User Interface Overloading: A Novel Approach for Handheld Device Text Input

James Allan Hudson, Alan Dix & Alan Parkes

Computing Department, Lancaster University, Bailrigg, Lancaster LA1 4YR, UK

Tel: *+44 1524 592326*

Email: *j.a.hudson@lancs.ac.uk, {app, dixa}@comp.lancs.ac.uk*

Text input with a PDA is not as easy as it should be, especially when compared to a desktop set up with a standard keyboard. The abundance of attempted solutions to the text input problem for mobile devices provides evidence of the difficulties, and suggests the need for more imaginative approaches. We propose a novel gesture driven layer interaction model using animated transparent overlays, which integrates agreeably with common windowing models.

Keywords: text input, PDA, transparent layers, visual overloading.

1 Introduction

The major difficulty with designing graphical interfaces for small touch screen displays is a regular text document has to be divided into very small pages, making comprehension awkward. An additional problem is control elements take up precious display area, making the view of a document ever smaller. One approach is to reduce the size or number of these controls, to free up usable display area, however this affects the usability of an interface. The problem is in maintaining a reasonable sized interface without affecting usability.

This paper considers these problems associated with handheld text input using touch screen graphical interfaces. It proposes the application of superimposed animated graphical layering, which we refer to as *visual overloading*, combined with gestural interaction as illustrated by Belge et al. [1993], Lokuge & Suguru [1995], Meyer [1995] and Silvers [1995] to produce a novel interaction model called User Interface Overloading or UIO [Hudson & Parkes 2003b]. We argue that this approach can help to address the problems of touch screen text input, especially for devices with limited display real estate.

The difficulty in constructing good solutions to interaction for handheld and portable devices with small graphical display has spawned much interest from researchers specializing in multi modal and tangible forms of interaction, however UIO suggests much more can be made of these small graphical displays. This paper examines some of the popular approaches to text input, some of these being currently under development. To set the benefits of the UIO model in suitable perspective, the paper then goes on to identify and discuss the individual features and difficulties of the PDA text input problem as demonstrated by Kamba et al. [1996], Masui [1998], MacKenzie et al. [1999] and MacKenzie & Zhang [1999]. We then introduce our model. Finally, we examine an implementation of a UIO text input application.

2 Handheld Text Input

Many proposed solutions to the handheld text input problem fail to acknowledge the true obstacles of preserving portability and compactness, ease and convenience of interaction and the deft conservation of screen real estate. Before these factors are addressed and in order to illustrate the problem of text input for handheld devices, this section outlines some of the more successful approaches, this section critically examines a number of text input solutions.

Plug-in keyboards or the very appealing laser projected variety, such as iBiz virtual laser keyboard would seem to offer a solution to the problem of easily entering text on small devices. However, this could likened to buying an anchor to make your PDA behave like a desktop. The integration of a full size keyboard into the design compromises the necessary limit on size and ergonomics of use, not to mention the portability of the device, by requiring a flat surface.

A different approach is the chorded keyboard, more usefully implemented for handhelds as a device held in the hand. Here there is a significant learning overhead due to the user having to learn key combinations to select each letter or number, however this approach does outstrip all one handed text input rates at 50wpm. A downside to this approach is with current implementations the need to hold a chorded keyboard in one hand, does affect the ergonomics of interaction. The obvious solution would be to integrate the keyboard into the device itself. Similar to the chorded keyboard is the T9 predictive text found on many mobile phones. Entering a series of keys will generate a list of possible words. This approach does however pose difficulties, if the word is not found in the dictionary or the suggested word is at the bottom of the list of suggestions.

Clip on keyboards may seem to provide a usable text entry facility for small devices, at least on physical grounds. However, they do add bulk, and thus adversely affect the trade-off between size, portability and practicality. An alternative to the clip on is the *overlay keyboard*. Though these do not increase the size of the device, they do have usability implications. The overlay is essentially no different from a soft keyboard (discussed below), and actually is a very expensive sticker that permanently renders the utility of a portion of the display for text input only, restricting the use of an already limited resource.

The soft keyboard is not really too different from the clip-on keyboard, except it is implemented as a graphical panel of buttons rather than a physical sticker. The

soft keyboard has the added hindrance of consuming screen display area, as does the overlay approach. However, the soft keyboard does permit the user to free-up display area when required.

The soft keyboard seems to be the most commonly accepted solution [see Kamba et al. 1996; Kölsch & Turk 2002; MacKenzie et al. 1999; MacKenzie & Zhang 1999]. However, it is a solution that is greedy in terms of screen area. Two examples can be used to illustrate the trade off between redundancy, ergonomics of use and visible display. Firstly, a full screen keyboard offers direct manual interaction due to larger keys and a capacity for more keys but at the expense of display real estate. Secondly, the standard split screen keyboard already limited in size, sacrifices redundant controls to permit larger keys and to make more visible display available, yet its small size results in the need to use an additional device, such as a stylus, which results in an approach that is difficult to use dexterously with the fingers.

