LucidTouch: A See-Through Mobile Device

Daniel Wigdor[1,2], Clifton Forlines[1,2], Patrick Baudisch[3], John Barnwell[1], Chia Shen[1]

[1]Mitsubishi Electric Research Labs
201 Broadway, 8th Floor
Cambridge, MA, 02139,USA
{forlines | barnwell | shen}@merl.com

[2]Department of Computer Science
University of Toronto
Toronto, ON, Canada
dwigdor@dgp.toronto.edu

[3]Microsoft Research
One Microsoft Way
Redmond, WA, 98052, USA
baudisch@microsoft.com

ABSTRACT

Touch is a compelling input modality for interactive devices; however, touch input on the small screen of a mobile device is problematic because a user's fingers occlude the graphical elements he wishes to work with. In this paper, we present LucidTouch, a mobile device that addresses this limitation by allowing the user to control the application by touching the *back* of the device. The key to making this usable is what we call *pseudo-transparency*: by overlaying an image of the user's hands onto the screen, we create the illusion of the mobile device itself being semi-transparent. This pseudo-transparency allows users to accurately acquire targets while not occluding the screen with their fingers and hand. LucidTouch also supports multi-touch input, allowing users to operate the device simultaneously with all 10 fingers. We present initial study results that indicate that many users found touching on the back to be preferable to touching on the front, due to reduced occlusion, higher precision, and the ability to make multi-finger input.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces - Graphical user interfaces.

General terms: Design, Human Factors.

Keywords: portable multi-touch, direct touch, LucidTouch, transparent devices, bimanual input, pseudo-transparency, augmented reality.

INTRODUCTION AND MOTIVATION

Many mobile graphical applications require pointing input, whether it is selecting links in a mobile web browser or manipulating windows, icons, and menus in a GUI. Because designers of mobile devices strive to minimize the overall size of the device, pointing input is often accomplished with small joysticks or 4-directional button sets that allow users to move a pointer stepwise across the screen or from link to link. The poor pointing performance of these types of devices is well known [8]. In part to overcome this limitation, manufacturers have equipped some mobile devices with touch screens, usually with input and display devices calibrated for direct-touch input.

UIST'07, October 7–10, 2007, Newport, Rhode Island, USA.

Touch screens offer good pointing performance for large user interface targets [22], and save device space while maximizing the visual display area by superimposing the control surface over the screen. This has allowed the creation of devices such as the Sony Cybershot DSC-N1 digital camera and, more recently, the Apple iPhone, in which nearly the entire front surface is occupied by the screen and the size of the device is limited only by its display.

Two fundamental problems with direct-touch finger input are that the user's finger occludes the target in the critical moment before touching the display (the *occlusion problem*) and that the touch area of the finger is many times larger than a pixel of the display (the *fat finger problem*). Because of these two issues, a user is often unable to accurately specify the point of contact with the display [19]. For small mobile devices, the occlusion problem is especially drastic because the hand often covers the majority of the display. Previous solutions to these problems have attempted to provide software or hardware aids. These aids generally break the direct-touch input paradigm [12, 19, 27], map multiple points of input to a single cursor [5, 10], require additional on-screen graphics [1], or suffer from lack of visual feedback [29].

In this paper, we propose a device whose design addresses the occlusion and fat finger problems with fewer drawbacks than previous solutions. *LucidTouch* combines a behind-the-display multi-touch input surface with a *pseudo-transparent* display that overlays a live image of the user's hands onto the screen. Figure 1 shows an early concept drawing of the proposed device.

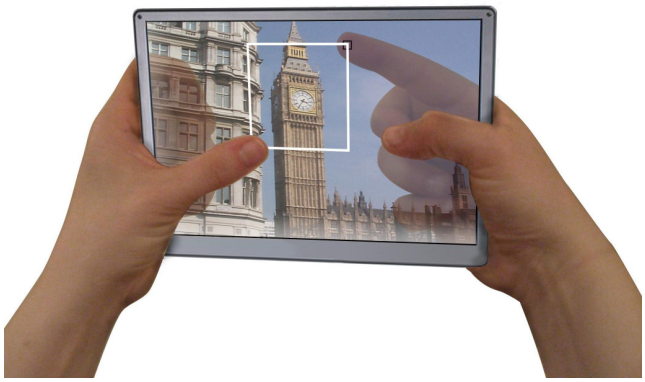

Figure 1. Concept sketch of *LucidTouch*: a *pseudo-transparent* device.

LucidTouch

The fundamental problem with a behind-the-display input surface is that the display hides the user's hands, making accurate pointing difficult [29]. Many direct touch input devices provide only two input states: out-of-range and dragging, the assumption being that the user's finger or stylus provides all the feedback they need in order to anticipate the point of interaction [6]. When the hands are behind the display, this visual tracking is not possible.

As shown in Figure 2, our pseudo-transparency approach allows users to see their hands as they are attempting to acquire a target from the back of the device, thus solving not only the occlusion problem, but also the lack of tracking feedback. In order to overcome the fat finger problem, simple computer-vision techniques are applied, allowing each finger's touch points to be visualised prior to making contact the touchpad. As a result, LucidTouch enables fast and intuitive *land-on* selection, in contrast to the *take-off* selection techniques other opaque devices employ [19, 29].

While the use of pseudo-transparency for a single point of input is a valuable contribution, we set out to leverage this technique to create a multi-touch portable device. Because portable devices are held while in use, multi-point direct-touch input from both hands has been limited two points of contact. By providing a mechanism to move direct-touch input to the back of the device, we were able to build LucidTouch so that it could receive input from all 10 fingers. Figure 2 shows our working prototype and illustrates how a user operates the LucidTouch device.

