A Pervasive Keyboard - Separating Input from Display

Carsten Magerkurth, Richard Stenzel
Fraunhofer Institute for Integrated Publication and Information Systems (FhG IPSI)
AMBIENTE - Workspaces of the Future
Dolivostraße 15
D-64293 Darmstadt, Germany
{magerkurth, stenzel}@ipsi.fhg.de

Abstract

A novel text input for public displays and palmtop computers is presented that separates the input from the display of the edited text. While the public display shows both the edited text and a character generation interface, palmtop computers are used for input in a blind way, i.e. without the need to look at the palmtop's small screen. This allows for an effective text input with multiple users interacting simultaneously on the public display. Advanced features such as a dictionary-based word completion can run directly on the public display's computer instead of the palmtop with its limited resources.

1. Introduction

During the past decade of ubiquitous and pervasive computing research there has been a growing amount of scientific activity dedicated to the integration of the different sized and -shaped technologies that are relevant for pervasive computing environments.

Tandler (2001) presents such an environment (i-LAND) and its architecture that has a strong focus on tight collaboration of heterogeneous devices. Within the i-LAND environment, palmtop computers are used for asynchronous and private work on parts of a shared document space (Prante et al. 2002). Palmtop computers and stationary i-LAND devices such as wall-sized displays interact by ad-hoc synchronizing shared data between each other.

With the work presented here we augment this interaction by making use of the characteristics of each coupled device and split the involved tasks between the devices in respect to their specific capabilities.

2. Characteristics of Devices and their Combination

The interaction of tabs (e.g. palmtop computers) and boards (e.g. large/ wall-sized displays) was envisioned to become the real power of ubiquitous computing (Weiser, 1991). Clearly, their characteristics differ enormously.

2.1. Palmtop Computers

The most prominent feature of a palmtop computer is its small size which implies mobility and commonly leads to a 1:1 relation of users and devices, i.e. every user has one personal palmtop device that becomes her personal digital assistant (PDA).

Due to its small size, a palmtop computer suffers from very limited screen real estate which makes it inadequate for many display oriented tasks and even hinder common applications such as web browsing (Levy 2002). Limited memory and computing resources are a downside of its mobility, since power consumption is a crucial issue for any mobile device. On the other hand, most palmtop computers offer an outstanding array of interaction capabilities. There is always a set of hardware buttons and even PC-card-sized PDAs such as the Xircom Rex offer an additional touch sensitive display. Some palmtop computers even come with voice control, although voice recognition features are mostly missing still.

2.2. Large Displays

Large or even wall-sized displays are often found in public places such as an entrance hall of an office building as well as in train stations or airports. Due to its size, a large display is commonly used for presenting public information in contrast to the mostly privately used palmtop computer. Also, a large public display has at least a standard personal computer (PC) behind it that often has a fixed connection to a local network and thus comes with virtually unlimited computing resources.

Although a large display is most adequate for displaying information, interaction capabilities are limited. Due to its public nature there is hardly ever a mouse or a keyboard available. Some displays offer a touch surface (or a similar technology, e.g. Paradiso et al. 2000), but its large dimensions and the vertical surface orientation hinder an ergonomical touch interaction, *especially for text input* (Rekimoto 1998).

2.3. Combination of Palmtop and Large Display

If we combine the palmtop computer's interaction capabilities and its status as a personal device together with the large display's adequateness for displaying information and its computing resources, we gain the best of both worlds and can draw conclusions for using palmtop computers to support the interaction with a public display: The actual input should be performed on palmtop devices, while necessary computational activities should take place on the more powerful public display as well as anything related to perceiving information. Since every user has her own palmtop computer, multiple users should be able to interact simultaneously avoiding limitations of the touch display's hardware that might not support multiple touch positions. Also, user authentication can be realized easily through the PDA and makes explicit logins at the public display obsolete.

3. The MINPUD system

We have developed a text input that attempts to integrate the specific strengths of both devices (MINPUD, Mobile Input for Public Displays). Our setup consists of a SMART board located in a hallway inside our office environment. It has a standard personal computer equipped with two infrared receivers attached to it. The public display is normally used to show information about our research division. This includes a weekly schedule of activities such as meetings, talks, and visits. As a first sample application, this schedule software is built on top of MINPUD and allows members of our division to quickly alter the schedule using their palmtop computers that runs the MINPUD palmtop component.

