Tilt-Based Automatic Zooming and Scaling in Mobile Devices – a state-space implementation

Parisa Eslambolchilar[1], Roderick Murray-Smith[1,2]

[1] Hamilton Institute, National University of Ireland, NUI, Maynooth, Co.Kildare , Ireland
parisa.eslambolchilar@may.ie

[2] Department of Computing Science, Glasgow University, Glasgow G12 8QQ, Scotland
rod@dcs.gla.ac.uk

Abstract. We provide a dynamic systems interpretation of the coupling of internal states involved in speed-dependent automatic zooming, and test our implementation on a text browser on a Pocket PC instrumented with an accelerometer. The dynamic systems approach to the design of such continuous interaction interfaces allows the incorporation of analytical tools and constructive techniques from manual and automatic control theory. We illustrate experimental results of the use of the proposed coupled navigation and zooming interface with classical scroll and zoom alternatives.

1 Introduction

Navigation techniques such as scrolling (or panning) and zooming are essential components of mobile device applications such as map browsing and reading text documents, allowing the user access to a larger information space than can be viewed on the small screen. Scrolling allows the user to move to different locations, while zooming allows the user to view a target at different scales. However, the restrictions in screen space on mobile devices make it difficult to browse a large document efficiently. Using the traditional scroll bar, the user must move back and forth between the document and the scroll bar, which can increase the effort required to use the interface. In addition, in a long document, a small movement of the handle can cause a sudden jump to a distant location, resulting in disorientation and frustration.

Speed-dependent automatic zooming is a relatively new navigation technique [7, 8, 14, 22, 25, 26] that unifies rate-based scrolling and zooming to overcome these limitations. The user controls the scrolling speed only, and the system automatically adjusts the zoom level so that the speed of visual flow across the screen remains constant. Using this technique, the user can smoothly locate a distant target in a large document without having to manually interweave zooming and scrolling, and without becoming disoriented by extreme visual flow.

In this paper we demonstrate that, as suggested by Igarashi and Hinckley [14], SDAZ is well suited to implementation on mobile devices instrumented with tilt sensors, which can then be comfortably controlled in a single-handed fashion. We also describe an alternative stylus controlled implementation for the PocketPC. A further

contribution is the use of a state-space formulation of speed dependent zooming, which we believe is a promising reformulation of the technique, which opens the path to the use of analytic tools from optimal and manual control theory.

2 Speed-dependent automatic zooming – a brief review

Several techniques have been proposed to improve the manipulation of scroll bars [14, 19]. They allow the user to control scrolling speed, enabling fine positioning in large documents. LensBar [18] combines these techniques with interactive filtering and semantic zooming, and also provides explicit control of zooming via horizontal motion of the mouse cursor. A rate-based scrolling interface is described in [29] that maps displacement of the input device to the velocity of scrolling.

Zoomable user interfaces, such as Pad and Pad++ [4], use continuous zooming as a central navigation tool. The objects are spatially organized in an infinite two-dimensional information space, and the user accesses a target object using panning and zooming operations. A notable problem with the original zoomable interfaces is that they require explicit control of both panning and zooming, and it is sometimes difficult for the user to coordinate them. The user can get lost in the infinite information space [16]. Bimanual approaches also exist, such as that of Guiard *et al.* [11] where a joystick in one hand controlled zoom level, and a mouse in the other provided navigation. They showed that by using zooming interfaces, bit rates far beyond those possible in physical selection tasks become possible.

Information visualization techniques, such as Fisheye Views [9, 12], Perspective Wall [17], and the Document Lens [21] also address the problem of information overload by distorting the view of documents. The focused area is magnified, while the non-focused areas are squashed but remain in spatial context. The user specifies the next focal point by clicking or panning. Van Wijk derived an optimal trajectory for panning and zooming in [24], for known start and end points.

