Submitted to CHI'98, Los Angeles, CA., April 18-23, 1998

The Design and Use of Squeezable Computers: An Exploration of Manipulative User Interfaces

Beverly L. Harrison **Ken Fishkin** **Anuj Gujar** **Carlos Mochon*** **Roy Want**

Xerox PARC
3333 Coyote Hill Road,
Palo Alto, California, USA 94304
{beverly, fishkin, agujar, cmochon, want}@parc.xerox.com

ABSTRACT
This paper reports on the design and use of tactile user interfaces embedded within or wrapped around the devices that they control. We discuss three different interaction prototypes which we built. These interfaces were embedded onto two handheld devices of dramatically different form factors. We describe the design and implementation challenges, and user feedback and reactions to these prototypes. Implications for future design in the area of manipulative or haptic user interfaces are highlighted.

KEYWORDS: physical UIs, tactile UIs, haptic UIs, evaluation, pressure sensors, user interface design, interaction technology.

INTRODUCTION
Over the past 5 years there has been increasing interest in augmented reality and physically–based user interfaces [4, 6, 7, 8, 10, 12, 15, 16, 17]. A goal of these emerging projects is to seamlessly blend the affordances and strengths of physically manipulatable objects with virtual environments or artifacts, thereby leveraging the particular strengths of each. Typically this integration exists in the form of physical input devices (e.g., phicons [7], bricks [4]) attached to and tracking with electronic graphical objects. Movement of the physical object or handle results in a corresponding movement or reorientation of the associated electronic item. This mapping is further reinforced by tightly coupling the placement of the physical objects relative to the electronic items on a flat table-like display surface.

Another approach has been to use standard monitors or even stereoscopic monitors with more realistic input devices [6, 8]. In these cases, unique physical input devices are cleverly matched to the requirements of the specific application domain (e.g., MRIs, remote telerobotic control). The affordances of the input devices are well matched to the virtual representation of the object that they represent. Designers of commercial video games have been taking such an approach to user interface manipulation since the invention of the Data Glove™, and more recently with such games as flight simulation and car racing, where the UI is controlled by steering throttles or steering wheels. Again, in these instances a specialized input device controls a separate display.

These extensions to graphical user interfaces seem logical in view of the widespread support and acceptance of direct manipulation interfaces [11] and of real world metaphors such as trash cans and file folders [12]. We argue that such physical user interface manipulators are the logical realization in making the next UI metaphor the real world itself, real objects having real properties which are linked to or embedded in the virtual artifacts that they control. Furthermore, we conjecture that this increasing interest from the UI design community reflects what will become the 4th generation user interface paradigm

What are the implications of adopting such a paradigm for the next generation of user interfaces? The goal of this paper is to share our experiences in formulating useful frameworks for thinking about this new class of user interface, and to share what we have learned from designing, building, and using several prototype physical interface mechanisms.

Our work differs from the above discussed work on physical handles and unique input devices in at least one fundamental way. We are investigating scenarios where the physical manipulations are *directly* integrated with the device or artifact that is being controlled. They are not input devices per se, but rather are embedded within the artifact or within the device casing. Physical manipulations of the device being controlled are recognized and interpreted by that self-same device (i.e., the manipulations are the "input device").

Several research prototypes have been influential and reflect elements of this "embedded physicality" approach. Fitzmaurice [3], Rekimoto [9], and Small & Ishii [12] attached sensors to small handheld displays and subsequently used these displays to "scroll" through or view a larger virtual space. Movement of the display is mapped

* - This author was an intern from the MIT Physics Dept.

to corresponding movements in the virtual space, such as changes to the view perspective [9] or to the degree of magnification [12]. These prototypes demonstrated the intuitiveness of this embedded physicality approach, unfortunately none were subsequently user tested or evaluated. Additionally, no conceptual framework or systematic principles were proposed to guide other designers. The work we report here incorporates different manipulations from these prior examples to further improve our understanding of the breadth and potential of these new kinds of interactions.

We have been investigating both *user-initiated* physical manipulations and *environmentally-sensed* physical changes, in the interests of exploring a broad range of possible applications and designs.

