategy as a whole, as it does not represent a consistent strategy, but more a patchwork of various implementation suggestions. Those suggestions were chosen because of their originality and interest, and are listed here:

1. Drag by holding a finger on a specific tab, and using another finger to tap a destination target to move the window to.
2. Rotate by performing a one finger dragging gesture on a corner of the window.
3. Resize by pulling the window apart with both whole hands.
4. Minimize by dragging the window to the bottom of the surface.
5. Hide by placing and holding a full hand on the window.
6. Exit by dragging the window to a specific location on the surface.

Suggestions 4 and 6 are similar, and they are the only ones that scored above 0.6. This suggests that moving the window off screen is a natural way to remove focus from the application. It is interesting to see how this correlates with the open suggestion analysis below.

Open suggestion analysis

The open suggestions were multiple and heterogeneous, but further analysis showed a definite tendency in three situations.

In the case of dragging, half of the participants suggested using one or more fingers to touch inside the window containing the remote UI, and perform a drag gesture. This is interesting, because it is an obvious conflict for the developer, i.e. any touch inside the window is forwarded to the remote UI, and can therefore not be interpreted as a surface UI manipulation. It must be noted that dragging was the first primitive, and it can therefore be assumed that the participants did not yet have a full understanding of the remote UI concept. However, this result shows well that the ideal goal would be to design an interactive system whose UI could interpret the intention of the user.

In the case of resizing, 8 out of 12 participants suggested grabbing the sides of the window with 2 fingers, and pulling the window apart to enlarge it. This shows that the gesture is very intuitive, and that a good usability would require implementing this interaction technique.

The third consensus was for minimizing, where 7 out of 12 participants suggested dragging the window offscreen (or to a specific location on the surface edge). The same suggestion reoccurred for the primitives hiding and exiting, though with less decisiveness. Once again, it shows that the gesture is an intuitive one. Moreover, there is a direct parallel between removing a real piece of paper from the center of a table, and the act of minimizing, hiding or exiting the display extension application. Moving the application out of focus seems like a good solution, and using a simple dragging gesture is a natural way of doing it.

3.5 Design Decisions

The usability study provided precious input from potential users of the system, and this input supported the designer's views that in order to build a successful system, the focus should be on usability and intuition. In this context, intuitive interaction means that the user should be able to learn the system through free exploration, all the time guessing and discovering functionalities. To allow this phenomenon, the implementation should focus on three things. First, it should be coherent. If the features are consistent, the user will be able to derive one from the other. Second, common interaction techniques should be used, such as the picture manipulation gestures (drag, pinch, rotate) that are already successful on touch-based interactive screens. Third, the implementation should refer to the table metaphor when possible, using the user's familiarity with normal table objects to hint at specific features.

The implementation choices are as follows:

- Dragging, rotating and resizing are done by manipulating an active border that frames the window. The active border has the visual appearance of the body of the physical device.

- When the window is dragged off an edge of the table, the application is minimized. Minimizing can also be done by double tapping the active border.
- In its minimized state, the application appears as an icon on the tabletop. This icon can be moved around, restored, or closed.
- On closing the application, the user will be prompted to confirm exiting the application.
- Exiting can also be done by dragging the window to a corner of the tabletop, or by pressing and holding one finger on the active border.
- Hiding is achieved by minimizing the application. Minimizing is therefore also implemented as a result of holding a whole hand over the window.
- When reduced under a certain threshold, the application is minimized. Similarly when enlarged above a certain size, it is maximized to a fullscreen mode. To escape the fullscreen mode, buttons are implemented in the corners of the tabletop.
- the buttons present on the physical device are a logical part of the active border, and their functions are implemented accordingly.

In most software applications, there exists two ways to activate the same command. The advantage is that one implementation can be made to be easy to discover, while the other is more obscure, but a faster or easier interaction. The best example of this are keyboard shortcuts. They can not be guessed by the novice user, but they are used by the expert user for their efficiency. Similarly, the display extension can be minimized easily and quickly by double tapping the active border, but the novice user can easily discover the feature by simply dragging the window off screen.

Chapter 4

The TIDE application: Tabletop Interactive Display Extension for mobile devices

refer to problem formulation:

Following is an open list of problems that we will address in order to achieve device composition by means of implicit interaction.

1. *Setup*: How is a device enabled for integrating with a tabletop? The setup should be simple, to be performed only once by non-technical users. An initial survey of possible solutions points towards the use of tagging mechanisms and/or camera-based object recognition.

2. *Discovery*: How do the tabletop and the device discover and communicate with each other? How do we solve the issues of discovery, handshake, network connectivity, and encryption mechanisms to ensure privacy?