One approach based on the standard keyboard and akin to one we propose is one that uses a static soft keyboard placed in the background of the display text. A letter is selected by tapping the appropriate region in the background. This solution permits manual input and does preserve some screen real estate. However, the number of available controls and hence redundancy is limited due to the necessary larger size of the controls, required to make the keys legible through the inputted text. This limit on the number of controls necessitates an awkward need to explicitly switch modes for numbers, punctuation and other lesser used keys. Another drawback is the slight overhead in becoming accustomed to the novel layout.

A lot of effort has been expended to improve the soft keyboard approach, however these attempts are still subject to the drawbacks already describe with this approach, moreover they are subject to a learning overhead imposed by remodelling the keyboard layout. On the Unistroke keyboard [Zhai et al. 2000; Mankoff & Abowd 1998] all letters are equidistant, thus eliminating excessive key homing distances. The Metropolis keyboard [Zhai et al. 2000] is another optimised soft keyboard layout, statistically optimised for single finger input, improving efficiency by placing frequently used keys near the centre of the keyboard. Both approaches can be effective, but both impose a learning overhead due to a new keyboard layout. The user must expend considerable effort to become familiar with the keyboard for relatively slim rewards, not to mention the overhead inherent with soft keyboards, such as the consumption of screen real estate.

Handwriting recognition was for some time the focus of PDA text input solutions. However, evaluation revealed that gesture recognition for text input is balky and slower, some 25wpm at best, than that of, say, other less sophisticated approaches, such as the soft keyboard [Dix et al. 1998, p.6]. Problems with handwriting and similar approaches such as 2D gesture interaction, for example Graffiti, is one of learnability, slow interaction and skill acquisition. The obvious problem with handwritten input is the need and time expended to write each letter of a word, whether this is consecutively or all at once, the user must still write the whole thing out, whereas the keyboard solution requires merely the pressing of a button. A problem originally addressed with the invention of the mechanical typewriter. In

addition to this difficulty, as with the standard soft keyboard, text input requires the use of a stylus, thus occupying the user's free hand (i.e. the need to hold the PDA) when entering text. The learning curve of this approach is steep due to the need to learn an alphabet of gestures and the saving in real estate is not so apparent, since some approaches require a large input panel.

We now consider alternative, less well known, solutions to the problems of text entry for small devices. One approach to PDA text input is the use of a mitten outlined by Goldstein & Chincholle [1999]. Sensors in the hand units measure the finger movements, while a smart system determine appropriate keystrokes.

This novel approach is an intriguing solution. The main problem is the need to carry around a mitten that is nearly as big as the device itself. Finally, a mitten is not so appealing to the user and the sensors on these devices can be bulky affecting freedom of movement. Dynamic dialogues illustrated by Ward et al. [2000], when applied to limited display size, are a very innovative data entry interface which incorporates language modelling. The animations are driven by continuous two-dimensional gestures, where the user selects strings of letters as they progress across the screen. Letters with a higher probability of being found in a word are positioned close to the centre line. Though the dynamic dialogue approach makes use of 2D gestures, these are supported by affordance mechanisms and they have been kept simple for standard interaction, making them readily learnable. Users achieve input rates of between 20–34 words per minute, which is acceptable when compared with typical one-finger keyboard touch screen typing of 20–30 words per minute [Sears et al. 1993]. However, the input panel for text entry consumes around 65% of the display, leaving as little as 15% remaining for the text field. The approach does not improve on the constraints of limited display area or on text input rates. What it does do is require the user to become familiar with a new technique for no extra benefit.

3 Evaluation of Handheld Text Input

The major problem with many text input solutions is the lack of investigation into the true problem of handheld device text input. The important thing is not the *mechanism* for inputting text in itself but rather the consideration of constraints such as on the available size of a text input panel and free display area.

We next discuss the constraints on the design of text input interfaces for handheld devices, in order to derive several requirements and to set the introduction and discussion of the UIO model in a suitable perspective.

4 Layout Constraints and Ease of Use

The layout of a text input mechanism is subject to some physical constraints which affect usability. In order to free up as much screen display as possible, input dialogues are reduced in size, which reduces the size of individual keys, making them more difficult to select. Increasing the number or redundancy of controls limits the space available. The size of keys is also subject to the population of keys on the keyboard. Lots of keys means less space per key, or a smaller input text panel. Alternatively, to minimise the display area used by the keyboard and maintain a reasonable sized key, designers resort to using menus or modes. Seldom