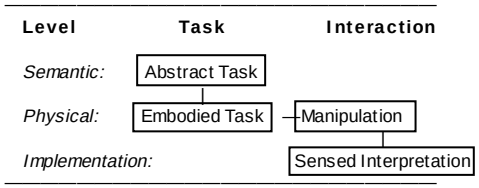


FIGURE 1. Conceptual Framework of Design Process for Physically manipulatable UIs

CONCEPTUAL FRAMEWORK

We now consider the elements which influence how and when a physically manipulatable interface will operate. First, the users start with a conception of which task or operation they wish to accomplish at a device-independent level of abstraction (*Abstract Task* in Figure 1), for example, "go back to the previous page". The user interface will be instructed to perform this action by means of some physical interaction made by the user and, more importantly, understood by the user to correspond to that task goal. For example, "go back to the previous page" might correspond to flicking the left corner of the document with their finger from left to right as they would do when turning a real physical page back *(Kinesthetic Manipulation*) [. (The elemental units of kinesthetic manipulation are known as kinemes). There are two components to well-designed "methods" reflected here: choice of kinemes and choice of semantic bindings.

Kinesthetic manipulations are constrained by what is physically possible and comfortable. Within the context of computer systems, they are additionally constrained by limitations on the magnitudes of manipulations or movements and the elimination of potentially embarrassing movements. (Imagine physical manipulations of computational devices while on a plane, for example). Semantic bindings have a number of constraints including: obvious metaphors, existing social practice or convention, ease of learning and remembering, capabilities of the human operator, and capabilities of the technology [2].

Assuming the users understand a particular manipulation as the one which corresponds to their task goal, and they perform this manipulation (e.g., flicking the left corner), the device must sense and interpret this manipulation (*Sensed Manipulations*, Figure 1). Two critical constraints apply. First, the manipulation must be sense-able to the device in an unambiguous way with the appropriate parameters if any. Second, there must be mechanisms either internal to the technology or explicit in the interface design to differentiate intentional versus inadvertent actions, for example, higher threshold values for intentional actions. A combination of the kinesthetic manipulation and the sensing capabilities of the embedded technology allow the computational device to perform its own *Embodied Task*, which is then executed with appropriate state change feedback provided to the users.

The above framework highlights the major physical manipulation interface design stages which play key roles in determining the success or failure of particular design choices. The boxes in Figure 1 represent the components and the corresponding constraints imposed at each stage of the interaction interpretation and execution. The transitions between each of the components reflect design choices for mappings and semantic bindings. Using this model, one can trace out the conceptually associated physical manipulation, semantic binding, sensing technology, executed task, and status feedback based on and derived from the users' task goal. This helps to avoid designs based on technology-centric perspectives which do not tie in to real tasks or real user needs. Potential problems can more easily be traced back to the position where mismatches occurred – between user model/goal and associated kinesthetic manipulation, between kinesthetic manipulation and sensing technology, between sensing technology and the contextualized execution of the task, or at the status feedback stage.

This framework also allows us to derive rules and design principles at each component stage which apply specifically to the creation of physically manipulatable interfaces. These will be discussed below in the context of the manipulation interactions we embedded into several prototype interfaces. These general design principles and the conceptual framework are discussed in detail in [2].

THE "SQUEEZABLE COMPUTER" PROJECT

In order to determine whether the above framework was at all helpful and to increase our understanding of physical manipulation interfaces, we began a research program aimed at designing, building, and testing a variety of embedded sensor-based technologies which supported user interaction through kinemes. A multi-disciplinary team was put together which included expertise in hardware and sensor technology, software engineering, user interface design, ergonomics and user evaluation.

Choosing the User Task

We chose several diverse and previously untried user tasks. The diversity allows us to try alternative kinds of manipulations, different types of sensing technologies, and better test our framework. In starting with untried

user tasks, we are contributing to the general body of knowledge about physically manipulatable interfaces. Additionally, we selected tasks which represented different types of interaction, in particular, active user interaction via explicit physical manipulations and passive user interaction. Finally, we selected tasks which were relevant for other PARC research groups who were implementing applications for portable document devices [10]. For this latter reason, we focused on portable pen-based systems.

We chose several simple tasks: navigation within a book or document, navigation through long sequential lists, and sensing handedness with optimization for annotation. (Note that the latter "task" is actually implicit in usage of the device, what we term passive user interaction.)

Navigation within a book or document was divided into several simple (and hopefully obvious) sub-tasks: turning to the next page, turning to the previous page, and moving forward or backwards in large "chunks" relative to the beginning and ending of the book or document. These tasks were related directly to the needs of the portable document device research projects.