3. *UI transfer*: Given the computational constraints of mobile devices, how can the UI transfer be efficiently implemented so as to support native applications and guarantee a seamless user experience?

4. *Input*: How can the users interact with their applications on the tabletop (touch and other peripherals)?

5. *Interaction Design*: What means of interaction are best-fitted for the tabletop-based systems that we propose to develop? How can we best adapt to public/private uses and single/multiple users? How can we take advantage of the larger interaction surface?

4.1 Device Composition

4.1.1 Device setup

4.1.2 Detection and discovery

–¿ how it is not new, what are the existing options, what would I recommend in this context. Discussion. How did I solve it and why.

Bluetooth isa used by personal server

4.1.3 Vision-based device tracking

vision-based device tracking detection options: iPhone App, Tag, camera based

4.1.4 UI transfer

(I/O approach) - technology issues (slow Veency)

4.2 Surface application UI

4.2.1

Chapter 5

Evaluation

in some cases, compare different interaction strategies for same command.

Compare same implementation with no visual aids, discrete visual aids, and overkill visual aids.

5.1 Experiment

Compare and discuss

5.2 Results

Chapter 6

Discussion

Implementing same functionality with several parallel interaction techniques is a good choice because it will augment the number of situations in which a user will obtain the desired effect intuitively on his first try, without referring to any manual.

Chapter 7

Conclusion

Bibliography

3M. 3m. http://www.3m.com/, 2012. (accessed 2/12).

Apple. iphone. http://www.apple.com/iphone/, 2012. (accessed 2/12).

Richard Arthur and Dan R. Olsen, Jr. Xice windowing toolkit: Seamless display annexation. *ACM Trans. Comput.-Hum. Interact.*, 18:14:1–14:46, August 2011. ISSN 1073-0516. doi: http://doi.acm.org/10.1145/1993060.1993064. URL http://doi.acm.org/10.1145/1993060.1993064.

David Benyon. *Designing interactive systems: a comprehensive guide to HCI and interaction design.* Addison Wesley, 2010. URL http://books.google.com/books?id=P923PwAACAAJ.

Paul Dietz and Darren Leigh. Diamondtouch: a multi-user touch technology. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*, UIST '01, pages 219–226, New York, NY, USA, 2001. ACM. ISBN 1-58113-438-X. doi: http://doi.acm.org/10.1145/502348.502389. URL http://doi.acm.org/10.1145/502348.502389.

Displax. Displax. http://www.displax.com/, 2012. (accessed 2/12).

Microsoft. Microsoft surface. http://www.microsoft.com/surface/, 2012a. (accessed 2/12).

Microsoft. Pixelsense. http://www.microsoft.com/surface/en/us/pixelsense.aspx, 2012b. (accessed 2/12).

MultiTouch. Multitouch. http://multitouch.fi, 2012. (accessed 2/12).

PQ Labs. Pq labs. http://multi-touch-screen.com/, 2012. (accessed 2/12).

Tristan Richardson, Quentin Stafford-Fraser, Kenneth R. Wood, and Andy Hopper. Virtual network computing. *Internet Computing, IEEE*, 2(1):33–38, 1998.

Pierre Wellner. Interacting with paper on the digitaldesk. *Commun. ACM*, 36:87–96, July 1993. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/159544.159630. URL `http://doi.acm.org/10.1145/159544.159630`.

Christian Winkler, Christian Reinartz, Diana Nowacka, and Enrico Rukzio. Interactive phone call: synchronous remote collaboration and projected interactive surfaces. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '11, pages 61–70, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0871-7. doi: http://doi.acm.org/10.1145/2076354.2076367. URL `http://doi.acm.org/10.1145/2076354.2076367`.

Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. User-defined gestures for surface computing. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 1083–1092, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-246-7. doi: http://doi.acm.org/10.1145/1518701.1518866. URL `http://doi.acm.org/10.1145/1518701.1518866`.

Appendix A

Scenarios

The coffee shop: single user on a public tabletop

Alice is sitting in a Coffee Shop, waiting for her friend Bob. The table is an interactive surface, which allows her to order her drink via the digital interface.

She takes her phone and notebook out of her purse and places them on the table. A dialog pops up on her smartphone, asking her id she wants to establish the connection between the two devices. Alice confirms this by a simple tap. A menu appears on the table beside her phone. Alice taps the 'Connect' button, and her phone's display appears on the table beside her phone.

She resizes the window to her convenience, and moves it closer to her by sliding her phone on the surface. The screen goes gray to notify Alice that an object (the notebook) is in the way. Alice removes the notebook, and the window becomes active again. She accesses her phone's applications, and starts typing an email.