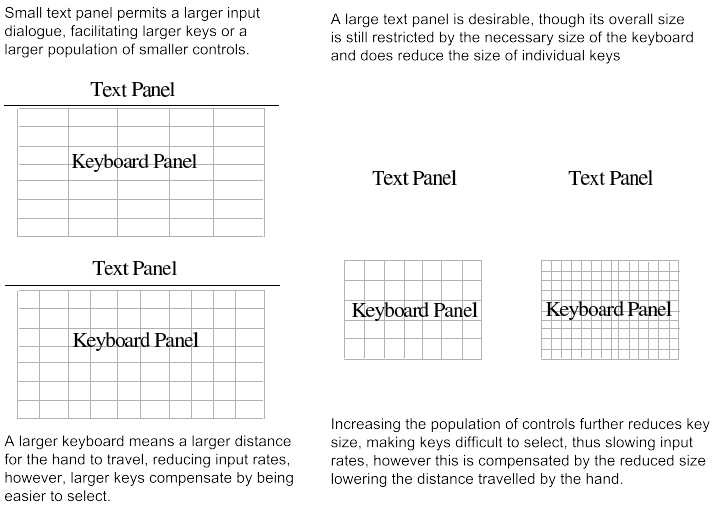

Figure 1: The possible combinations of text input panel, the constraints between input rates, size of keys, number or keys with the remaining available display and the size of text input dialogue and display.

used commands inevitably feature in sub-menus, which leads to a slow and awkward interaction approach [Kamba et al. 1996]. These constraints are subject to the constraints defined in Fitts' law, a large dialogue is subject to a time overhead from increased hand travel while smaller keys take up less space and merit a reduced hand travel, yet may incur a time overhead due to a fine motor control requirement in selecting a key. Overly small keys result in either unacceptable increases in error rates or unreasonably slow input rates for text input, due to awkwardness of selecting a key accurately. This suggests a larger keyboard should be favoured.

4.1 Unnecessary Interaction Aids

Pointers, such as a stylus, clip on keyboards and data gloves, impede device usability. To interact with the device the user must either don the interaction accessory or, say, pick up a stylus, which in the case of many portable devices, ties up both hands [Goldstein & Chincholle 1999].

4.2 Learning and Skill Acquisition Overhead

Many small device text input approaches are not easily learned, consider the work by MacKenzie et al. [1999; MacKenzie & Zhang 1999]. The use of 2D alphanumeric gestures is a good example of such an approach. Here the user expends time to learn numerous gestures and the different contexts they can be used in.

5 Design Requirements

Drawing from the evaluation of text input solutions a definition of the design requirements can be constructed, permitting the development of a fresh and fitting

solution, rather than, further optimising on approaches that fail to address relevant issues such as screen real estate or convenience of use, for example the over engineered optimisations of the conventional soft keyboards

Consideration of the contributing factors in the design of interaction models for handheld and mobile devices leads to the following design requirements:

- Larger keys for manual interaction should be favoured over interaction aids. For example styluses, obstruct the freedom of a hand, posing a hindrance to handheld interaction.
- We must seek a good balance between redundancy in the number of visible input device features and availability of display area.
- The device must reflect an effective trade-off between display area, size of elements in the input panel, and usability.
- The approach must be easy to learn to use and understand or there must be a justifiable benefit for any learning overhead, as with the chorded approach.

In view of the above requirements, we now discuss our proposed approach to the problem of text input for small devices.

6 User Interface Overloading

Here we introduce a novel system of interaction called user interface overloading, whereby a user can selectively interact with multiplexed or Visually Overloaded layers of transparent controls with the use of 2D gestures.

Transparency is commonly used to optimize screen area, which can often be consumed by menu or status dialogues. The aim is to provide more visual clues [Bier et al. 1994], in the hope the user will be less likely to lose focus of their current activity. Bartlett [1992] and Harrison et al. [1995] consider that the conventional approach of using a layer of transparency to display a menu is done at the cost of obscuring whatever is in the background (Figure 2 right). This is not actually visual overloading, but rather a compromise between two images competing for limited display area. In fact, an underpinning feature of the scheme described by Harrison et al. [1995; Harrison & Vicente 1996] is the investigation of levels of transparency to optimize this compromise.

Visual overloading is different from the use of static layered transparencies. Rendering a transparent animated image or a wiggling panel on a static background, illustrated by Belge et al. [1993], Silvers [1995] and Cox et al. [1998], will visually multiplex or visually overload the overlapping images (see Figure 3). The upshot is a layer of controls appear to float over the interface without interfering with the legibility of the background.

The introduction of gestural input [Meyer 1995] is partly a consequence of implementing visual overloading, since it is necessary to resolve the issue of layer interaction. There is nothing new with gesture activated controls, the concept was first introduced by Kurtenbach & Buxton [1994] with marking menus. However, this approach did only use simple gradient stokes or marks, whereas UIO also makes

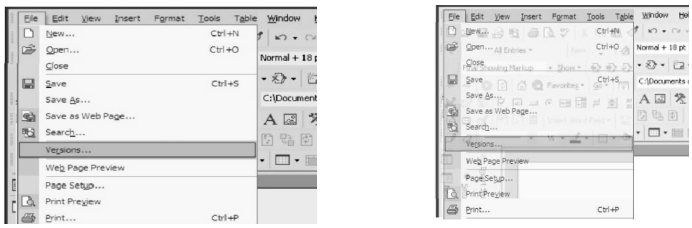

Figure 2: The benefi ts of transparent over conventional solid menus. (Left) the solid menu conceals the background image. (Right) a transparent menu obscures the background image, without completely concealing it.