Generally, users conceptualize lists in different ways than books or documents (though similar navigation techniques could be used). We decided to create a Rolodex metaphor–based technique for list navigation (see Figure 4a), where navigation was continuous instead of discrete (as in page turning). The circular list is manipulated by turning the Rolodex, while the direction of the turn determines whether the cards flip from A to Z or from Z to A. Again, such navigation through lists of items was a dominant document activity expected to play a significant role in the portable document device research projects.

Sensing handedness meant that text or graphics would be moved towards the non-dominant hand while screen space would be maximized on the opposite side (next to the hand holding the stylus). This strategy is appropriate for maximizing legibility of the text while holding the stylus and annotating adjacent to the text. This implicit "task" provided us with a unique interface optimization for pen-based systems when annotation plays a key role, which was of particular relevance to the software applications being developed. In general, we were also interested in exploring different mechanisms for determining handedness unobtrusively.

Choosing the Kinesthetic Manipulations

Semantic Binding

We decided to use well understood real world manipulations for the document navigation task. In particular, a flick on the upper right corner from right to left would indicate "forward one page". A flick on the upper left corner from left to right would indicate "back one page". Moving forward or backward by chunks relative to the beginning or ending of a document was more difficult to represent for virtual documents. We decided to use a grasping manipulation at the top of the device, where the relative position of the grasp determined the relative position within the document. Far left corresponded to page 1 and far right corresponded to the last page. While this was not tightly related to known real world metaphors, appealed to the well-known GUI metaphor of the scroll bar.

For the list navigation task, we decided to rely on the Rolodex metaphor, which is often associated with address lists (i.e., a circular list of cards on a roller). In this case, turning a circular list towards the user would begin flipping through from A towards Z (assuming an alphabetized list) and vice-versa. Turning "harder" (i.e., to a larger extreme) moves faster through the list. To stop at or select a particular item, the user turns the roller moving through the list until the desired item is close at hand and then the user positions it at the top so that the item can be grasped and removed, for example.

Finally, we have the handedness detection "task". In this case, we do not need a semantic binding since this is not a user-actuated or explicitly invoked function. Passive user interactions (or passive kinemes) do not require semantic bindings.

Kinemes

Based upon what we felt to be reasonable semantic bindings, we then determined what kinesthetic manipulations (or kinemes) the user would need to invoke the appropriate command sequence. "Reasonable semantic bindings" were determined through discussion and informal design review. For more complex and less obvious tasks more rigorous evaluation at this stage would ensure solid semantic binding choices before proceeding further in the design.

For the document navigation task, page turns mapped readily to a downward finger pressure, followed by a short directional stroke, ending with a finger lift. The directional stroke determines the direction of page turning. This manipulation is extremely close to page turning in physical documents, although there is less tactile feedback than when a physical page is turned. The grasping manipulation was essentially a pinch with the thumb and any finger of the same hand, at a specific position along the top of the device.

The list navigation task required users to tilt the device or the "list container" in an arc away from them or towards them to move through the list. The direction of tilt determines the direction of the scrolling (A to Z versus Z to A). Users alter the rate of list movement by the extent of tilt they apply, as presented by Rekimoto [9]. Selecting a particular item is done by first moving through the list to a location close to the item and then ceasing to tilt further (i.e., maintain the list container in a neutral or vertical position relative to that item).

Finally, for the handedness detection task we needed to understand something about how users hold and operate the intended device. Essentially, no special kineme

should be needed other than picking up the device. In general, manipulations in the "passive user interaction" category should not require any special manipulation. In this case, the manipulation is holding the portable device with one hand and the stylus with the other.

At this stage of the design, we can evaluate whether the range of kinemes defined meet kinesthetic constraints. All are easy to perform (assuming users are not physically challenged) and do not seem to be anatomically stressing. None of the proposed manipulations appear to require embarrassing or culturally offensive movements. All seem useable in a variety of situations.

Choosing the Sensed Manipulations

We now have physical manipulations specified, semantic bindings for these manipulations (if needed), and an understanding of the task goals. We next determined what sensor technologies and software were needed to support the manipulations.