The particular input device used can also influence the effectiveness of rate control. An experiment on 6 DOF input control [29] showed that rate control is more effective with isometric or elastic devices, because of their self-centring nature. It is also reported that an isometric rate-control joystick [2] can surpass a traditional scroll bar and a mouse with a finger wheel [29]. Another possibility is to change the rate of scrolling or panning in response to tilt, as demonstrated by Rekimoto [20] as well as Harrison *et al.* [13], suitable for small screen devices like mobiles phones and PDAs.

A common problem with scrolling and zooming interfaces is that when users are zoomed out for orientation, there is not enough detail to do any ‘real work’. When they are zoomed in sufficiently to see detail, the context is lost. To reduce this problem, multiple windows can be provided, each with pan and zoom capability. Although this is reasonable for small information spaces, the many windows required by large spaces often lead to usability problems due to excessive screen clutter and window overlap. An alternative strategy is to have one window containing a small overview, while a second window shows a large more detailed view [3, 10]. The small overview contains a rectangle that can be moved and resized, and its contents are shown at a larger scale in the large view. This strategy, however, requires extra space for the

overview and forces the viewer to mentally integrate the detail and context views. An operational overhead is also required, because the user must regularly move the mouse between the detail and context windows.

Speed-dependent automatic zooming (SDAZ) is a navigation technique first proposed by Igarashi & Hinckley [14]. It couples rate-based scrolling with automatic zooming to overcome the limitations of typical scrolling interfaces and to prevent extreme visual flow. This means that as a user scrolls faster the system automatically zooms out, providing a constant information flow across the screen. This allows users to efficiently scroll a document without having to manually switch between zooming and scrolling or becoming disoriented by fast visual flow, and results in a smooth curve in the space-scale diagram. In traditional manual zooming interfaces, the user has to interleave zooming and scrolling (or panning); thus the resulting pan-zoom trajectory forms a zigzag line. Cockburn *et al.* [7, 8, 22, 25, 26] presented further developments, with a usability study of performance-improved SDAZ prototypes.

3 Dynamics and interaction

In this paper we use systems of differential equations to describe the interaction between user and computer. Skeptics might question this *"Why introduce dynamics, when dynamic systems tend to be more difficult to control than static ones? Vehicle control systems tend to go to great trouble to hide the underlying dynamics of the vehicle from the driver."*

We explicitly include dynamics because we can only control what we can perceive, and while, in principle, we can navigate instantly in an arbitrary information space, given a static interaction mechanism (e.g. clicking on a scroll bar), if we are dependent on feedback to be displayed while pursuing our goals, there will be upper limits on the speed at which the display can change. This is especially true in cases where there is uncertainty in the user's mind about where to go, and when they have the option to change their goal on route, as more information becomes available. In order to cope with this, interface designers have a long history of hand-crafting transition effects in a case-by-case manner. Nonlinear mouse transfer functions are long-established examples of finely-tuned dynamic systems driven by user input.

One of our long-term goals is to investigate whether describing the dynamics of interaction using the tools of control engineers allows us a more consistent approach to analyzing, developing and comparing the *'look-and-feel'* of an interface, or in control terms, the *'handling qualities'*. Control synthesis often focuses on analysis of coupling among system states. Speed-dependent zooming is an obvious example of this, but if we generalize the approach to other interaction scenarios, with possibly a larger number of interacting states/inputs, we will require more general methods to analyse the consequences of coupling effects. Control methods are likely to be especially important for design for mobile devices, where sensor noise, disturbance rejection, sensor fusion, adaptive self-calibration and incorporating models of human control behaviour are all important research challenges.

In cases such as the use of accelerometers as input devices, the direct mapping of acceleration in the real world to acceleration in the interface provides an intuitive

mapping, which also suggests a range of other affordances, especially for multi-modal feedback, which can then be utilized by interface designers. Real-world effects such as haptic feedback of springs, or friction linked to speed of motion are easy to reproduce in a dynamic system, and we can choose to explicitly use these features to design the system to encourage interaction to fall into a comfortable, natural rhythm. Furthermore, the act of performing a continuous input trajectory to achieve a goal, creates proprioceptive feedback for the user which can then be associated with that particular task. The mechanisms of gesture recognition can be 'opened up' and explicitly made visible *during* the motion, to provide a link for the user between the control input and the task completion. We describe a probabilistic, audio/vibrotactile approach to this in [28], which can ease learning and reduce frustration.