One point worth noting is that by displaying only the positions of the fingertips prior to touch, without the overhead of pseudo-transparency, we could, in effect, create several *touch-cursors*. As we will discuss, however, when making multiple points of input, most user study participants found the pseudo-transparency necessary in order to understand the correlation between each touch-cursor and the particular finger it represented. Without the overlay, they were unable to accurately control the touch-cursors.

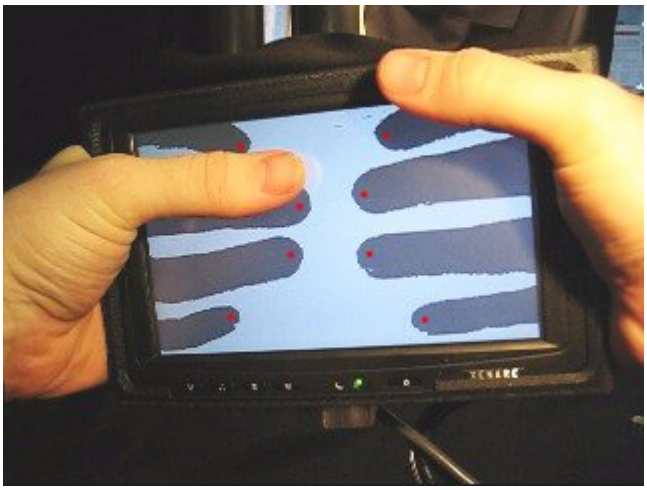

Figure 2. The LucidTouch prototype.

The remainder of this paper is organized as follows: we first review related work, then present a detailed discussion of the issues in creating a pseudo-transparent input device. We then explain in detail our implementation, and those aspects of the ideal device we are able to implement with presently available off-the-shelf technology. We also discuss properties and characteristics of LucidTouch which necessitate rethinking of traditional and multi-point direct-touch interaction. Finally, we present a trio of interaction techniques we have developed and adapted for use on the LucidTouch, and the results of a user evaluation intended to elicit feedback on pseudo-transparency and on our novel device. We conclude with a discussion of how to iterate on the design of the system, and list several open research questions.

RELATED WORK

There are several areas of research which are highly relevant to the present paper; we will review each in turn.

Several technologies have recently been demonstrated which enable the detection of multiple points of simultaneous contact, and which allow for input and display devices to be overlaid, creating a direct touch interface. Wilson's *TouchLight* system used two cameras located behind a display to detect the position of users' hands [30]. The apparatus is similar to that presented by Han, who used frustrated total internal reflection to cast touch points back towards a single camera [14]. Finally, two capacitance-based systems, *SmartSkin* [20] and the *DiamondTouch* [9], allow the detection of multiple points for tabletop systems. Forlines and Shen's *DTLens* [11] is an example of the utility of these multi-point direct-touch input.

There are several mobile devices which have enabled direct touch input with a finger or stylus. Early examples of such devices include the Apple Newton and the PARC TAB [21], each of which was designed for input with a stylus, treated in a direct-touch manner. Although not commercially available at the time of this writing, the *iPhone* has been announced to support two-point, direct touch input (http://www.apple.com/iphone/). The device is intended to be held between the palms of both hands, with input delivered with the thumbs. Although a step towards enabling multi-point input, the form factor of this device does not lend itself to more than two simultaneous touch-points, and still suffers from the problem of occlusion while touching. Several researchers have attempted to overcome the occlusion problem, while allowing input with fingers, though each has had limitations [1, 5, 10, 12, 19, 27, 29].

One possible solution to the occlusion problem is to move the input surface to the back of the device, as has been explored in the BehindTouch [15], HybridTouch [23], and Under the Table [29]. Moving input to the back of a device introduces a new occlusion problem, since the device prevents the user from seeing his hands. This problem is illustrated by a user study in [29] that found that users could only accurately land on targets of approximately 4.5cm in diameter when they were unable to see their fingers. The

LucidTouch overcomes this limitation by making use of transparent or pseudo-transparent display and input surfaces, allowing users to continue to see their hands while touching the display.

See-through devices are similar in concept to several projects in *Augmented Reality* (AR): like AR, we aim to seamlessly blend real-world and virtual content on the device. Unlike AR, our device is handheld, not head-mounted, and our real-world content is limited to the user's hands in order to avoid interference with background images. An overview of early AR work can be found in [2]. Ours is not the first system to visualise users' hands using video: both J.C. Tang and Minneman's Video Draw [25] and A. Tang et al.'s Video Arms [24] systems made use of video representations of users' hands, though theirs was intended to embody a remote user. More similar to the LucidTouch is the use of video in the *TactaPad* (http://www.tactiva.com), which presents a filtered video image of the user's hands on the interaction area. Unlike our device, however, the TactaPad is not a direct-touch device: the video of the hands is not registered with their physical position relative to the display.

The LucidTouch will make use of simultaneous input from both hands. Researchers have previously identified several advantages to bimanual over unimanual input, including increased performance [3, 7, 16], reduced cognitive load [17], and increased alignment with more "natural" practices [13]. These benefits are generally attributed to their closer mimicry of real-world interaction. As explained by Guiard, the non-dominant hand performs gross actions to lead, and providing the spatial frame of reference for, the dominant hand, which performs fine motor tasks within this established reference frame. Unclear, however, is to what degree these findings will continue to hold true for the LucidTouch: because the device is held between the palms of both hands, only the fingers and thumb actively move on the device. The roles of the hands naturally adopted by the user, and how best to leverage those roles, may not follow the Guiard model. Although it can be used as a starting point, further study is required before these results are directly applied to the LucidTouch.

It is our hope that, a device which enables multi-point, direct-touch interaction on a portable device, we will enable designers to begin to port these and other interaction schemes to the populous mobile domain.

DESIGN OF SEE-THROUGH DEVICES

LucidTouch is a mobile device that allows for direct touch input while minimizing occlusion through pseudo-transparency. It also supports simultaneous direct-touch input from all 10 fingers. In this section, we take a closer look at the technical aspects of our prototype and the different design choices we have explored.