The document navigation task requires that the left and right upper corner detect a finger press, the direction of a stroke, and a release of pressure. Several options are possible. Within each application where document reading occurs, a touch sensitive display can detect pressure points and their origin, determine if this aligns with a document upper corner, track the path of pressure to determine the stroke direction, and execute the appropriate page turn. Alternatively, the case of the device itself can have embedded pressure sensors on its surface, detect when it is being pressed, detect the direction of pressure from a stroke, and have the currently active application respond appropriately. We decided to try the latter approach since this would provide us with opportunities to later use the sensor technology in other application contexts and across applications. Also, we did not need to use valuable screen real estate for the large area graphics needed to display a finger operated button. Finally, this allowed us to "retrofit" pressure-sensing technology onto a normally pressure-insensitive device.

Based on our semantic mapping and derived kineme for navigation by "chunks", we again decided to use pressure sensors. To achieve a grasp with the thumb and a finger, pressure sensing strips could be attached along the back and front top edge of the device. (Recall that whether to use the top or a side and even the particular concept of using a grasp was design reviewed during the semantic binding stage). In fact, only one pressure sensing strip is actually needed to determine the press location within the strip, the second strip is redundant (and thus our final implementation used only one top-mounted pressure strip). Grasping any portion of the strip moves to a document position relative to the beginning or ending of the document, where the beginning maps to the far left of the strip and the end maps to the far right of the strip.

Finally, we examined sensor options for unobtrusively determining handedness, based upon the user simply holding the portable device in one hand and the stylus in the other [Figure 2]. Several technologies were possible. Heat sensors on either side of the device could potentially detect whether contact with a hand occurred on the right or left side of the device (based on heat from the user's hand). However, this detection would be complex since the device itself generates heat and because the surface of human physical extremities such as hands and feet generate much lower levels of heat than we originally anticipated. Another alternative was to detect properties of the writing which are unique to left handed or right handed writing styles. This is somewhat problematic since these algorithms are complex and the system can only take effect after the user has started writing. We decided to again use pressure sensing technology to determine the points of contact and derive how the device was being held (if at all). Pressure sensing pads were embedded on the device case backing at the left and right sides in alignment with positions used for holding the device.

Implementing the Device Embodied Task

At this stage of the design process, we have determined the sensor technologies and approximate locations on the device, and we know what activation of these sensors is supposed to do in terms of task goals. We then determined system software required, communication protocols for the sensors, and modified applications software. Lastly, we decided on the status feedback to the user to indicate that the manipulation had the desired effect (or some effect). For purposes of this paper and audience, we focus on the implementation details and issues which directly impacted the user interface and interaction design.

Selection of Devices

Our design criteria were that the devices chosen be handheld, support pen-based input, allow serial port input (for sensor communication), have a development environment for custom applications, and be cost effective (since we anticipated embedded hardware). Ideally, we wanted several devices with different form factors.

We decided to use two different portable devices to test our manipulations. We chose to use a palmtop computer for the page turning and handedness detection manipulations (a Casio Cassiopeia™). For the list searching manipulations, we chose a Palm Pilot™. Clearly, a number of other devices could have been selected. (see Future Research section).

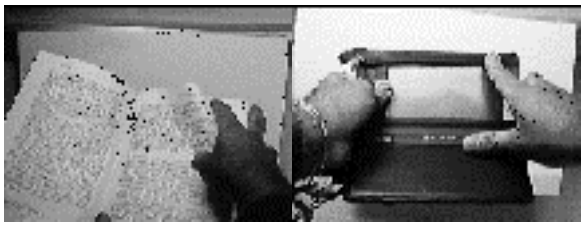
Implementation Overview - Document Navigation And Handedness

The document navigation and handedness sensing application was implemented using Windows CE. The Casio device was augmented with a network of pressure sensors. Two pressure pads on the back detect handedness, and three overlaid strips on the top edge detect the page turning manipulations (Figure 2 and 3). The pressure sensor network reports its current values through an interface connected to the RS232 port on the device. A simple communications protocol was devised, where each packet indicates the ID of the reporting sensor, and the current

value of the sensor. Packets are only sent when the value changes. Absolute values, rather than deltas, are reported, so that we can recover from dropped/damaged packets.

(detailed blowup of screen)
FIGURE 2. Handedness detection

The document reading application runs as a multi-threaded application: one thread performs user I/O, while the other monitors the sensor stream.



3(a). Physical page turning 3(b) next page manipulation
FIGURE 3.