The use of dynamic models of interaction allows *intelligent interaction*, if the handling qualities of the dynamics of the interface are adapted depending on current *inferred* user goals. Using this approach, actions require less effort, the more likely the system's interpretations of user intentions, equivalent to a fewer bits from the user, in communication terms. This was used by Barrett *et al.* in [2], and we used this approach for text entry in Williamson & Murray-Smith [27], and the approach can be linked to methods which adapt the control-to-display ratio, such as Blanch *et al.* [5] in classical windows interfaces. These approaches, which work with relative input mechanisms, cannot be used if we use static mappings, such as a stylus touching an explicit point on the screen.

4 Speed-dependent automatic zooming on a mobile device

Implementing the SDAZ technique on a mobile device with inertial sensing allows us to investigate a number of issues: the use of single-handed tilt-controlled navigation, which does not involve obscuring the small display; the usability consequences of tilting the display; the relative strength of stylus-based speed-dependent zooming, compared to mouse and tilt-based control, and combinations of stylus, and tilt-based control. If successful, the user should be able to target a position quickly without becoming annoyed or disoriented by extreme visual flow, and we want the technique to provide smooth transitions between the magnified local view and the global overview, without the user having to manually change the document magnification factor.

4.1 Hardware/software environment

We implemented this method using Embedded Visual C++ on an HP 5450 Pocket PC (Figure 1). Here, tilting the device moves the zooming-window. The accelerometer (Xsens P^3C, 3 degree-of-freedom linear accelerometer) attached to the serial port of the Pocket PC provides the roll and pitch angles.

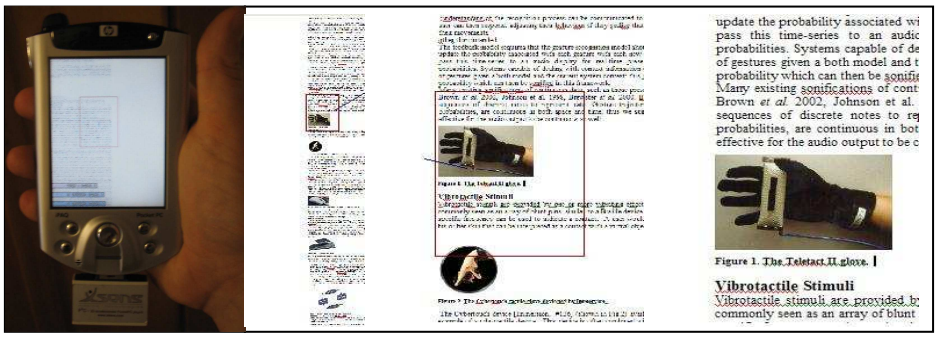

Fig. 1a **Fig. 1b**

Fig. 1. PocketPC and accelerometer attached to serial port (1a). Screen shots of the document browser (1b). The left picture shows a red box moving rapidly over the picture, the middle picture shows the user has found the picture and landing there, and right picture shows the zoomed-in picture.

4.2 Design and implementation of Speed-dependent automatic zooming

State space modelling is a well-established way of presenting differential equations describing a dynamic system as a set of first-order differential equations. There is a wealth of knowledge and analysis techniques from systems theory, including designing estimators and controllers for multi–input–multi–output systems, optimal control, disturbance rejection, stability analysis and manual control theory [6]. State-space modelling allows us to model the internal dynamics of the system, as well as the overall input/output relationship as in transfer functions, so this method is an obvious candidate for the representation of the coupling between the user's speed with zoom level. There are many advantages to modelling systems in state space, especially for multivariable problems, where the matrix formulation is particularly useful for analysis purposes.