When Bob arrives, Alice minimizes the surface application, keeping her phone in place. Bob orders a drink and they start catching up. After a while, Bob leaves. Alice restores the application, finishes up her email, and exits the application by lifting the phone off the table.

The meeting: multiple users on a public tabletop

Jim, Jack and Jill are having a meeting about the development of a product. They are sitting around an interactive table, with different artifacts, including paper, pens, computing devices and coffee cups. Jill is responsible for the meeting's agenda, which is stored on her smartphone.

Jill places her phone on the top right corner of the table and establishes a UI transfer. She uses the 'Grab' button to drag the display window to the left of the phone where there is space. She opens the document containing the agenda, and uses the 'Resize' corner of the window to enlarge the display, so as to allow convenient visual reference for all present.

It is now time for Jack to present a diagram of the development process. He switches on his tablet computer, opens said diagram, and places the tablet on the table for the others to see. The screen is however too small, so Jack decides instead to use the UI transfer application. With a single tap on the 'Grab' button, Jack pins the UI to the table, allowing him to remove the physical tablet while keeping the transferred display active. By using the 'Resize' corner, he enlarges and flips the orientation of the window to a landscape view. By the use of a double touch on the 'Grab' tab and the active window, Jack rotates the display to a convenient position, and presents the diagram to his colleagues. When done, Jack minimizes the window by tapping a button. The window takes the shape of an active icon, ready to be restored or closed as convenient. When the meeting is over, Jack taps the 'Close' tab on the icon to exit the application.

The office: single user on a private tabletop

It is monday morning and Bill arrives at his office. His working desk is made up of an interactive table, extended with a vertical screen, mouse and keyboard. On it are various physical objects, including a stack of papers, some books, pens, an empty cup and a lamp.

Bill wakes the tabletop up from its standby state by simply placing his smartphone on it. The devices know each other, so a UI transfer is automatically launched. Bill uses a widget on his phone to push application widgets to the table space. Bill places his calendar up in one corner, together with his Skype widget.

After reading through his mail on the vertical screen, Bill starts typing an answer using the keyboard. He needs to refer to a document that is stored on his phone. Bill uses the 'Grab' button beside his phone to attach the display beside the device. He enlarges the window and moves the display to a convenient location by sliding the phone. Bill types on..

Suddenly the phone rings. Bill taps the 'Grab' button to effectively pin all applications and UI display to the tabletop, allowing him to pick up the phone without interrupting the UI transfer.

Appendix B

Preliminary usability study - experiment script

B.1 Introduction

“You are here to participate in a short experiment whose purpose is to assist in the design of an application called Displex. The experiment will last about 30 minutes and is being recorded, both as audio and video. I will give you instructions by reading from this script. After an introduction you will be able to ask questions before we start the experiment.

First, let me introduce Displex. Displex is an application that connects a smartphone (iPhone) with a tabletop computer (Microsoft Surface). It allows you to interact with your phone on a larger screen, by transferring the display of your phone to a window on the screen of the interactive surface. Do you understand the basic concept of the application?

During this experiment, I will ask you to perform a task using the application. This will lead you to perform a number of actions that use the basic features of Displex. For each action, there will be 3 steps:

- First, I will explain the action, and show you its effect on this screen
- Second, I will ask you to describe to me how you would suggest performing this action with the user interface of Displex.
- Third, I will give you 3 suggestions of how to perform the action, and ask you to order them according to your preference.

This experiment is based on prototypes, which means that we will use the available paper representations in order to describe the user interface of Displex. There are iPhone screenshots and different types of buttons and controls. Paper,

pen and scissors are available for building your own prototypes if necessary. You are welcome to draw on the prototypes if you want. We will also use the iPhone, the MS Surface, and of course verbal communication.
This iPhone screenshot printout is a representation of the main window of the Displex application. The idea is that you can interact with this window in exactly the same way as you would interact with your phone's screen. For example, by tapping the Photos icon, you would launch the Photos application, and if you are viewing a picture, by performing a two finger pinching gesture, you could zoom on the picture. At the same time, we need a way to manipulate the window, and that is what this experiment is going to focus on.
Do you have any questions concerning the general course of the experiment?
Let us begin. Your general task is to write an email to a friend using your iPhone and the Displex application on the Microsoft Surface. We will talk about 7 basic actions."

B.1.1 Example: Pairing

This first action is only an example, meaning that I will go through all the steps myself. The action is called pairing.
Scenario: In order to use Displex, I have to pair my iPhone with the Surface and launch the Displex application. Here is a visual representation of the effect of this application. (SHOW PPF)
Suggestion:
I asked my advisor Juan, and his suggestion was to launch Displex on the iPhone, then search for available surface computers within the application, and connect to the Surface.
Selection:

A The application launches automatically when the smartphone is placed on the surface, and a dialog window appears on the smartphone, offering the user to establish the connection.