The typical usage scenario of the sample application is that a division member walks by the public display, perceives the schedule and is immediately empowered to edit it on demand avoiding explicit authentication. This is realised in a fast and ergonomical way without the need to use the rather awkward input methods available for large horizontal displays. When multiple users walk by the display, e.g. after a meeting, they can change the entire group's schedule quickly by working concurrently on different cells (see figure 1).

Figure 1: Using MINPUD

3.1. Public Display Component

The software component running on the public display shows the current week's schedule in a grid-layout similar to personal information management systems. To alter an entry, the user taps on the according cell and then uses his palmtop computer to edit the cell's text by sending characters from the infrared port of his palmtop computer (via a Tiny-TP connection) to the receiver before the public display.

To ensure that the user is authorised to alter the schedule, an authentication code is transmitted prior to sending characters. For each word being entered there is a dictionary-based auto-completion to speed up editing. Since the dictionary is located on a personal computer, there are no memory constraints to take into account and the dictionary can grow infinitely when the user enters new words (in contrast to *T9* or similar dictionary-based systems on mobile devices). In addition to displaying the text being entered and showing suggestions from the dictionary, the public display software also provides visual support for the palmtop component's character generation interface described below (see figure 2).

While one user is editing a cell, another user may tap on a different cell and start editing it in the same way as the first user. If the desired cell is blocked by the floating character generation interface of the first user, a single tap inside the client rectangle moves it to a different area of the screen gaining space to select the desired cell.

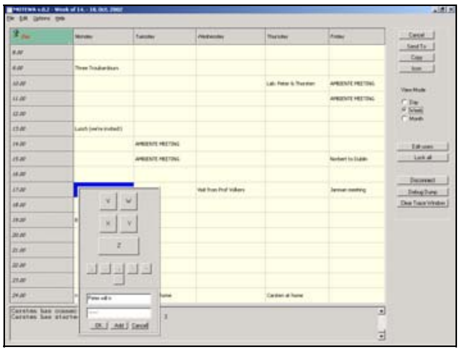

Figure 2: Public display software component

Currently, the number of concurrent users in our setup is limited to two, but this is only because of the infrared sensors being connected with our PC's only two provided serial ports. To deal with this other than getting additional ports, different communication media such as WLAN or Bluetooth are also an option, however, a Tiny-TP infrared connection has the advantage of connecting fast (unlike Bluetooth) and requiring no configuration (unlike WLAN).

Other MINPUD applications than the schedule component can be developed easily, because the MINPUD core system is mainly responsible for translating the palmtop input to according Windows-messages and takes care of sending them to the correct windows on the screen. By using this technique, many legacy applications can be equipped with MinpuD functionality in a very short time, provided that multiple users do not have to be explicitly supported.

3.2. Palmtop Component

The palmtop software component is used for authentication and input of characters. Upon startup, an authentication code (the user's *hotsync* name) is transmitted to the public display over the infrared port. After its acceptance, the user may enter characters with her palmtop using a novel form of interaction that is specifically adapted to the opportunity of perceiving information from the large, public display instead of the palmtop's very limited screen. The palmtop computer is operated in a two-handed way with the thumb of one hand pressing a row of keys beneath the palmtop's display and the thumb of the other hand operating on a set of large virtual buttons directly on the palmtop's touch display (see figure 3). The combination of key-press and virtual button on the device's display creates a character.

The MINPUD palmtop component can be used "blindly", because the virtual buttons are sufficiently large so that it is not necessary to visually supervise the positions of the thumbs as with other display-oriented input methods (see figure 4). The actual interaction design is described below in more detail.

Figure 3: Two-handed interaction

In addition to the public display, a visual feedback can also be given on the palmtop's screen. This is because the palmtop software might even be used as a standalone text input without the public display. Of course, in this case there is no PC-based word completion available. However, during standalone operation, a simple audio feedback can be turned on that creates a beep sound of a different pitch for every generated character. This is to ensure a fairly robust usage when text is entered blindly.