Figure 3: Three frames from a UIO mobile phone interface with overloaded icons, showing where gestures are executed. The envelope icon is for the messaging function and 'Register' for the call register function. For example a 'C' starting over the envelope will go to a compose dialogue. Please note visual overloading is diffi cult to present in print.

use of more sophisticated gestures. The underlying principle of marking menus is to facilitate novice users with menus while offering experts a short cut of remembering and drawing the appropriate mark without waiting for the menu to appear. What makes our UIO interaction model novel and where it differs significantly is the use of selective layer interaction. We now discuss some of the features and properties of UIO.

The approach incorporates 2D mouse gestures to activate commands associated with a control (see Figure 3), offering the necessary additional context required beyond that of the restricted point and click approach. This enables the user to benefit from the added properties associated with an overloaded control by enabling the selective activation of a specific function related to a control contained in the layers.

UIO permits the intensive population of a display through the layering of control elements. This we achieve without compromise in size of the inputted text panel or to the size of control elements described by Hudson & Parkes [2003a]. An advantage that effectively gets round the constraints described earlier, (see Figure 1) by permitting background and subsequent layers to occupy the same screen real estate.

Another benefit is the availability of real estate permitting larger controls, which are easier to locate, improving input rates and facilitate manual interaction.

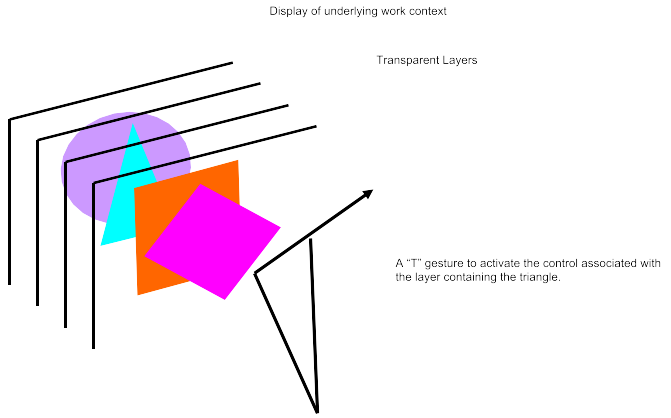

Figure 4: A schematic depiction of an overloaded button or icon. By executing an appropriate gesture, over the collection of layered shapes, such as a 'D' for the diamond or a 'T' for the triangle, etc., a call can be made to the action associated with the desired layer.

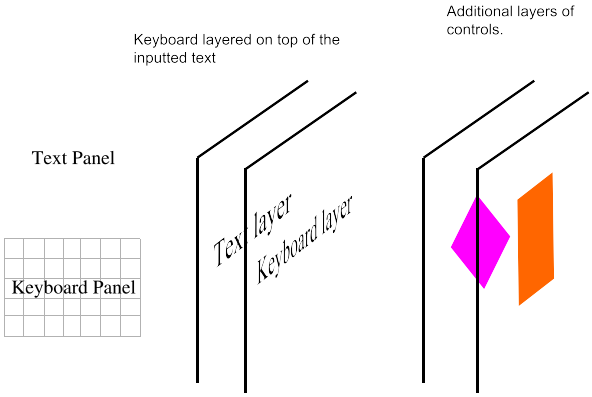

Figure 5: The benefi ts of layering a keyboard over the text panel, essentially doubling their size and breaking the conventional physical constraints (see Figure 1) associated with user interface design. It is also clear additional layers of controls can be added.

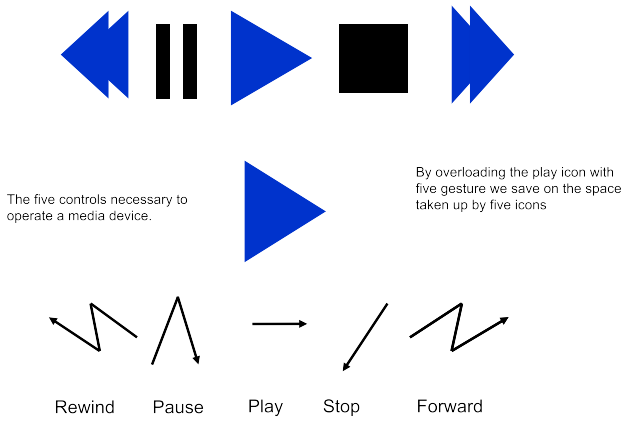

Figure 6: How real estate can be salvaged, by associating gestures with an icon rather than using ever more control elements.

The constraints of this approach are that elements lose coherence gradually or the interface essentially becomes visually noisy as layers are added, however carefully chosen layers permit a good number of controls before this constraint takes effect.

It is clear UIO eliminates the constraints between the size of the display and the input dialogue. In addition the redundancy of controls can be increased in a new way, by overloading the functionality of a control with a selection of gestures (see Figure 6) while avoiding the use of obtrusive context menus.

Expressed in a different way, we can limit population of these controls by overloading their functionality with gesture interaction offering significant savings in screen real estate for handheld devices and other touch screen interfaces.