Controlling Transparency

With a see-through display there are up to three layers of visible elements: (1) on-screen elements, (2) the hands behind the screen, and (3) background scenery located behind the device. In order to obtain good results, the visibility of all three layers needs to be controlled. In general, we want best possible visibility of on-screen contents and reconcilability of the hands. The visibility of background scenery, in contrast, is optional if not hindering.

As has been explored by the augmented reality community, there are two ways of achieving a see-through effect: actual *optical see-through* and *closed-view* pseudo-transparency [2]. We tried both.

Our first mock-up used a physically transparent display (Figure 3), which allowed users to see their fingers without the need for a video overlay. What we found, however, was that it was hard to control the visibility of the three layers. With physical transparency, the visibility of each layer depends on its illumination: the more light one shines on a layer, the more visible it is. Unfortunately, controlling illumination turned out to be difficult. In particular, a hand approaching the LCD prevented illumination from reaching both the hand and the respective area of the screen. As a result, the user's finger and the respective screen area turned dark, leading to a new type of occlusion problem. While we believe that it might be possible to solve the problem by illuminating the LCD and the user's hand with an illuminated semi-transparent bi-directional diffuser layer inserted behind the LCD and the user's hand, it soon became clear that a pseudo-transparent solution would give us all the control we needed. In addition, actual transparency would have required us to also use a see-through tracking device, while pseudo-transparency allowed us to explore a broader range of tracking solutions. Fundamentally, actual see-through devices do not allow any anything to be stored behind the screen. Processing hardware, such as circuit boards and hard drives therefore require space outside the sides of the screen, impacting the mobile form factor. Imaging-based solutions avoid this issue and allow for a common tablet-like form factor.

Figure 3. Our first LucidTouch mock-up created a physical see-through effect using a modified overhead-projector LCD. It's main drawback: lack of illumination led to occlusion when touching the screen.

Rendering Tracking Feedback

By mounting a camera behind the display to capture video of the user's fingers and hands, we gained complete control over their appearance and regained an evenly lit LCD. By adjusting the parameters controlling the compositing of the GUI and hands, we developed several methods of rendering hover feedback. For a device that supports a single contact point, a simple pointer image will typically be sufficient. In the case of LucidTouch, there are up to eight contact points on the back of the device. While it would be possible to show simply eight touch-cursor images, we were concerned that such a display might make it hard for users to determine which touch-cursor corresponded to which finger, especially in the case of individual fingers entering and leaving the tracking range. To help users interpret the tracking feedback, we chose to overlay an image of the user's hand (similar to [24 and 25]), in addition to displaying touch-cursors over each of the fingertips during both the hover (tracking) and touching (engaged/tracking) states. We experimented with various visualisations for the touch-cursor, eventually deciding on a simple dot to minimize distraction. To give additional feedback to the user, we used colour to depict which touch-cursors were hovering (red) and which were touching (blue).

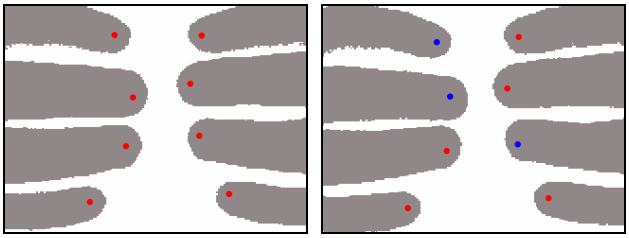

Figure 4. Left: all touch-cursors are red: no fingers are touching the device. Right: the three fingers with blue touch-cursors are touching the display.

There is a multitude of options for rendering the hand in pseudo-transparency. Given that our current prototype captures an actual camera image, one option was to display that camera as is, but this offered more detail than necessary. In addition, the camera films the backside of the user's hands, which creates an undesirable effect, as users would expect to see the inside of the hand. We therefore opted for a solid, but non-textured silhouette of the hand, which offers enough detail to interpret pointer positions while creating a minimum amount of clutter (for a discussion of how to minimize interference between layers see [4]).

Display Size and Finger Reach

In order to allow for input from all 10 fingers on a mobile device, LucidTouch is held between the palms of both hands. This enables freedom of movement for the fingers behind the display, while positioning the thumbs above the display. Although thumb input on the front of the device occludes on-screen content, as discussed earlier, enabling input from the front side provides an additional input channel. This can be useful when running applications explicitly designed for touch, such as the presentation mode of Microsoft PowerPoint™. It also allows us to create additional dual-sides gestures, and leverage the semantics of touch-side information, as explored in [29].

The holding of a handheld device constrains the movement of one's fingers - device size, grip, and input area are inextricably linked. With our prototype, interaction occurs with the fingers while the device is held between one's palms. Before deciding on this particular footprint, we explored three size classes of see-through devices. First, *small* devices allow all portions of the screen to be reached by each of the fingers. This also allows for fully functional one-hand usage. Second, on a larger, medium sized device all parts of the display are within reach of a finger, but no one finger can reach the entire display. This can require a hand-off between fingers if objects are moved across the screen. Finally, a large device would require the user to temporarily hold the device with one hand while the other hand traverses the back of the device in order to acquire some targets. Figure 5 illustrates the three sizes.