Figure 4: Virtual buttons and hardware keys

The current software implementation runs on Palm PDAs, and could easily be ported to other palmtop computers, provided that they have an infrared port, a touch display, and a row of buttons beneath the display. These demands are fulfilled for most models currently on the market. Since the palmtop component is based on a platform independent application framework for mobile computing (Magerkurth 2001, Magerkurth & Prante 2001) which has been ported to Palm OS and Windows CE, a version for PocketPC devices is trivial.

4. Text Input on Mobile Devices

Designing a text input for mobile devices is a tough problem. The general plight is that the size of a mobile device does not allow for the multitude of different, comfortably layouted keys we are used to with the keyboards on desktop PCs. It is therefore not surprising that no system can match the PC's keyboard speed or robustness and thus mobile text input is in general still awkward and slow (Levy 2002).

4.1. Current Text Input Solutions

While most mobile phones do not come with a touch display, but offer a relatively large set of hardware keys, text input on mobile phones is commonly realized by pressing the hardware keys multiple times (e.g. *Multitap* or *T9*). Since palmtop computers come with a touch-display and fewer keys than mobile phones, common input systems for palmtop computers are display-oriented (e.g. virtual keyboards, or Handwriting recognition). A virtual keyboard builds up a standard QUERTY layout of keys on the screen and thus has the apparent advantage of being very intuitive to use. However, a dilemma exists regarding the screen real estate, because a virtual keyboard should on the one hand have rather large virtual keys that are fairly easy to hit with a stylus, but on the other hand be small enough not to cover too much of the text being edited. In contrast to this, a handwriting recognition software does not necessarily take up any screen real estate, but often comes with certain constraints on the usage side such as limited speed or the demand for special gestures that are similar but not identical to handwritten text (e.g. Goldberg & Richardson 1993). Therefore, there usually is a certain learning penalty with handwriting or gesture recognition.

There have been successful attempts to increase the performance of both virtual keyboards and handwriting recognition systems. For instance, MacKenzie & Zhang (1998) present a new key layout for virtual keyboards that is specifically adapted to the one-finger (or one-stylus) usage of a palmtop computer by re-ordering the ten-finger optimized layout of a QWERTY keyboard. This leads to impressive speed improvements. In addition, the onscreen keys can be enlarged to make them easier to hit at the expense of sacrificing the display of the text being edited (www.palmgear.com). This, however, does not improve robustness as much as one would hope for, because the keys are still hard enough to hit. Also, several software patches exist that make the *Unistrokes*-based (Goldberg & Richardson 1993) *Graffiti* gesture recognition found on Palm PDAs faster or more robust to use (www.palmgear.com).

These and other, partially very unique approaches (e.g. Perlin 1998) still share the common problem of being display-oriented, i.e. they revolve around a very small area of interaction that is not only hard to read from, but also requires a calm hand to use. With MINPUD we leave this limited area of interaction by integrating an additional device to get rid of the display-orientation on the palmtop. However, leaving the palmtop's display does not necessarily lead to a better text input, since there are clearly more issues to regard when developing a mobile text input.

4.2. Text Input Design Requirements

So what is important for the design of a mobile text input? Levy (2002) provides an answer in form of his compilation of "ideal" mobile interface characteristics. His first three characteristics are clearly most desirable and measurable and thus are explicitly followed in the design of MINPUD:

1. "No training to use" (Ease of learning): If an interface is not easy to learn, people will not use it.
2. "Error-Free operation" (Robustness): If people still create errors after having learned the technology, they will get frustrated.
3. Speed. At the end of the day, this is the most important criterion for long-term usage.

4.3. Design realisation of MINPUD

To address the first design requirement of being easy to learn we have applied an anchor concept for associating functionality to the hardware keys and the virtual buttons, which is described in the next section. Additionally, a helper interface is displayed both on the public display and in standalone mode also on the screen of the palmtop computer to show the meaning of the keys before the user has completely internalized them. In contrast to practically all the gesture based input mechanisms there is no need to practise certain motor skills that often make gesture recognition hard to master for beginners.

The second requirement, robustness, is also addressed by providing the helper interface as well as the audio feedback. By using large virtual buttons on the display of the palmtop computer that are operated by a thumb, the display-oriented problems of tapping on relatively small areas of the screen (e.g. accuracy of calibration) are eliminated, too.