To implement the page turning manipulations, three pressure sensors are overlaid at the top edge of the device. One type of sensor strip reports pressure, but not spatial location. The second type reports spatial location, but not pressure. Unfortunately, spatial sensors tend to have a great deal of jitter. In order to compensate for this, a third spatial sensor was used, oriented such that its values increase in an opposite direction to the second sensor. The sum of the second and third sensor values should be a constant - if they differ too much from this constant, the values are rejected. Otherwise, they are averaged, and the mean value is assigned. The {location, pressure} values are stored from the moment of pressure-down to pressure-up. If the sum of the inter-location differences is negative, the user is deemed to be stroking from right-to-left. If the sum is positive, the user is deemed to be stroking from left-to-right. If, regardless of this sum, the range of spatial locations is in a narrow range, the user is deemed to be pressing at a certain spot (i.e., the grasp gesture meaning go to that relative position in the document).

The sensor thread of the program detects handedness by determining which pressure pad (left or right) provides a high value. If either provides a high value, the user is holding the device with (at least) that hand.

Implementation Overview - List Navigation

In order to implement a tilt detection mechanism for continuous list scrolling on a handheld computer, we investigated a number of commercial sensor products. A commercial tilt-sensor design that has been successfully used in many products, is based on an electrolyte bordered on two sides by a pair of conductive plates. As the device is angled towards or away from either plate, the amount of electrolyte in contact with the plate varies. The area of fluid in contact with each plate will affect the impedance presented by the contacts of the sensor. By monitoring this impedance and converting its change into a voltage, a simple ADC interface to a microcontroller can capture the data and then process it. In our system the tilt angle is converted into a 4-bit value and transmitted to the Palm Pilot™ across an RS232 link after being prefixed with the 4-bit sensor-ID, a total of 8-bits for each tilt sample.

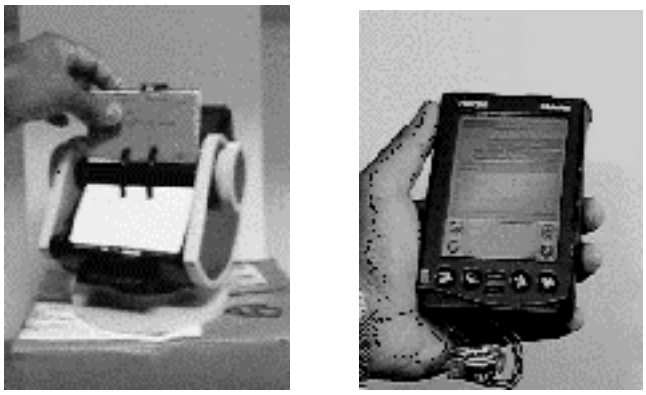


FIGURE 4. Rolodex and Physical manipulation of Pilot™

By mounting a tilt sensor of this type to the case of a Palm Pilot™, with the sensor plates parallel to the plane of the display, we were able to use the sensor readings as a crude measure of the computers orientation relative to gravity. We arranged it so that the Pilot generated a neutral reading at the 45 degree point and produced 8 readings forward and backwards from that position: 45 degrees being close to the comfortable angle that most people read from the display of the Pilot. Even though the range of angles detectable is thus very coarsely defined, we found that it has been adequate to implement and support the Rolodex–like metaphor (See Figure 4).

In addition to tilt sensing, we determined that sensors were needed to differentiate between inadvertent device movement, such as when walking with it, and intentional tilting, when the user wished to navigate. There are two possible ways of addressing this issue. The first method is to apply higher threshold values to the tilt sensing itself, thereby removing manipulations which are not of extremes and hence presumably retaining only deliberate user requests. This was infeasible in our desired application since we wished to use ranges of tilt to indicate the rate of list movement. Another possible solution is to create a second specific manipulation which indicates user intention. In our case, we decided to use an initial squeeze of the device to indicate the desire to navigate through the list, followed by a second squeeze to "grasp" the desired item, thereby ending the navigation task. Users did not have to maintain the squeezing pressure during navigation in order to avoid muscle stress. The device was padded with foam to further suggest squeezing capability.