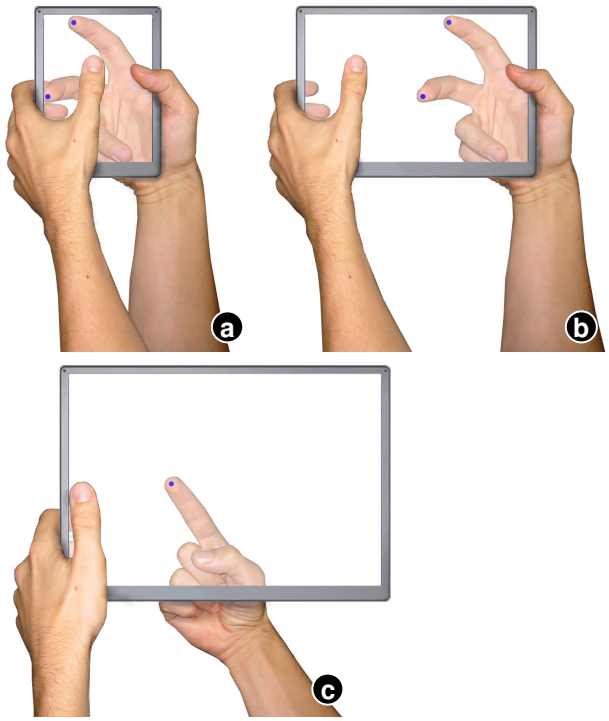

Figure 5. Concept images: (a) a smaller LucidTouch, where both hands can reach the entire display, (b) a medium LucidTouch (as in our prototype), and (c) a large LucidTouch, temporarily held in one hand in order to reach the centre of the screen.

Input Surface and Semantics

In [29], Wigdor et al. describe the potential for designers to assign semantics to input based on which side of a two-sided device is touched. Having both hands giving input on the bottom surface, for example, should have different meaning to the system than having both hands touching the same portion of the screen from the top. In the case of the

LucidTouch, the surface on which touches occur could again have semantic implications to an application. Additionally, designers should take in to account that input to a particular surface implies the use of a particular digit to make that input: touches to the front of the screen are delivered by the thumb, while touches to the back are made with the fingers. Also, unlike the two-sided touchtable in [29], the same hand can deliver input to both surfaces of the LucidTouch: as such, interactions requiring coordinated actions, such as pinching or rubbing, are possible.

Hardware and Software

Our final prototype was built using pseudo-transparency. The display is a widescreen *Xenarc 700TS* touch screen, running at 800 x 480 pixels. A *Logitech Quick Cam Pro 3000* was extended on a fixed boom and directed to the back of the screen in order to capture the fingers for display and to detect touch-cursor locations, A *Fingerworks iGesture* touch pad, capable of detecting several points of contact (http://www.fingerworks.com), was mounted behind the display to detect finger touch positions. A black matte, was placed over the iGesture pad in order to aid background subtraction of the camera image. The image from the camera and the input from the iGesture pad were registered with the screen in software. The components were connected via USB to a desktop computer, running *Windows XP*. All software was implemented in C++, JNI, and Java. Figure 6 demonstrates the technology layers of our prototype, as well as a future vision in which the camera is replaced by a flat imaging device positioned on the back of the LucidTouch.

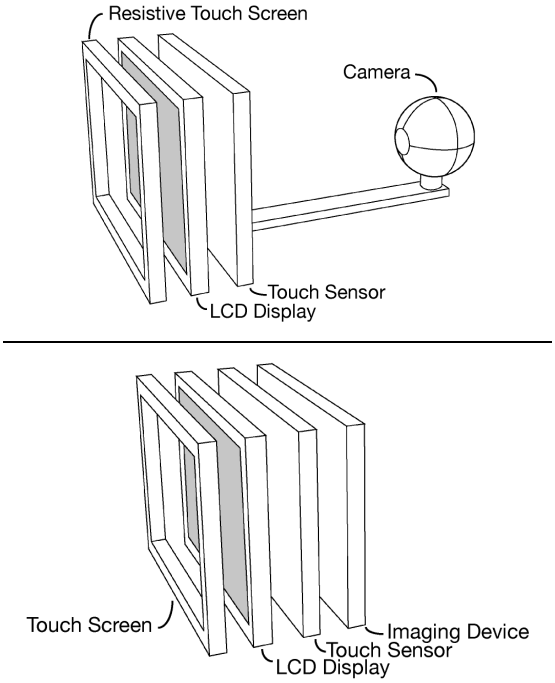

Figure 6. (a) Schematic view of the layers in our current camera-based LucidTouch prototype, (b) our envisioned device, in which the camera and boom are replaced with a micro-array imaging device capable of capturing an image of the hands.

The LucidTouch had several limitations. First, because the resistive touch screen is capable of detecting only a single point of contact, only one thumb can touch the front of the display at a time. Second, because the camera was mounted such that it faced downward towards the hand, the image seen by the user did not accurately approximate a see-through display: the back of the hands were shown, and so movements away from the display (and therefore towards the camera) made the fingers larger, rather than smaller. Third, because of the need to register the camera image and process alpha levels for various on-screen objects, there was a noticeable lag (~100ms) in the camera image. Finally, no processing was done in the device itself, but rather it was tethered to a desktop computer via VGA, USB, and power cables, limiting the portability of the device.

Although it allowed us to explore the design space, the clear limitations of a camera-based solution suggest a strong need for the alternative imaging device envisioned in Figure 6 (b). We now turn our attention to this issue.

Alternative Sensing Technologies

Although none has yet been implemented, a number of technologies show promise as an alternative to a boom-mounted camera. We now briefly describe some of these technologies.

Capacitive Array

A capacitive array, such as that in [20], can detect positions of fingers both on and above the surface of a touchpad. Although the iGestuer pad we used in the LucidTouch employs such an array, it was tuned by the manufacturer so as to detect positions only when the fingers are actually touching the surfaceof the pad. Retuning such a pad would allow for some rudimentary imaging of the fingers behind the device.

LED Array

An array of IR LED's, alternating between flashing and detecting one another's flashes, could be used to image the fingers. IR light reflected by the hand on to the non-flashing LED's could provide position information. The resolution of the image would be limited by the density of the array.

Stereo Cameras

The system seen in [30] provides 3D position information of the hands. Small cameras could be embedded in the body of the device. In addition to a 2-D image of the hand, such a 3D imaging solution would also provide depth information, allowing for additional information for designers.