To follow the third requirement of a high input speed there are no chains of interaction necessary to generate a character, i.e. no keys have to be pressed several times for one character. Also, no disambiguation methods are used, for they are only slightly faster than triple-tapping (MacKenzie et al. 2001). Every important character can be created with one single action. A dictionary on the public display was added to further speed up the input of longer words.

4.3.1. Anchor Concept. To minimize the amount of training needed to learn the system (design requirement 1), we build on common knowledge the users already have when associating functionality to the hardware keys and virtual buttons.

One can assume that everybody knows the chain of vowels in the alphabet “A-E-I-O-U”. This chain of vowels is associated to the row of hardware keys beneath the display area as shown in figure 4. Accordingly, pressing the first, i.e. leftmost, key creates an ‘A’, while the fifth, i.e. rightmost, key creates a ‘U’, etc. So far, this is obviously very easy to use, but the consonants are still missing.

How are consonants created? If we regard the distribution of the vowels in the alphabet, we can make an astonishing observation: The vowels are very evenly distributed among the consonants. First comes the ‘A’, then follow three consonants (B-C-D), then another vowel (E), then three consonants again (F-G-H) etc. This even distribution is exploited when creating consonants. *Every consonant following a vowel is associated to this vowel*, i.e. ‘B-C-D’ are associated with the ‘A’, ‘F-G-H’ are associated with ‘E’, and so forth. This is in so far easy to comprehend, as the entire alphabet is stored in our minds as a sequential list, so that one can tell immediately that ‘P-Q-R’ follow the ‘O’, whereas it is much harder to tell which characters actually precede the ‘O’. To create a consonant, the user simply holds the corresponding vowel key and simultaneously presses one of the five virtual buttons on the display with the other hand. The first virtual button creates the first consonant after the vowel, the second virtual button creates the second consonant, etc.

Pressing the virtual buttons alone, i.e. without a hardware key that sets the corresponding vowel, has no effect. The interaction of hardware keys and virtual buttons can thus be seen in analogy to using Shift-, Alt-, or Ctrl-keys together with character keys on standard keyboards.

4.3.2. Visual Support. As described above, for each of the five vowel keys pressed, the same virtual button creates a different consonant such as the first consonant following the ‘A’ or the first consonant following the ‘E’. These alternating meanings of the virtual buttons in dependency of the vowel keys are visualized on the public display (and eventually on the display of the palmtop) in real time. Figure 5 (left) shows this support interface on the public display, when no hardware key is pressed and the virtual buttons thus have no effect. Figure 5 (middle) illustrates the updated interface after the user has pressed the first key (A) while figure 5 (right) shows the same for the fifth, rightmost key (U).

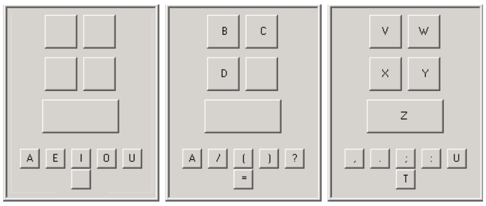

Figure 5: Visual support for creating characters

Please note that the helper interface of the public display includes real buttons, so that the user can actually hold the palmtop in one of her hands pressing the hardware keys. With the other hand, the user can create consonants directly on the surface of the public display, provided that it is touch-sensible. This functionality had been requested by one of our colleagues who felt uncomfortable tapping on the display of his palmtop with anything but the designated stylus.

4.3.3. Special Characters. Text input includes but is definitely not limited to creating vowels and consonants. There are also special characters such as punctuation marks, umlauts, capitalization, and certain functions such as deleting a character, or accepting a suggestion from the dictionary, that must also be taken into account. However, user studies about mobile text input (e.g. Dunlop & Crossan 2000 or James & Reischel 2001) illustrate that for many applications it is possible to go without certain special characters and concentrate more on the actual information to be conveyed. Thus, we chose to optimize the creation of the frequently used standard characters and accept that special characters are harder to create and harder to learn. They are created by pressing multiple hardware keys together which involves moving the hand that operates on the virtual buttons to the bottom of the device.