A problem of gesture interaction is the steep learning curve, because of the need to be familiar with a multitude of gestures and their contexts. An addition to the UIO approach, to support learnability is to introduce a mechanism where an easily remembered '?' gesture will prompt the interface to display the gestures associated with a control or area. In this way the user can become familiar with the system gradually, summoning help in context and when needed. This *blossom help* (described in Figure 7) also functions as a mechanism to support goal navigation and exploration. This help approach is an elaboration on the marking menu reported by Kurtenbach & Buxton [1994]. To improve the usability a function can be activated using the correct gesture or using a text label as a buttons. In addition there is no reason why this help system could not include permitting a straight-line mark from the icon to the label, as with a marking menu, however this will only work with less than eight options and would fail to be useful with layers of controls.

Essentially, a UIO control is a making menu with a transparent graphical image, which means UIO benefits from the properties of marking menus. As with marking menu, UIO requires a procedural memory component, suggesting this style of interaction has a strong cognitive salience.

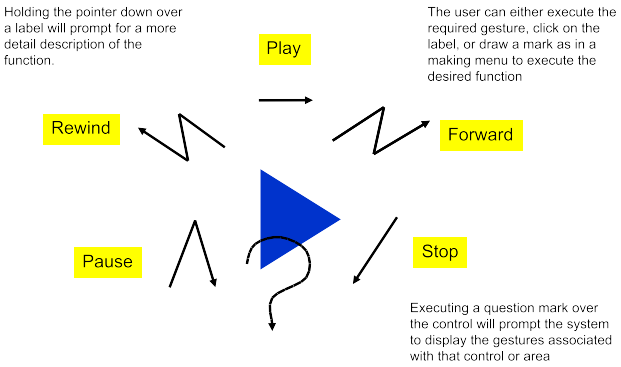

Figure 7: How learnability is supported with the use of help dialogues that 'blossom' when a '?' gesture is executed over a control or area.

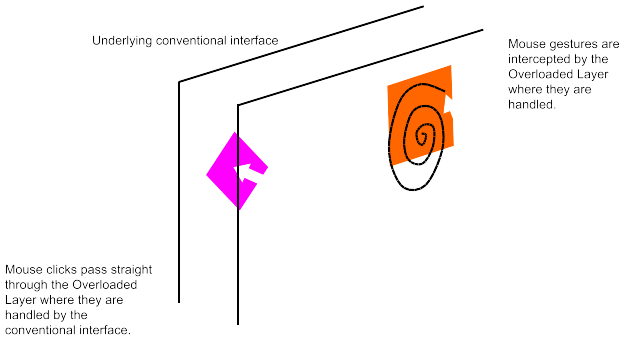

Figure 8: How this approach can be seamlessly incorporated into a conventional point and click interface. Mouse clicks are not intercepted by the Overloaded Layer and pass straight through, where they are handled by the conventional interface, whereas gestures are handled by the Overloaded layer.

A benefit of UIO is that it integrates seamlessly with WIMPS offering extended functionality by intercepting gestures but allowing standard point and click interaction to pass through the layers where they are handled in a conventional way (see Figure 8). An obvious comment is user interface overloading may interfere with drawing packages and text selection. The solution to this is the same used by Sensiva's Symbol Commander, conflicts are avoided with a small time delay to switch modes or simply using the right mouse key to activate gesture input.

There are some downsides to the UIO approach. As with keyboard shortcuts, by letter association, ambiguity can lead to controls possessing the same gesture. This can be overcome with good design either by planning letter associations or incorporating an options dialogue for selecting between commands with the same gesture.

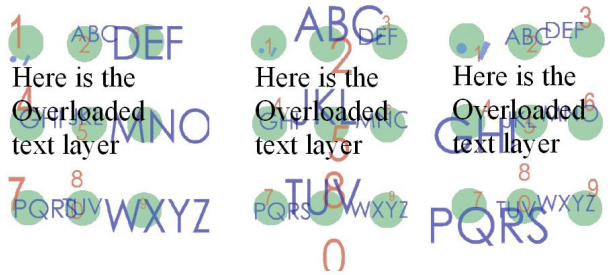

Figure 9: Three frames from a screen shot of the VODKA approach, a mobile keyboard layer over a layer of text. Although, it cannot be seen in print, the layer of letters, when in motion, stand out against the background and appears a lot more coherent.

In practice we found that multiple layers of animated transparent elements were too visually noisy, suggesting only one layer for this type of icon. We did find wiggling or moving non-animated panels [Belge et al. 1993; Lokuge & Suguru 1995] can support in practice up to four layers. We did find that overloaded transparencies work with very low levels of transparencies, lower than the 30% opacity for static images suggested by Harrison & Vicente [1996].

Other restrictions exist that can be avoided with good design are, the choice of colours conflicting with the background, and in the poor choice of animations which may result in difficulties selecting moving elements or distinguishing between layers. However, this is no more an overhead than in designing graphics for a standard interface or website. Another restriction is animated controls can be obscured on a moving background, such as a media clip.

We now examine an implementation of a user interface overloading text input application.