B The application launches automatically when the smartphone is placed on the surface, and 2 dialog windows appear, first on the surface, then on the smartphone, offering the user to establish the connection.

C The application launches automatically when the smartphone is close enough to the surface, and a dialog window appears on the surface, offering the user to establish the connection.

Juans order of preference was B, A, C. What about you?
(SHOW ALL PPF) There are 6 actions left, and I will now show you visual representations for each of those actions.

B.2 Experiment

B.2.1 Dragging

Scenario: Your iPhone screen is now active on the surface, and you need to move it closer to yourself. Therefore, you drag the window across the surface.
Suggestion
Selection (group 1)

A By performing a one finger dragging gesture on a specific action tab. (1)

B By performing a one finger dragging gesture on the action bar. (2)

C By tapping a tab to render the window inactive, then performing a one finger dragging gesture anywhere on the window. (3)

Selection (group 2)

A By performing a one finger dragging gesture on the active border of the window. (4)

B By performing a one finger dragging gesture on the active border (excl. corners) of the window. (5)

C By holding a finger on a specific tab, and using another finger to tap a destination target to move the window to. (6)

B.2.2 Rotating

Scenario: the application window is not oriented correctly, so you need to rotate it to the correct orientation.
Suggestion
Selection (group 1)

A By performing a two finger touch rotating gesture on the action bar. (2)

B By tapping a tab to render the window inactive, then performing a two finger touch rotating gesture anywhere on the window. (3)

C By performing a two finger touch rotating gesture on the active border. (4)

Selection (group 2)

A By performing a two finger touch rotating gesture on the active border. (5)

B By performing a one finger dragging gesture on a corner of the window. (6)

C By performing a two finger touch rotating gesture with one finger placed on a specific tab, and the other anywhere on the window. (1)

B.2.3 Resizing

Scenario: Now you open the Safari App by taping on the correct icon, but the window is too small for you to type an email, so you resize it to make it bigger.
Suggestion
Selection (group 1)

A By tapping a tab to render the window inactive, then performing a two finger pinching gesture anywhere on the window. (3)

B By performing a two finger pinching gesture on the active border. (4)

C By performing a one finger dragging gesture on an active corner. (5)

Selection (group 2)

A By pulling the window apart with both whole hands. (6)

B By performing a one finger dragging gesture on a specific tab. (1)

C By performing a two finger pinching gesture on the action bar. (2)

B.2.4 Minimizing

Scenario: Before you start writing your email, you want to verify some facts in a document. You decide to minimize the Displex application to make room for the paper, and be able to restore the window to its previous state soon after.
Suggestion
Selection (group 1)

A By double tapping the active border. (4)

B By using Resizing on an active corner to reduce the window until it snaps to icon shape. (5)

C By dragging the window to the bottom of the surface. (6)

Selection (group 2)

A By tapping a specific tab. (1)

B By double tapping the action bar. (2)

C By tapping a tab to render the window inactive, then performing a specific gesture anywhere on the window. (3)

B.2.5 Hiding

Scenario: Later, you are writing another email of a personal nature, and one of colleagues is approaching. You wish to quickly and temporarily hide what you are doing.
Suggestion
Selection (group 1)

A By double tapping an active corner. (5)

B By placing and holding a full hand on the window. (6)

C By tapping a specific tab. (1)

Selection (group 2)

A By performing a specific gesture on the action bar. (2)

B By tapping a tab to render the window inactive. (3)

C By double tapping the active border. (4)

B.2.6 Exiting

Scenario: Finally, you are finished and want to leave. You exit the Displex application.
Suggestion
Selection (group 1)

A By dragging the window to a specific location on the surface. (6)

B By tapping a specific tab. (1)

C By performing a specific gesture on the action bar. (2)

Selection (group 2)

A By tapping a tab to render the window inactive, then performing a specific gesture on the window. (3)

B By using Resizing on the active border to Minimize, then tapping a specific tab. (4)

C By using Resizing on an active corner to Minimize, then tapping a specific tab. (5)

Appendix C

Preliminary usability study - result form.

TIDE Early Usability Study (1)

	Action Tabs	Action Bar	Active /Inactive Window	Active Border	Active Border + Corners	Other	Open Suggestion
Dragging							
Rotating							
Resizing							
Minimizing							
Hiding							
Exiting							

Figure C.1: Form used to gather participant answers during the preliminary usability study.