5. User study

Even though it is immediately plausible that the integration of palmtop computers and public display is potentially beneficial, it must still be proven that MINPUD performs better than existing alternatives. Such a proof could be addressed with mathematical prediction methods. There are indeed several, partially very complex mathematical models for predicting the performance of a text input. However, it has been shown that “neither model [is] particularly accurate” (James & Reischel 2001). Also, the underlying assumption that users are experts who make no mistakes is neither plausible nor does it help measuring the ease of learning (design requirement

1) or the robustness (design requirement 2) of an input method. It is therefore essential to test a text input in a controlled user study.

For a text input that works on an ad-hoc connected pair of public display and palmtop computer it is, however, not fully clear what the control condition(s) should consist of. From the perspective of the research on public displays it stands to reason that other means of entering text on an interactive wall would have to be considered. This could be a virtual keyboard or a handwriting recognition software operating on the public display's surface. But it is problematic to regard other text input methods for large displays, because putting text on a *public* wall clearly involves more than just creating characters. First, user authentication is always an issue, otherwise public walls could simply not offer any interactivity (or would not be public). Means of authentication are manifold, however, and can not be regarded without the context of a specific application (e.g. Russell & Sue 2002). Second, it may or it may not be desired to have multiple concurrent users working with the wall. Virtual keyboards that take away screen real estate might therefore *not* be harmful, when there is just one person at the wall. Third, depending again on the context, it might be useful to create awareness about what the person interacting with the wall is exactly doing. In this case, it is undesirable to hide private parts of the user's interface. It could, of course, just be the other way round. Therefore, it is inevitable to have several confounding factors when trying to find an answer on how *good* a system compared to others might be. MINPUD was designed to enable multiple concurrent users to get quick access to textual information on a public display which naturally includes an implicit authentication mechanism. We are not aware of a text input for interactive walls with a similar focus.

The other applicable perspective would be the research on mobile text input. From this perspective one would take into account traditional means of entering text on mobile devices such as Graffiti or a small virtual keyboard. The advantage of this perspective is that there are fewer confounding factors, because e.g. one device is always operated by one user and not by multiple users and the possibility of other persons perceiving the interface of the user operating the palmtop computer is not an issue, too.

When choosing such a mobile text input perspective, the research question can be formulated as in how far the coupling of the small mobile device with a large stationary one and the related distribution of input and display allows for a better text input than with the display-oriented input methods possible on the palmtop device alone.

5.1. Design

The experimental design of the user study was taken from the carefully crafted study on text input performance for mobile phones presented by James & Reischel (2001). Details can be found there. One benefit of picking up their design was the gained ability to provide a comparison with text input on mobile phones *for free*, although, of course, for various reasons one cannot derive much more than careful tendencies from such a comparison.

5.2. Method

5.2.1. Subjects. Thirty students recruited from the Technical University of Darmstadt, Germany, (m = 24, w = 6, ∅ 28.4 years) took part in the study. None of them had previous usage experiences with palmtop computers. Subjects were randomly assigned to three different groups. One text input was used within each group (MINPUD, Graffiti handwriting recognition, and the Palm Virtual keyboard). Half of the participants within each group were assigned to being *experts*, i.e. they were encouraged to train their text input intensively before the study commenced, whereas the other half were *novices* who only received a demonstration and a short time span of twenty minutes to get acquainted with the text input.

5.2.2. Setting and Procedure. Participants were seated at a table with a Palm computer and a standard PC monitor (for the MINPUD group) before them. Ten sentences were presented consecutively to each participant who had to enter them into her device as fast and as accurate as possible without correcting herself. The sentences were taken from the study of James & Reischel and used without being translated to the native (German) language of the participants in order to preserve the correct number of characters in each sentence. This way, a comparison with the 2001 study could be realized. Since all of the subjects were either good or very good at speaking English, we believe that the internal validity of the study did not suffer significantly from this fact.

Half of the sentences presented to the subjects were short, informal sentences that are typical for SMS or "chat" communication using an Instant Messenger (e.g. "let me know if we should wait" or "hi joe how are you want to meet tonight"). The other half consisted of longer, more complex sentences typically found in a book or a newspaper such as "the slinky and the jump rope are among the five toys that will be designated wednesday". Punctuation marks or capitalization were not used.