7 Proposed Solution

The user interface overloading technique was implemented in a handheld text input application and evaluated with respect to the specified requirements as follows. Our proposed approach to text entry on small devices the visually overloaded 2D gesture keyboard Application or VODKA utilises the UIO technique. To test the approach a prototype was implemented on an Ipaq 3600 using the Java virtual machine. The graphics were produced using a drawing application. The gesture engine was an optimised version of Javastroke, ported from the open source gesture recognition engine Libstroke.

The implementation of VODKA incorporated a visually overloaded ISO keyboard layout (standard on mobile phones) and a number pad layered over the text. Gestures were incorporated using simple gradient strokes to select a letter and simple meaningful gestures to access other functions, such as numbers and uppercase letters.

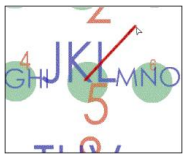

Figure 10: In this example a letter is selected by drawing a gradient stroke that begins over the button. The green dots indicate where the gesture must start.

To operate the keyboard (see Figure 10) the user makes very simple gradient gestures, as described by Kurtenbach & Buxton [1994]. To select a letter, a gradient stroke that starts over the selected button is performed. The centre point of a button is indicated with the green dot. The angle of a gesture supplies the context indicating which element is being selected. 'L' would be selected with a right terminating gesture, as above, while 'K' would be selected with a vertical up or downward stroke. To improve usability the 'space' character is easily selected with a 'right-dash' gesture, that can be executed anywhere on the display, similarly a delete command is selected with a global 'left-dash'.

To access lesser used functions other than basic text input, the approach uses more elaborate gestures such as selecting the number '5' with a meaningful and easily associated 'n' gesture.

Other options are, text can be cleared from the screen with a 'C' gesture and a capital can be entered by drawing a 'U' for uppercase after the desired letter. The need to learn these associations does pose a learning overhead, however they are easily learned using the Blossom mechanism (Figure 7). Initially, this use of symbols is no less awkward than selecting a mode or menu option, however as the operation becomes familiar, it ceases to be as obtrusive as the other approaches. Point and click interaction is left alone to demonstrate that the approach could incorporate the T9 approach and could still use standard text interaction, such as with text editing in conventional graphical interfaces.

8 Evaluation

The UIO approach leads to several benefits. The layering of controls increased the size and available population of controls while permitting the largest of text panel. This permits manual interaction obviating the need for a stylus improving the usability of the device. Gestures further reduced the outlay of screen real estate that would be necessary to provide a control for each function. The marking menu style of interaction for regular text input made VODKA simple to use and easily understood. The gradient gestures found in marking menu interaction for simple text input are trivial to parse, with error rates no worse than point and click approaches [Kurtenbach & Buxton 1994]. The use of gestures for more elaborate interaction made it simple to access modes such as capital letters and numbers. The use of Blossom help (Figure 7) offers a good solution to the problem of remembering the associations for more sophisticated gestures, by providing a mechanism that can

be configured to passively reinforce these associations. Another example of the usefulness of the blossom approach is in acquiring the skill for the execution of a gesture, by assisting in learning the correct form for that gesture.

Viewed against our set of requirements VODKA does appear a suitable solution:

- There is no reliance on additional interaction aids, since the dialogue elements are large enough to support manual operation.
- The approach reflects an effective combination of redundancy in input device features and availability of display area.
- The approach provides an adequate trade-off between display area, size of elements in the input panel, and usability, with an approach that circumvents these constraints.
- Finally, the approach is easily understood and learned with a simplistic interaction style. Moreover, any learning overhead in learning symbols is arguably justified when weighed against the benefits.

As has been illustrated, our user interface overloading technique resolves the text entry problem and goes a considerable way to satisfying the design requirements.

VODKA is not clearly a gesture input approach, especially for regular input, since gradient gestures are no more difficult to learn and execute than pointing. So, the UIO approach falls under neither point and click or gesture input. Therefore, we decided to not compare it to gesture based input, since VODKA is more like a keyboard than anything else. We decided to test against a qwerty layout, in the hope this would offer a clear indication of the value of our approach when compared with more successful and conventional approaches. We decided not to conduct a longitudinal, (clearly a longer study would be the next step) since we wanted to demonstrate the input rates achievable by a novice. We have avoided predictive text, since it can be introduced to all forms of text input, including our model; our interest is in raw input rates.

To test out approach fifteen subjects were used in a study, all with experience in using the common mobile ISO alphabet keyboard and the qwerty layout, with 30 minutes experience with the VODKA text input approach. We tested the input rates with VODKA and compared them against input rates on the same Ipaq device with a qwerty layout. The average input rate for the qwerty was 27wpm. The average rate for VODKA was 21wpm, with experienced users achieving an acceptable 28wpm.

The results were analysed using an ANOVA of the logarithm of the time spent. Logarithmic analysis was chosen as the data is positive and skewed and most effects were expected to be multiplicative. Because the data was paired differences between the two trial texts were cancelled entirely by the differences between pairs (quotient of raw data).