To achieve the squeeze feature, we attached pressure sensors along both sides of the Palm Pilot™ in positions which aligned with the users' fingers and thumb (independent of which hand was holding the device). To differentiate squeezing from holding the device, we tested a number of users and derived an appropriate threshold value for the pressure sensors. (In this case using higher pressure threshold values to differentiate inadvertent from intentional action was appropriate).

Status Feedback

Status feedback is tightly coupled with the original task goals and specification. For these particular prototypes, we used visual feedback within the application context to indicate when a manipulation was sensed and had taken effect. We did not need to differentiate between sensing a manipulation and its execution since the hardware and software were optimized to ensure a fast response time.

For the page turning manipulations, the contents of the document shown on the display change. We display the current page number which indicates where in the document the user currently is. After a page turning manipulation, both the page number and contents change to reflect either the preceding or next page, depending upon the direction of the stroke. A grasp gesture will move to a new location in the document and display the new location's page number and contents.

The handedness detection is immediately visible when the user invokes any application that wants to be "handedness-aware". In the specific task we implemented, for a left handed user, text is immediately right justified and annotation space remains at the left side of the text. The reverse is true for right handed users. When both hands are used or the device is set down (i.e., no hands), the text appears centered.

The list navigation task provides two levels of user feedback. Since the device is often moved about, the "tilt feature" is initially disabled. When users wish to navigate through lists they commence movement by squeezing the device. At this point the device is tilt-enabled. At present we have a message displayed indicating this to the users (it says "Go Tilt!"). Clearly, a different message and a different means of conveying tilt-enabled would be better. Independent of this or any message, it is visually obvious when tilt-based navigation is enabled. Tilting works as described and users can see the list flipping through entries at varying rates of speed in the appropriate direction, depending upon the direction and magnitude of the tilt. The display ceases moving when the user either holds the device in the neutral position or again squeezes the device, thereby disabling tilt. (This is akin to grabbing the currently displayed item).

USAGE

A number of interesting and diverse design issues arose along the way at each stage of design and prototype development. In addition to the insights and course corrections made along the way, we learned many interesting things when we had people use the prototypes. In this section, we highlight both positive and negative feedback from our test users.

General Comments and Impressions

In general, users found the manipulations "intuitive", "cool", and "pretty obvious in terms of what was going on". Some interactions needed slight explanation or a quick demonstration so that users understood that their manipulations would be interpreted. Our test users had little or no exposure to physically embedded user interfaces and therefore often did not expect interaction with the device to be understood. Undoubtedly, conveying the basic paradigm will be necessary here in the same way that users needed to understand the conceptual foundation for direct manipulation interfaces and using a mouse. One interesting point is that once users understood the basic premise, they immediately begin to explore the range of interaction. For example, with GUIs, users try out what is "click-able" by moving around the screen with the cursor and trying. In the same manner, users working with our prototypes tried a variety of manipulations to see what the range of detectable interactions was. For example, to turn pages they tried long strokes, short strokes, quick and slow strokes, light pressure, hard pressure, and starting the stroke at different points on the device surface.

While the user explicit interactions were quickly understood, the passive interaction (handedness) was perceived as "magical". Since no explicit commands or manipulations were needed, users seemed amazed that the device recognized and optimized for handedness. They were unable to tell how this was accomplished without us explaining it. This suggests not only that passive kinemes can be powerful, but, when well integrated with the device form factor, they greatly impact the users' interaction experience. We clearly need to explore more passive manipulations to see if this is a general property for passive kinemes.

Document Navigation Task

Within the document navigation task a number of interesting usage observations were made. Because of our need to overlay pressure sensors, users now had to exert harder pressure than they anticipated for the page turning manipulations. Users try out kinemes based on their expectations of the real-world analogies. A page turn in a paperback book, for example, takes very little pressure. Users initially attempted the exact same manipulation on the device, which was too light to be sensed. They were able to quickly adjust through practice, however, all users made the same initial attempt. Improvements in our sensor solution to detect lighter page turning strokes would clearly be an improvement. In general, we believe that users will attempt to exactly replicate the analogous real–world manipulation, when those metaphors are used, and they will expect them to work. If we are striving for enriched interaction experiences, the more exactly we can support or match these expectations the better.

Users had no problem in discovering the manipulation needed for "previous page", once they had tried the "next page" manipulation. Despite slight differences in the pressure required over that of real–world interaction, users relied on extending their understanding of the real–world metaphor to guide their further assumptions about what was possible with the device–embedded interface. As in GUI design, small inconsistencies in metaphor seem to be forgiven (or perhaps it is because of experience with GUIs that this occurs).