4.2.1 State space model

For an introduction to the basic ideas, see any introductory control theory book, e.g. [1,6]. The generic form for the state equations is given by equation (1)

$$\begin{aligned} \dot{X} &= f(x) + g(u) \\ \dot{Y} &= h(x) \end{aligned} \tag{1}$$

where f(x), g(u) and h(x) can be nonlinear functions, and where $X(t)$ is an $n \times 1$ state vector where n is the number of states or system order, $U(t)$ is a $r \times 1$ input vector where r is the number of input functions, and $Y(t)$ is a $p \times 1$ output vector where p is the number of outputs. The more specific case of a linear system, (2)

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned} \tag{2}$$

where *A* is an $n \times n$ square matrix called the *system matrix, B* is an $n \times r$ matrix called the *input matrix, C* is a $p \times n$ matrix called the *output matrix and D* is a $p \times r$ matrix which represents any direct connection between the input and output.

4.2.2 Coupling the user's velocity with the zoom-level

In this section we show how an SDAZ-like approach couples the user's motion with the zoom-level. The inputs to the system are the tilting angles measured using an accelerometer attached to the serial port of PDA, and in a second experiment the stylus position on the PDA touch screen. The state variables chosen are $x_1(t)$ for position, $x_2(t)$ for speed of scroll and $x_3(t)$ for zoom, and the state equations are:

$$x_2(t) = V = \Delta x_1 \tag{3}$$

$$x_3(t) = Z = f(x_1, x_2, u) \tag{4}$$

So the zoom-level is a function of position, velocity and tilting angle. An initial suggestion is to reproduce the standard second-order dynamics of a mass-spring-damper system, in the hope that giving the scrolling movement and zoom level some inertia will provide a physically intuitive interface. The first time-derivative of the state equations can be written as below, as a linearization of the system at a given velocity and zoom:

$$\dot{x}_1(t) = V = x_2(t) \tag{5}$$

$$\dot{x}_2(t) = \dot{V} = \frac{-R}{M} x_2(t) + \frac{k}{M} u(t) \tag{6}$$

$$\dot{x}_3(t) = \dot{Z} = \frac{-b}{M} x_2(t) + \frac{-R}{M} x_3(t) + \frac{a}{M} u(t) \tag{7}$$

The standard matrix format of these equations is:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & \frac{-R}{M} & 0 \\ 0 & \frac{-b}{M} & \frac{-R}{M} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{k}{M} \\ \frac{a}{M} \end{bmatrix} u(t) \tag{8}$$

This shows how a single-degree of freedom input can control both velocity and zoom-level. The non-zero off-diagonal elements of the *A* matrix indicate coupling among states, and the *B* matrix indicates how the inputs affect each state. This example could be represented as having zoom as an output equation, rather than state, and the coupling between zoom and speed comes only through the *B* matrix, which is not particularly satisfying. However, this paper is intended as an initial exploration of the area, and as more interesting behaviour can be obtained by fully interacting nonlinear equations, such as those elegantly derived by van Wijk in [24], we have left it in this format. In the experiments, *R=1, M=1, k=1 and b=0*, but we also experimented with varying the parameters, essentially including nonlinearities by a function relating ve-

locity with zoom factor, as will be discussed in the next section. We include saturation terms for maximum and minimum zoom levels, and there can be specific rules for behaviour at the limits associated with the start and end of the document. For nonlinear functions we can locally linearise around any given state [*x v z*] leading to time-varying matrices *A(t),B(t).* We can analytically investigate the local dynamics for different operating points by, for example, looking at the eigenvalues of the *A* & *B* matrices to check for oscillatory (eigenvalues are complex conjugate pairs) or unstable behaviour (real part of eigenvalue in right half plane – i.e. positive). For more background see any control textbook (e.g. [1, 6]). Importantly, the system itself might be stable, but when coupled with the time delay and lead-lag-dynamics of typical human control behaviour, the combined closed loop system might be unstable, as in pilot-induced oscillations in aircraft control [15,23].

The dynamic systems implementation allows us to deviate from a static link between speed and zoom level. In this paper, our basic assumption is that zoom should lead speed when speed increases, in order to avoid extreme visual flow. Zoom should, however, lag speed when $|v|$ decreases, to allow the user to slow down but still maintain the overview. This also allows, for example, the user to zoom out, without changing position in the document, by repeated positive and negative acceleration.