Micro-Imaging Array

A micro-array imaging array, similar to that described in US patent application # 10/873,575, is an array of tiny, 1-pixel light sensors. Such an array could be embedded in the back of a device, and provide a 2D map of finger locations.

Despite its limitations, our implementation of the LucidTouch was sufficient to design several interaction scenarios, allowing us to begin to validate our concept with a user population, and to begin work on designing a multi-point, direct touch UI.

INTERACTION DESIGN

In order to investigate interaction with the LucidTouch, we explored the design of interaction techniques for two of the most common input tasks: entering text and dragging objects across the screen. We also set about implementing a common bimanual scenario: virtual map navigation.

Text Entry

Although several soft keyboard designs have been proposed for stylus or single finger input [28, 31, 32], the ability of the LucidTouch to detect multiple points of contact suggests that better performance could be achieved. The iPhone device has been demonstrated with a soft version of a two-thumb keyboard [18]. As a starting point, we implemented a similar keyboard, as shown in Figure 7, left. As has been well explored in text entry research, full QWERTY keyboards are much faster than two-thumb keyboards, in part because fingers move in parallel to reach future letters. We wished to leverage this in a soft keyboard for the LucidTouch, while maintaining the relative positions of the fingers and keys for touch typists. The result is the keyboard shown in Figure 7, right: each half of the keyboard is positioned on the screen above the appropriate hand. As in touch typing on a physical QWERTY, the spacebar is intended for use by the thumbs.

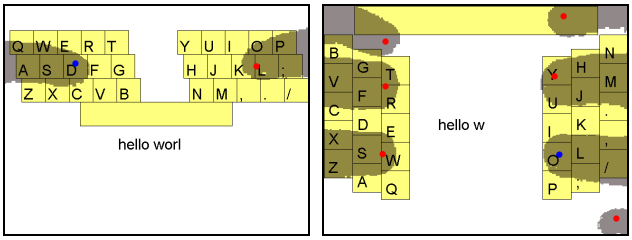

Figure 7. Two soft keyboards on the LucidTouch. Left: a traditional QWERTY keyboard layout. Right: the QWERTY keyboard reoriented so as to maintain the usual 'home row'.

It is not immediately clear which of these two approaches is superior: the first version maintains the visual position of the QWERTY keys, while making it difficult to use all 10 fingers to type. While we hypothesise that the second QWERTY implementation will leverage motor memory of touch typists, it is visually jarring and does not lend itself to visual search by those familiar with the QWERTY layout, in the visual space is quite different than in the traditional layout.

Coordinated Bimanual Input

As we have discussed, multi-point direct input schemes frequently require coordinated actions between the two hands. The LucidTouch, in particular, may have a special need for coordination. Because it falls in to the middle of the sizes of transparent devices we described earlier, all areas of the screen are within reach of the fingers, but no one finger can reach the whole screen without moving the hands. As such, simple screen-wide operations, such as dragging objects from one side of the screen to the other, require a 'hand off' mechanism to seamlessly transition between fingers. Our simple drag technique, shown in Figure 8, causes small objects being dragged across the screen to expand as they reach the centre. In so doing, they provide a sufficiently large target for the other hand to reach and grab them.

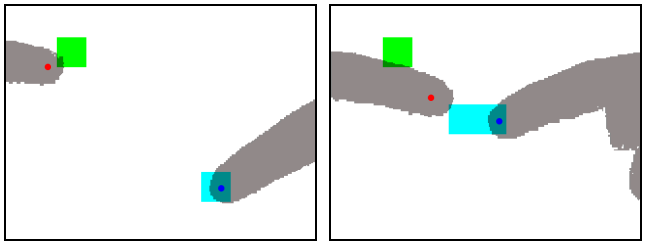

Figure 8. Left: the user selects an object. Right: as the object is dragged towards the centre, it expands to support easier hand-off between the fingers.

Map Browser

The map browser application features interaction similar to that implemented by Ullmer and Ishii in [26]: the user selects either one or two points on the map. If one point is selected, and then dragged, the map is translated, keeping the position of the finger constant on the map. If two points are selected, the map moves, rotates, and translates with the fingers such that the position of both of the fingers on the map remains fixed. Figure 9 illustrates the interaction.

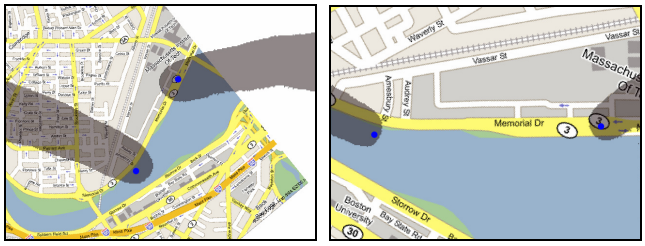

Figure 9. Our map browser application: the map rotates, translates, and scales to remain under the user's fingers.

USER EVALUATION

In order to evaluate the LucidTouch and our interaction designs, 6 participants were given the opportunity to interact with the device, and were asked targeted and open-ended questions. The goal of this evaluation was not to get feedback on particular use scenario or to evaluate user performance, but rather to seek initial impressions of various aspects of the system: the use of pseudo-transparency, the form factor, the use of touch input on the front versus the back of the device, and our interaction techniques. Wherever possible, users were asked to compare options rather than to give impressions, in order to reduce known effects for participants wishing to 'please' the experimenter. Although a small group, the topics of consensus and disagreement provided valuable insights.

Participants

Six participants between the ages of 26 and 43 were recruited from other parts of our lab. Five had experience working with a direct touch device, 1 had never used such a device. Their education level varied from undergraduate to post-graduate. No compensation was offered.

Procedure

Participants sat at a table with our LucidTouch prototype, and were asked to lift it between their hands. Several task applications were loaded and explained to the user. While performing a required task, users were asked various questions. The ordering of scenarios and of conditions within each scenario was balanced between participants.