Users needed to have the navigation by "chunks" mechanism described to them. Almost certainly this was because the device did not resemble a book, nor did the kineme map directly to that method of manipulation in the real world. Grasping along a strip which indicates relative position is unique to this interaction. Once described or briefly demonstrated, users had no trouble in remembering this or applying it.

One difficulty arose as a consequence of our implementation strategy. Since page turning strokes and grasping are both done on the same region and both use pressure sensing, it was sometimes difficult to differentiate between a very short stroke and a "wide grasp". For example, the disambiguating algorithm would sometimes guess incorrectly. This would surprise the users. This problem is not easily solved by adjusting our algorithms for sensing strokes since users' finger widths vary and we want to support short strokes. We need to re-examine (a) whether there are better sensing technologies available or a different configuration which would solve this, (b) whether minor alterations to the kineme used would help differentiate these two, or (c) whether there is a better semantic binding for navigation by chunks.

In general, the "navigation by chunks" task illustrates the tradeoff between intuitive real–world mappings which try to stay true to the real–world (and hence may be difficult to implement) versus learned mappings (which may be more easily integrated into the device). At this point, it is unclear how much learning is reasonable given that the overall goal is enriched interaction experience and intuitive interfaces.

List Navigation Task

The list navigation task revealed some additional design issues. (This provided support for our design approach of prototyping using different device form factors, different manipulations, and differing sensing technologies).

We discovered that the tilt sensing was highly sensitive and hence any movement tended to effect it. For this reason, we combined the tilt manipulation with a squeeze gesture to indicate intentional navigation. Different tilt sensing mechanisms might minimize or alleviate this problem.

Another issue was determining the range of angles for the tilt operation and the value for the neutral angle where the device remains in a resting state. We determined the neutral angle through evaluation of a number of users. The range of tilt angles was partly based on just noticeable differences (JNDs) both in terms of discernable tilt angles and in terms of discernable list scrolling speeds. Range of perceptible motion is clearly an important determinant in setting and assigning values for command parameters. At present we have 6 different rates of scrolling based on 16 tilt angles.

One outcome of user testing was the difficulty in stopping at a particular entry within the list. Users would scroll quickly to the correct general area, then attempt to scroll slowly to the desired entry. We now believe that our slow scrolling speed is still set too high as users tend to overshoot the target item. In general, fine tuning is characteristic of continuously issued gestural commands which control rate and/or direction of a corresponding action. We are investigating this issue further to determine how much individual differences amongst users effects ability to precisely control list manipulation. This suggests that some "layered manipulations" may be useful, with one type of manipulation for coarsely specified actions, followed by a second manipulation for finely specified actions.

Finally, as a consequence of using tilt to control list navigation, display visibility was an issue. In particular, we avoided use of extreme angles of tilt since the Palm Pilot™ display was not readable at these angles. Different devices and/or displays have different viewing angle restrictions with must be taken into account if the display is the primary feedback mechanism or if the display plays a central role in the task goal.

Handedness Detection

The passive manipulation used in the detection of handedness worked amazingly well. It seamlessly detected and responded correctly and users did not need to alter their usage of the device in any way from what seemed natural. All users remarked on the "magical" nature of this feature.

Although there were virtually no problems with the manipulation itself, we did test a number of users to fine-tune the placement of pressure pads to accommodate different sized hands, slight differences in method for holding the device, and whether the left and right hand were used in exactly the same positions.

One aspect of this interaction that still requires adjustment is the changeover from one hand to another. If users momentarily shift the device from one hand to the other, the contents of the screen immediately move as well. One improvement is to briefly delay the change to determine that the user is not merely rapidly shifting the device to be more comfortable. Determining a time duration for detecting a "resting state" versus a "transient and temporary change" might improve the current interface.

FUTURE DIRECTIONS

In this paper we have outlined a design process framework applied to three specific cases of physical manipulation prototypes. We discussed the methods for creating "good" designs and of identifying problems using this framework. Finally, we briefly outlined some of the results of user testing of these prototypes. Interesting research remains to be done.

Obvious extensions are to implement other novel manipulations, new semantic bindings, and test out new sensor technologies. We are interested in further exploring user explicit manipulations as well as seamlessly sensed passive manipulations, with a goal of better understanding this new paradigm and enriching the user's interaction experience.