5.3. Results

For the analysis, a 3 (input method) x 2 (text type) x 2 (user experience level) design was used. The mean words per minute (wpm) are listed in table 1 and the aggregated total errors for each group are listed in table 2. To provide a comparison, the results for mobile phones from the James & Reischel study are also listed in the columns 4 and 5.

Significant main effects of the analyses of variance were found for the user experience level with the wpm as the dependent variable ($F(2, 151) = 11.7$, $p < 0.5$) and for the input method with the total errors as the dependent variable ($F(2, 151) = 10.8$, $p < 0.5$). A significant interaction effect was found for input method x user experience level with the wpm as the dependent variable ($F(1, 151) = 12.3$, $p < 0.5$).

Table 1: Mean text input speed (wpm)

Text	Users	[1]	[2]	[3]		[4]	[5]
Chat	Novice	14,62	19,93	4,6		10,98	10,37
	Expert	24,5	31,43	15,42		25,68	10,53
News	Novice	12,58	14,15	4,33		7,21	5,59
	Expert	21,27	26,61	12,64		15,05	5,33

[1] MINPUD, [2] Virtual Keyboard, [3] Graffiti, [4] T9, [5] MultiTap

Table 2: Total errors

Text	Users	[1]	[2]	[3]		[4]	[5]
Chat	Novice	24	25	99		24	17
	Expert	21	24	53		17	31
News	Novice	26	35	147		20	48
	Expert	20	29	88		17	85

[1] MINPUD, [2] Virtual Keyboard, [3] Graffiti, [4] T9, [5] MultiTap

5.4. Discussion

MINPUD's speed of input does not quite reach that of the virtual keyboard on a descriptive level. However, it is clearly a lot faster than Graffiti and also than the input methods on mobile phones. This supports the design requirement of high input speed.

The small differences in total errors for novices and experts support the design requirement of being easy to learn. Also, the small absolute number of total errors both for experts and novices supports the design requirement of being robust to use.

When comparing the error rates and the input speed of MINPUD and the other input methods for palmtop computers with those for mobile phones, the mobile phones input methods appear to be rather robust, too, but only partially at a similar level of speed. It must be kept in mind, however, that the experimental situation was a different one in the James & Reischel study. Especially psychological variables such as the response bias, i.e. the individual bias to sacrifice speed for accuracy or accuracy for speed, could not be controlled. Therefore, interpretations have to be careful here. Regarding the research question, if it is beneficial to leave the display-oriented approach of traditional text input methods for palmtop computers in favour of distributing the actual input and the visual feedback between the palmtop and an additional display, this clearly is a promising approach given the results of the user study.

6. Related work

Several research groups have worked on the interaction of palmtop computers and stationary devices such as interactive walls or personal computers with the goal of manipulating information on the stationary devices with the palmtop computers.

Myers et al. (1998) present *Pebbles*, a system for connecting multiple Palm PDAs to a personal computer. The PDAs are used as remote controls that let several users interact with the PC via the PDAs. The *PebblesDraw* application enables multiple users to draw an image simultaneously with the palmtop computers. This application is especially elegant, because shared image drawing involves many critical issues such as modeless interfaces or adapted undoing functionality.

Greenberg et al. (1999) are dealing with a large display that runs a single display groupware (SDG) system and palmtop computers for taking personal notes that can be "publicized" to the large display. This *SharedNotes* setup focuses on important issues regarding private and public information and the transition between them. Even though Greenberg et al. are not specifically addressing the details of PDA to wall text input, their work raises several highly relevant questions, such as the need for device adapted software functionality.

Very close to MINPUD in terms of multiple device integration is Rekimoto's (1998) painter's palette approach. He presents a setup where users at an interactive wall are equipped with a palmtop computer held in one hand just like the palette of an oil painter. By using a new 'Pick-and-Drop' metaphor a user can select objects on one of the devices and simply drops them at an other device by tapping on the other device's surface, where they can then be edited. His system works with several different kinds of data objects and activities such as drawing images or placing cliparts.