The tasks were selected for their coverage of those required to build a basic UI, as well as the need for some multi-touch interaction; we will review each in turn.

Map Browsing

The map browsing task, described above, presents a map of the city of Cambridge, Massachusetts to the user. Participants were asked to find our lab on the map. This task was included in order to evaluate various aspects of the LucidTouch for traditional multi-touch interactions.

The task was presented four times: in the first two, participants were asked to manipulate the map using one thumb on the front of the device, and then one thumb on the front and one finger on the back (the front-touch only condition was limited to a single thumb by the hardware). In the remaining two presentations, all input was made through the back of the device: with the usual visual feedback of the fingers and touch-cursors, and then with pseudo-transparency disabled, showing only touch-cursors. Participants were asked for their preferences for visual feedback and input surface, and to describe their experience in using the LucidTouch to navigate the map.

Text Entry

For this task, participants were asked to enter their name using the two soft keyboards. This task was included in order to evaluate various aspects of the LucidTouch for the use of soft keyboards. Additionally, by requiring the users to select buttons for text entry, we hoped to elicit feedback on the use of the device for land-on selection.

Participants were asked to conduct this task with the two keyboards shown in Figure 7. For each layout, users were asked to type their names once using both the front (thumbs) only and back (fingers) only respectively. When entering on the back surface, the inverted keyboard was presented twice: once with the pseudo-transparency enabled, and once with it disabled, for a total of 5 conditions for the task completed by each participant. Before entering text, the use of the keyboards was explained to the participants. The participants were asked to give feedback on their preference for input side, keyboard layout, and hand visualisation. They were also specifically asked if they were able to intuitively locate keys on the QWERTY and inverted-QWERTY layouts.

Drag & Dock

There are three canonical tasks in a GUI: selection (moving a cursor to an object), docking (selection + drag of the object to a location), and path following (eg: steering through levels of a hierarchical menu). In this scenario, we asked users to repeatedly perform a docking task. This was done for two reasons: first, because this includes two of the three canonical tasks (docking and selection); second, because it allowed us to evaluate our two-handed drag technique, described above.

Participants were asked to repeatedly perform the docking task. We modified the task so that it could be completed in two ways: the user could select the object and move it to the dock, or they could select both the object and dock, and move them together, depending on instructions from the experimenter. Once the two objects were positioned so that they overlapped, the user deselected them both, and their positions were reset to opposite sides of the display. Each participant was asked to conduct several dockings using both techniques under each of four conditions: giving input using only their thumbs on the front of the device, the thumb of one hand on the front and the fingers of the back, or using the fingers of both hands on the back. In this final condition, they were asked to do this with the video image of the fingers on and also with it turned off, for a total of 8 different task conditions.

Participants were asked to give feedback on their preferences for visual feedback, input side, docking method (either dragging both or just the target object), and general feedback on the LucidTouch device in performing this task.

Results

Generally, users claimed to find pseudo-transparency helpful: 4 of 6 participants reported difficulty in keeping track of which activation point corresponded to which finger when this feedback was disabled. Their preference for whether they wished to have it enabled, however, and their preference for input surface, depended on the task.

Map Browsing

In this scenario, participants were asked to repeatedly find our lab on the map, while reporting preferences for touch-face (front or back) and for visualisation.

All participants were able to complete the task. On the issue of touch-side preference, 5 of the 6 participants preferred making input on the back of the device, while none preferred the front, and 1 had no preference. Of the 5, 2 noted that, if the device were capable of detecting multiple-touchpoints on the front, they would prefer to use that side for input. Of the 3 participants who preferred touching on the back, 2 specifically mentioned the issue of occlusion as a deciding factor. When asked about their preferences for visual feedback when touching on the back of the display, 3 participants said that they preferred to see the pseudo-transparency in addition to the activation points, while the other 3 preferred to see only the activation points.

Text Entry

In this scenario, participants were asked to enter their names using two different soft keyboards, while reporting preferences for the keyboard, input side, and visual feedback.

All participants were able to complete the task. While touching on the back, 4 of 6 participants preferred that the video overlay be present; 2 of these 4 specifically noted that it was more helpful for this scenario than for the map browsing scenario. 5 of 6 participants preferred giving input on the front of the device while using the QWERTY layout, while 4 of 6 preferred the back while using the inverted QWERTY. 5 of 6 participants reported that it was easy to locate keys on the inverted QWERTY layout once it had been explained to them. The participants were evenly split on preference: 3 preferred the QWERTY layout, while the other 3 preferred the inverted-QWERTY. Without prompting, 3 of 6 participants expressed a desire for physical feedback of touch input.

Drag & Dock

In this scenario, participants were asked to repeatedly perform a docking task, while reporting preferences for input surface, visualisation, and docking method (bringing objects together or dragging a single object across the screen).

4 of 6 participants preferred that the video overlay of the hands be present; 4 of 6 shared this preference with their preference in the text entry task, the remaining expressed the opposite preference. 3 of 6 participants preferred giving input on the back of the device, 2 preferred the front, while the remaining participant had no preference. For docking method, 2 participants preferred to drag and hand-off a single object, 1 preferred to move two objects to the centre of the display, while the other 3 participants had no preference. Table 1 summarises the results of questions common to each of the three tasks.

Table 1. Results for user preference for input surface and presence or absence of pseudo-transparency for each task in the experiment.

	Input Surface			***Pseudo-Transparency***	
Task	*Front*	*Back*	*No Pref.*	*On*	*Off*
Map	33%	50%	17%	50%	50%
QWERTY	83%	17%	0	67%	33%
Inverted-QWERTY	33%	67%	0		
Drag & Dock	33%	50%	17%	67%	33%

Discussion

The results of our evaluation generally fall into three categories: selecting a side of the device for touch input, pseudo-transparency for touches on the back, and results specific to an interaction scenario. We will discuss each in turn.