The VODKA input was slower than soft keyboard by a factor of only 1.11 (from log mean of 0.104). The individual variation was high with one subject nearly twice as slow. Taking into account the variation of the data we can say that at 99% confidence VODKA is on average no slower than 1.21 times the rate of a normal soft keyboard.

Given the novelty of the method for users and the need to further refine the details of the interface this is an encouraging result and makes it a clearly acceptable option where other considerations, such as screen real estate, are critical.

A number of negative comments were reported. The choice of animation for the keyboard did receive some criticism; the frame rate was too quick and the excessive motion made it difficult sometimes to locate the correct control. However, this did appear to improve as the user became familiar with the approach. Therefore, although, the interaction is trivial to understand, there was some difficulty in acquiring the necessary skill, which is possibly due to the unfamiliar design, a bit like using the mouse for the first time.

Finally, the design was to support single handed use, by supporting the device in the palm and entering the gesture with a thumbnail, sadly the physical size of the Ipaq meant that only a few of the users could achieve this, however it is possible a smaller device, such as a mobile phone could support this style of interaction.

9 Conclusion

This paper has illustrated the constraints on and issues relating to, the development of text input for mobile and wearable devices, as illustrated by Dunlop & Crossan [1999] and MacKenzie et al. [1999]. User interface overloading presents a viable approach to screen real estate optimisation and touch screen interaction, offering new twists on the constraints of developing handheld and public access interfaces (see Figures 1 & 5). Solution to these problems and shortcomings of existing schemes were introduced and discussed. A prototype of the solution, making use of user interface overloading, was implemented and evaluated against a set of derived requirements. It was argued that this prototype makes effective use of screen area, yet preserves the portability of the device. The results clearly indicate the approach is comparable with the better input methods available, moreover, the benefits, such as savings in screen area make it a promising candidate, which is full of potential.

This paper has challenged the accepted perspective and assumptions of graphical user interface design to develop this novel user interface overloading model, which integrates agreeably with common windowing systems offering effective, additional tools and functionality rather than the unrealistic proposition of a replacement model or significant remodelling of accepted designs.

10 Further Work

Our current work involves investigating the application of our techniques to support interaction for Databoards, public information kiosks, small devices, such as wearable devices [Masui 1998] and control dashboards for augmented and virtual reality interfaces. We are exploring the effectiveness of UIO itself, and seek to improve touch screen interaction, among other things.

We also intend to explore the use of VODKA in a predictive text application. Consider entering the specific first letter of a word and using a gesture to define the length of a word, then tapping on successive groups of letters, (as with the T9 dictionary), to generate a list of possibilities. Although, with VODKA it remains possible to enter specific letters in order to refine to search.

There are some other aspects of UIO we wish to explore. We noticed that users could perceive controls with indirect gaze making the model useful in peripheral displays, adaptive systems [Hudson & Parkes 2003b; McGuffin & Balakrishnan 2002] and designing interaction for the visually impaired, such as macular degeneration. Adaptive displays could also benefit from the freedom to place new items or reconfigure displays without upsetting the layout of controls.

Another property is, elements sharing the same motion appear grouped together, suggesting this approach could be used to implement widely dispersed menu options on a display without the necessary overhead of bounding them in borders, as is usually required to suggest a group relationship.

Finally, we recognise that our future research will benefit from an investigation into theories of perception. Such work may help us to minimise, and govern the effects of, *visual rivalry*, perhaps by introducing 3D elements and dynamic shading.

References

Bartlett, J. [1992], Transparent Controls for Interactive Graphics, WRL Technical Note TN-30, Digital Equipment Corporation.

Belge, M., Lokuge, I. & Rivers, D. [1993], Back to the Future: A Graphical Layering System Inspired by Transparent Paper, *in* S. Ashlund, K. Mullet, A. Henderson, E. Hollnagel & T. White (eds.), *INTERACT'93 and CHI'93 Conference Companion on Human Factors in Computing Systems*, ACM Press/IOS Press, pp.129–30.

Bier, E. A., Stone, M. C., Fishkin, K., Buxton, W. & Baudel, T. [1994], A Taxonomy of See-through Tools, *in* B. Adelson, S. Dumais & J. Olson (eds.), *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Celebrating Interdependence (CHI'94)*, ACM Press, pp.358–64.

Cox, S., Linford, P., Hill, W. & Johnston, R. [1998], Towards Speech Recognizer Assessment Using a Human Reference Standard, *Computer Speech and Language* **12**(4), 375–91.

Dix, A., Finlay, J., Abowd, G. & Beale, R. [1998], *Human–Computer Interaction*, second edition, Prentice–Hall Europe.

Dunlop, M. & Crossan, A. [1999], Dictionary Based Text Entry Method for Mobile Phones, *in* S. Brewster & M. Dunlop (eds.), *Proceedings of Second Workshop on Human Computer Interaction with Mobile Devices*, Springer-Verlag.

Goldstein, M. & Chincholle, D. [1999], The Finger-joint Gesture Wearable Keypad, *in* S. Brewster & M. Dunlop (eds.), *Proceedings of Second Workshop on Human Computer Interaction with Mobile Devices*, Springer-Verlag.