This new type of interaction can be very dependent on the form factor of the device being augmented. We are interested in incorporating new manipulations into a number of different devices including tablet computers, conventional scanners, copiers, and monitors.

User expectation is often based on real–world experience, particularly if strong real–world analogous situations are represented. We would like to augment our status feedback from the visual feedback of the current prototypes to also include auditory feedback, animation, and potentially increased tactile feedback (though not necessarily forced-feedback).

We would also like to prototype devices without displays and determine what kinds of manipulations and status indicators are possible and what types of computational power can be enhanced by such devices.

These all represent interesting research areas for further systematic investigation, such that we obtain a better understanding of physically embedded user interfaces as a new paradigm, its limitations and strengths, and derive design principles to guide others exploring this area.

ACKNOWLEDGMENTS

This work was carried out at Xerox PARC and benefited from input given by the Portable Document Reader group at PARC, DpiX, and FX-PAL. We would also like to acknowledge the PARC 6A program which funded Carlos Mochon as an intern, Tom Moran for comments, ideas, and suggestions, and Rob Burtzlaff, patent attorney extraordinaire, without whom this paper would not have been released for publication.

REFERENCES

1. Card, S. K., Mackinlay, J. D., and Robertson, G. A., A Morphological Analysis of the Design Space of Input Devices. ACM Transactions on Information Systems, (2), April 1991, pp. 99-122.
2. Fishkin, K., Moran, T., and Harrison, B. L. "Design Principles for Manipulative User Interfaces", submitted to *CHI'98*.
3. Fitzmaurice, G. Situated Information Spaces and Spatially Aware Palmtop Computers, *CACM*, Vol. 36, 7, July 1993, pp.38-49.
4. Fitzmaurice, G., Ishii, H., and Buxton, W. A. S. Laying the Foundations for Graspable User Interfaces. *Proceedings of CHI'95*, pp. 422-449.
5. Guiard , Y. and Ferrand, T. Asymmetry in Bimannual Skills, in *Manual Asymmetries in Motor Performance*, Elliot & Roy (eds), CRC Press, Boca Raton FL, 1995, pp. 176-195.
6. Hinckley, K., Pausch, R., Goble, J. and Kassel, N. Passive Real-World Interface Props for Neurosurgical Visualization, *Proceedings of CHI'94*, pp. 452-458.
7. Ishii, H. and Ullmer, B. Tangible Bits: Towards Seamless Interfaces between People, Bits, and Atoms. *Proceedings of CHI'97*, pp. 234-241.
8. Milgram, P. Rastogi, A., and Grodski, J. J. Telerobotic Control Using Augmented Reality. *IEEE Robot and Human Communication* (RO-MAN) 95, Japan, July 1995.
9. Rekimoto, J. Tilting Operations for Small Screen Interfaces. Proceedings of UIST '96. pp.167-168.
10. Schilit B. N., Golovchinsky, G and Price M. Beyond Paper: Supporting Active Reading with free-form digital ink annotations, submitted to *CHI'98*
11. Schneiderman, B. The Future of Interactive Systems and the Emergence of Direct Manipulation. *Behaviour and Information Technology*, Vol. 1 (1982), pp. 237-256.
12. Small, D. and Ishii, H. Design of Spatially Aware Graspable Displays. *Extended Abstracts of CHI'97*. pp. 367-368.
13. Smith, D. C., Irby, C. H., Kimball, R., Verplank, W., and Harslem, E. Deisgning the Star User Interface. *Byte* 7(4), April 1982, pp. 242-282.
14. Taylor, A. R., Nonverbal Communications Systems in Native North America, *Semiotica*, 1975, 13, 4, pp. 329-374.
15. Want, R., Schilit, B. N., Adams, N. I., Gold, R., Petersen, K., Goldberg, D., Ellis, J. R., and Weiser, M. An Overview of the ParcTab Ubiquitous Computing Experiment. *IEEE Personal Communications*, December 1995, pp. 28-43.
16. Weiser, M. The Computer for the 21st Century. *Scientific America*, 1991, 265(3), pp. 94-104.
17. Wellner, P. Mackay, W., and Gold, R. Computer Augmented Environments: Back to the Real World. *CACM*, Vol. 36, No. 7, July 1993.