Touch-Side Preferences

All of the tasks could be completed by giving input to either the front or the back of the device. In deciding their preference for touch side, users generally needed to consider and balance several factors: occlusion (when touching the front), the capability of multi-touch (available only when touching the back), a difference in visualisation (direct-touch on the front, or input on the back with either pseudo-transparency or only touch-cursors), and difference in fingers (thumbs for the front, fingers for the back). In particular, the issue of occlusion was mentioned by 2 users. The lack of precision of direct-touch input to the front of the device was well expressed by one user who reported that "my fat thumb keeps pressing the wrong button".

The general receptiveness of users to touching the back of the device is encouraging, especially given that the tasks included in the study could be completed by giving input to either side. Given the clear advantages of giving input to the back of the device for tasks not included in this study, such as selecting small targets or giving multi-touch input, this receptiveness clearly suggests the potential for adoption of a see-through device.

Pseudo-Transparency

In all 3 tasks, when giving input on the back of the device, participants were presented with two types of visual feedback: pseudo-transparency of the fingers with video overlay with touch-cursors, or the touch-cursors alone. 4 of 6 participants reported difficulty in understanding which touch-cursor corresponded to which finger in the absence of pseudo-transparency. While completing the tasks, 4 of 6 preferred that pseudo-transparency be included for the two tasks that required precision pointing (typing and docking). For the remaining task, only 3 of 6 preferred that the video-overlays be present. 2 of the remaining 3 noted that the precise location of the touch points was not necessary for the task, while all 3 described the pseudo-transparency as making the task more difficult.

The clear implication from these results is that there is a need to vary the pseudo-transparency between, and possibly within, an application, or to otherwise modify the rendering so as to maximize awareness while minimizing intrusiveness. In particular, we note that the 100ms lag in our system may have exaggerated the level of distraction. In tasks requiring precise input locations, our study suggests that displaying only the touch-cursors prior to touching is not sufficient, but that, for some tasks, pseudo-transparency may be distracting. It may well be, however, that this distraction could be reduced without eliminating the transparency entirely.

Interaction Technique Feedback

It is not surprising that, for the text entry task, users would prefer touching on the front for the QWERTY keyboard and on the back for the reverse-QWERTY, given that the layouts were optimised for that side. That 3 participants expressed frustration at the lack of tactile feedback is also unsurprising, but suggests a possible augmentation of the LucidTouch: the addition of a touchpad contoured to the shape of on-screen content to the back of the device might provide this feedback, while continuing to provide a continuous input surface. The potential for this contoured surface is an advantage of a see-through device, since providing this contour would not deform the screen image.

Finally, that user preference was split between the two methods of completing the docking task, dragging and handing-off a single object, or dragging both objects to the centre of the display, is encouraging. The first represents openness to a medium-sized device, since some tasks will require a hand-off. The second demonstrates openness to a multi-touch portable device, since it requires each of the hands to perform a portion of the task.

CONCLUSIONS AND FUTURE WORK

Based on our own experiences in working with and developing for the LucidTouch, along with the results of our user study, it is clear that pseudo-transparency presents a compelling paradigm for future development. There remain several open questions, however, which need to be addressed before the development of a successful platform.

First, while there was clearly positive feedback in our user evaluation on the usefulness of touching on the back of the display, reactions to the pseudo-transparency were mixed. Particularly noteworthy was that some users preferred the video image of the hands in some situations, while wishing it could be disabled in others. This suggests the need to vary the feedback, or to otherwise modify the rendering of the fingers on the display. How this should be done, and when, is left to future research.

Additionally, participants gave mixed feedback on the usefulness of the two soft keyboard implementations. While some preferred one over the other, most agreed that there were advantages and disadvantages to each design. Further exploring the benefits of each, and perhaps the development other techniques for text entry, is an open question for future research. Of particular interest may be an investigation of inverted handwriting or other stroke recognition, and how the inversion of the input space, affects user performances and preferences for input.

As we discussed, there are three categories of transparent device sizes: a small device, where all fingers can reach all parts of the screen, the medium-sized device, where all points of the screen are reachable while holding with both hands, but not by all fingers, and finally a larger device, where reaching many points of the display would require a hand to be moved. Our implementation fell into the second category, and so our interactions were developed for that platform. It may well be the case that interactions would be significantly different when developed for the smaller device. In particular, we note our earlier discussion of bimanual input , and our hypothesis that these results might differ when making input with only the fingers. If accurate, this would imply the need for different bimanual interaction techniques for each size-class of transparent device.

One of the main limitations of our implementation of the LucidTouch is that the video image is captured from above the hands. Not only does this limit the feeling of a 'transparent' display, but it also limits the field of view of the camera to the area between it and the display. As we discussed, future technologies might enable the capture device to be embedded within the device, with a field of view extending out and away from its surface. To maintain pseudo-transparency, it would be necessary that this device have a depth of field limited to the area immediately behind the display. If this depth of field could be dynamically adjusted, however, the device would be capable of capturing images beyond the display, similar in feel to looking at a digital camera's LCD. It would be possible, with such a device, to leverage some of the results of augmented reality, where a real-world view is enhanced with digital overlays. This application is left for future work.

Finally, the most clear avenue for future research is the development of a fully realised pseudo-transparent display, overcoming the limitations of the +, and leveraging cutting-edge technologies. By eliminating some of the technology artefacts introduced by our prototype, a more thorough exploration of the concepts of pseudo-transparency, see-through devices, and multi-touch mobile interaction would be enabled.

ACKNOWLEDGMENTS

We thank Garrett Winberg and Paul Dietz for providing hardware, Turner Whitted and Darren Leigh for hardware design discussions, Michael McGuffin for research assistance, members of the MERL and Microsoft Research labs for valuable discussions, and our experimental participants.