7. Conclusions

We have presented MINPUD, a novel text input for multiple concurrent users that works on a public display and palmtop computers. It separates the actual input from the display of the edited text and the character generation interface. Thus it realizes an interaction design that leaves the standard display-oriented input methods commonly used on palmtop computers. The design of MINPUD was optimized for ease of learning, robustness, and speed which was also shown in a user study.

The current sample setup includes an application to alter a group's schedule shown at the public display. Its grid-like user interface makes the realization of multi-user support rather easy. Our experiences are that walking by the display and editing its content works quite simply and in certain situations is a better alternative than going back to a PC workstation to edit the group calendar there. However, the full potential of concurrent usage has not yet been exploited. In the future, we will investigate the possibilities of a tighter integration between multiple users like with a shared text editor.

8. Acknowledgements

The authors would like to thank Norbert A. Streitz, Thorsten Prante and Sascha Nau for their helpful suggestions to improve this paper.

This work was funded by the Ladenburger Kolleg "Living in a smart environment" of the Daimler-Benz foundation.

9. References

Dunlop, M. D., Crossan, A. (2000). Dictionary based text entry method for mobile phones. In: Second Workshop on Human-Computer Interaction with mobile devices, 5-7.

Goldberg, D., Richardson, C. (1993). Touch-typing with a stylus. Proceedings of CHI 1993, 80-87.

Greenberg, S., Boyle, M. & Laberge, J. (1999). PDAs and Shared Public Displays: Making Personal Information Public, and Public Information Personal. In: Personal Technologies, Vol.3, No.1, 54-64.

James, C.L., Reischel, K.M. (2001). Text input for mobile device: Comparing model prediction to actual performance. In: Proceedings of CHI 2001, 365-371.

Levy, David (2002). The Fastap Keypad and Pervasive Computing. In: Proceedings of Pervasive 2002.

MacKenzie, I.S., Kober, H., Smith, D., Jones, T., Skepner, E. (2001). Letterwise: Prefix-based Disambiguation for Mobile Text Input. In: Proceedings of the ACM symposium on User Interface Software and Technology, 111-120.

MacKenzie, I.S., Zhang, S.X. (1999). The Design and Evaluation of a High-Performance Soft Keyboard. In: Proceedings of CHI 1999, 15-20.

Magerkurth, C. (2001). Programming Windows to Create Palm Games. In: Proceedings of the Game Developer's Conference 2001, 439-453.

Magerkurth, C. (2002). Entwicklung und Evaluation eines alternativen Texteingabesystems für Persönliche Digitale Assistenten. Proc. of Mensch&Computer 2002, 205-214.

Magerkurth, C., Prante, T. (2001). Towards a unifying approach to mobile computing. SIGGROUP Bulletin, April 2001, 22, 16-18.

Myers, B. A., Stiehl, H. & Gargiulo R. (1998). Collaboration Using Multiple PDAs Connected to a PC. In: Proceedings of CSCW 1998, 285-294.

Paradiso, J.A., Hsiao, K., Strickon, J., Lifton, J., Adler, A. (2000). Sensor Systems for Interactive Surfaces. IBM Systems Journal, Vol. 39, No. 3&4, pp. 892-914.

Perlin, K. (1998). Quikwriting: Continuous Stylus-based Text Entry. In: Proceedings of UIST 1998, 215-216.

Prante, T., Magerkurth, C., Streitz, N. (2002). Developing CSCW Tools for Creativity - Empirical Findings and Implications for Design. In: Proc. of CSCW 2002, 106-115.

Russell, D. M., Sue, A.(2002). Using Large Public Interactive Displays for Collaboration.Position paper at UbiComp 2002 Workshop on Collaboration with Interactive Walls and Tables. http://ipsi.fhg.de/ambiente/collabtablewallws/

Rekimoto, J. (1998). A Multiple Device Approach for Supporting Whiteboard-based Interactions. In: Proceedings of CHI 1998, 344-351.

Silfverberg, M., MacKenzie, I.S., Korhonen, P. (2000). Predicting text entry speed on mobile phones. In: Proceedings of CHI 2000, 9-16.

Tandler, P. (2001). Software Infrastructure for Ubiquitous Computing Environments. In: Proceedings of Ubicomp 2001, 96-115.