Harrison, B. & Vicente, K. [1996], An Experimental Evaluation of Transparent Menu Usage, *in* M. J. Tauber, B. Nardi & G. C. van der Veer (eds.), *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Common Ground (CHI'96)*, ACM Press, pp.391–8.

Harrison, B., Ishii, H., Vicente, K. & Buxton, W. [1995], Transparent Layered User Interfaces: An Evaluation of a Display Design to Enhance Focused and Divided Attention, *in* I. Katz, R. Mack, L. Marks, M. B. Rosson & J. Nielsen (eds.), *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'95)*, ACM Press, pp.317–24.

Hudson, J. & Parkes, A. [2003a], Novel Interaction Style for Handheld Devices, *in* K. Anind, A. Schmidt & J. F. McCarthy (eds.), *Adjunct Proceedings of UBICOMP'03*, Springer-Verlag, pp.52–5.

Hudson, J. & Parkes, A. [2003b], Visual Overloading, *in* C. Stephanidis & J. Jacko (eds.), *Human–Computer Interaction, Theory and Practice (Part II). Volume 2 of the Proceedings of Human–Computer Interaction International 2003*, Vol. 2, Lawrence Erlbaum Associates, pp.67–8.

Kamba, T., Elson, S., Harpold, T., Stamper, T. & Sukaviriya, P. [1996], Using Small Screen Space More Effi ciently, *in* M. J. Tauber, B. Nardi & G. C. van der Veer (eds.), *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Common Ground (CHI'96)*, ACM Press, pp.383–90.

Kölsch, M. & Turk, M. [2002], Keyboards without Keyboards: A Survey of Virtual Keyboards, Technical Report 2002-21, Department of Computer Science, University of California, Santa Barbara. http://www.create.ucsb.edu/sims/PDFs/Koelsch_and_Turk_SIMS.pdf.

Kurtenbach, G. & Buxton, W. [1994], User Learning and Performance with Marking Menus, *in* B. Adelson, S. Dumais & J. Olson (eds.), *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Celebrating Interdependence (CHI'94)*, ACM Press, pp.258–64.

Lokuge, I. & Suguru, I. [1995], GeoSpace: An Interactive Visualization System for Exploring Complex Information Spaces, *in* I. Katz, R. Mack, L. Marks, M. B. Rosson & J. Nielsen (eds.), *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'95)*, ACM Press, pp.409–14.

MacKenzie, I. S. & Zhang, S. X. [1999], The Design and Evaluation of a High-performance Soft Keyboard, *in* M. G. Williams & M. W. Altom (eds.), *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: The CHI is the Limit (CHI'99)*, ACM Press, pp.25–31.

MacKenzie, I., Zhang, S. & Soukoreff, W. [1999], Text Entry using Soft Keyboards:, *Behaviour & Information Technology* **18**(17), 235–44.

Mankoff, J. & Abowd, G. [1998], Cirrin: A Word-level Unistroke Keyboard for Pen Input, *in* E. Mynatt & R. J. K. Jacob (eds.), *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology, UIST'98*, ACM Press, pp.213–4.

Masui, T. [1998], An Effi cient Text Input Method for Pen-based Computers, *in* M. E. Atwood, C.-M. Karat, A. Lund, J. Coutaz & J. Karat (eds.), *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'98)*, ACM Press, pp.328–35.

McGuffi n, M. & Balakrishnan, R. [2002], Acquisition of Expanding Targets, *in* D. Wixon (ed.), *Proceedings of SIGCHI Conference on Human Factors in Computing Systems: Changing our World, Changing Ourselves (CHI'02), CHI Letters* **4**(1), ACM Press, pp.57–64.

Meyer, A. [1995], Pen Computing. A Technology Overview and a Vision, *ACM SIGCHI Bulletin* **27**(3), 46–90.

Sears, A., Revis, D., Swatski, J., Crittenden, R. & Shneiderman, B. [1993], Investigating Touchscreen Typing: The Effect of Keyboard Size on Typing Speed, *Behaviour & Information Technology* **12**(1), 17–22.

Silvers, R. [1995], Livemap — A System for Viewing Multiple Transparent and Time-varying Planes in Three Dimensional Space, *in* J. Miller, I. Katz, R. Mack & L. Marks (eds.), *Conference Companion of the CHI'95 Conference on Human Factors in Computing Systems*, ACM Press, pp.200–1.

Ward, J., Blackwell, A. & MacKay, D. [2000], Dasher — a Data Entry Interface Using Continuous Gestures and Language Models, *in* M. Ackerman & K. Edwards (eds.), *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology, UIST'00, CHI Letters* **2**(2), ACM Press, pp.129–37.

Zhai, S., Hunter, M. & Smith, B. A. [2000], The Metropolis Keyboard: An Exploration of Quantitative Techniques for Virtual Keyboard Design, *in* M. Ackerman & K. Edwards (eds.), *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology, UIST'00, CHI Letters* **2**(2), ACM Press, pp.119–28.

Author Index

Keyword Index