REFERENCES

1. Albinsson, P. and Zhai, S. 2003. High precision touch screen interaction. *SIGCHI Conference on Human Factors in Computing*. p.105-112.

2. Azuma, R.T. (1997). A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments* 6(4) (August 1997). p. 355-385.

3. Balakrishnan, R. and Kurtenbach, G. (1999). Exploring bimanual camera control and object manipulation in 3D graphics interfaces. *Proceedings of CHI '99*. p. 56-62.

4. Baudisch, P. and Gutwin, C. Multiblending: displaying overlapping windows simultaneously without the drawbacks of alpha blending. *Proceeding of CHI 2004*, p. 367-374.

5. Benko, H., Wilson, A., and Baudisch, P. (2006). Precise selection techniques for multi-touch screens. *Proceedings of CHI '06.* p. 1263-1272.
6. Buxton, W. (1990). A Three-State Model of Graphical Input. *Human-Computer Interaction - INTERACT '90.* p. 449-456.
7. Buxton, W. and Myers, B.A. (1986). A study in two-handed input. *Proceedings of CHI '86.* p. 321-326.
8. Card, S.K., English, W.K., Burr, B.J. (1978). Evaluation of Mouse, Rate-controlled Isometric Joystick, Step Keys, and Text Keys for Text Selection on a CRT. *Ergonomics* 21(8). p. 601-613.
9. Dietz, P. and Leigh, D. (2001). DiamondTouch: a multi-user touch technology. *Proceedings of UIST '01*. p. 219-226.
10. Esenther, A., Ryall, K., (2006). Fluid DTMouse: Better Mouse Support for Touch-Based Interactions. *Proceedings of AVI '06*. p. 112- 115.
11. Forlines, C., Shen, C., (2005). DTLens: Multi-User Tabletop Spatial Data Exploration. *Proceedings of UIST '05*. p. 119-122.
12. Forlines, C., Vogel, D., Balakrishnan, R. (2006). HybridPointing: Fluid switching between absolute and relative pointing with a direct input device. *Proceedings of UIST '06*. p. 211-220.
13. Guiard, Y. (1987) Asymmetric Division of Labor in Human Skilled Bimanual Action: The Kinematic Chain as a Model. *Journal of Motor Behavior*, 19. p. 486-517.
14. Han, J. (2005). Low-cost multi-touch sensing through frustrated total internal reflection. *Proceedings of UIST '05*. p. 115-118.
15. Hiraoka, S., Miyamoto, I., Tomimatsu, K. (2003) Behind Touch, a Text Input Method for Mobile Phones by The Back and Tactile Sense Interface. *Information Processing Society of Japan, Interaction 2003*. p. 131-138.
16. Kabbash, P., Buxton, W., and Sellen, A. (1994). Two-handed input in a compound task. *Proceedings of CHI '94*. p. 417-423.
17. Leganchuk, A., Zhai, S., and Buxton, W. (1998). Manual and cognitive benefits of two-handed input: an experimental study. *ACM Trans. Comput.-Hum. Interact.* 5 (4) (Dec. 1998). p. 326-359.
18. MacKenzie, I. S., Soukoreff, R. W. (2002). A model of two-thumb text entry. *Proceedings of GI '02*. p. 117-124.
19. Potter, R., Weldon, L., and Shneiderman, B. (1988). Improving the accuracy of touch screens: an experimental evaluation of three strategies. *Proceedings of CHI '88*. p. 27-32.
20. Rekimoto, J. (2002). SmartSkin: an infrastructure for freehand manipulation on interactive surfaces. *Proceedings of CHI '02*. p. 113-120.
21. Schilit, B., Adams, N.I., Want, R. (1994). Context-Aware Computing Applications. *Proceedings of the Workshop on Mobile Computing Systems and Applications*. p. 85-90.
22. Sears, A. and Shneiderman, B. (1991). High precision touchscreens: design strategies and comparisons with a mouse. *International Journal of Man-Machine Studies*, 34 (4). p. 593-613..
23. Sugimoto, M. Hiroki, K. (2006). HybridTouch: an intuitive manipulation technique for PDAs using their front and rear surfaces. *Proceedings of MobileHCI '06,* p. 137-140.
24. Tang, A., Neustaedter, C., Greenberg, S. (2004). VideoArms: Supporting Remote Embodiment in Groupware. *Video Proceedings of CSCW '04*.
25. Tang, J. C. and Minneman, S. L. (1990). VideoDraw: a video interface for collaborative drawing. *Proceedings of CHI '90*. p. 313-320.
26. Ullmer, B. and Ishii, H. (1997). The metaDESK: models and prototypes for tangible user interfaces *Proceedings of UIST '97*. p. 223-232.
27. Vogel, D. and Baudisch, P. Shift: A Technique for Operating Pen-Based Interfaces Using Touch. To appear in *Proceedings of CHI 2007*. (in press).
28. Ward, D. J., Blackwell, A. F., MacKay, D. J. (2000). Dasher - a data entry interface using continuous gestures and language models. *Proceedings of UIST '00*. p. 129-137.
29. Wigdor, D., Leigh, D., Forlines, C., Shipman, S., Barnwell, J., Balakrishnan, R., Shen, C. (2006). Under the table interaction. *Proceedings of UIST '06*. p. 259-268.
30. Wilson, A. D. 2004. TouchLight: an imaging touch screen and display for gesture-based interaction. *Proceedings of ICMI '04*. p. 69-76.
31. Zhai, S., Hunter, M., Smith, B. A. (2000). The metropolis keyboard - an exploration of quantitative techniques for virtual keyboard design. *Proceedings of UIST '00*. p. 119-128.
32. Zhai, S., Kristensson, P. (2003). Shorthand writing on stylus keyboard. *Proceedings of CHI '03*. p. 97-104.