In order to move more rapidly through the document at high levels of zoom, in this paper, we adapted *B* by making '*a*' in eqn. (8) a function of velocity. When speed is above the dead-zone threshold (here set to 0.1), *a = 3* but below this threshold *a=0.* We wish to avoid rapid drop effects when user changes direction. To achieve this, we set *a=a*0.2,* when the sign of velocity and input differ. For practical implementation on a PDA we converted the continuous-time system to a discrete-time one [1], with sampling time *h*, which involves the evaluation of a matrix exponential, $\Phi = e^{Ah}, \quad \Gamma = \int_0^h e^{As} ds B$.

$$x(kh+h) = \Phi x(kh) + \Gamma u(kh) \qquad (9)$$
$$y(kh) = Cx(kh) + Du(kh)$$

A phase plane figure shows an example of a trajectory through this state-space for the SDAZ implementation on the Pocket PC (Figure 2). This gives some insight into the transient dynamics of large and small translations of position through the document.

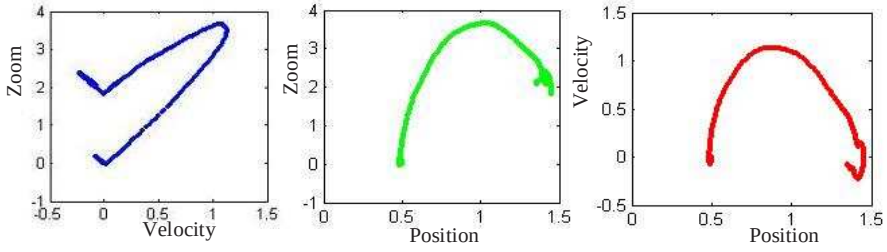

Fig.2. Phase plane trajectories showing velocity against zoom (left), zoom-level against position (centre) and velocity against position (right), from a record of participant browsing a long document on the PocketPC.

4.2.3 Control mode

We can now introduce transitions among control modes which alter the dynamics and the way user inputs are interpreted. A simple example of this approach uses state feedback to augment control behaviour, by making the state move towards some reference value *r*, we can create a control law $u = L(r - x)$, such that the new state equations are

$$\begin{aligned} \dot{x} &= Ax + Bu = Ax - BLx + BLr \\ &= (A - BL)x + BLr \end{aligned} \tag{10}$$

such that the system dynamics have changed from *A* to *(A-BL)*. In the SDAZ implementation in this paper, we switched from tilt-angle as acceleration, to tilt angle to indicate desired velocity, as soon as the speed passed the threshold at which zooming started. This made it easier for users to find and maintain a comfortable zoom level. Other similar examples can be created, where the interpretation of sensor inputs and their significance for control can adapt to context. Including position control, for example, would allow the user to tap on the screen to specify a goal, which is then dynamically acquired. While on route to that goal, the user changes their mind, they can break out and switch again to velocity control.

4.2.4 Calibrating SDAZ and the state space approach

SDAZ has many parameters that can be tuned, usually treated as a series of interacting, but essentially separate equations. The state-space formulation allows multiple variables, and derivative effects (e.g. position, velocity, acceleration) can be coupled with zoom level, without any further coding, by just changing the entries of the *A* matrix, simulating combinations of springs, masses and damping effects.

In SDAZ, the function linking zoom with velocity, $z = f(v)$, can be nonlinear, including threshold effects. Examples include linear, with thresholds, exponential, and 'modified exponential' [14,25]. Furthermore the document velocity $v=g(\delta)$ as a function of control input (mouse displacement, tilt-angle, or stylus displacement, depending on platform) tend to be static, linear, or piecewise linear functions [14, 25]. In the state-space representation, we need to reformulate these equations in terms of the time-derivatives of zoom and velocity, via the *A* and *B* matrices. For example, for ramp increases in speed, the modified exponential zoom-speed mapping corresponds to our suggestion of zoom leading speed, with the exponent being related to the difference between the time constants for zoom and speed.

To enhance the smoothness of the transition between the global overview and the magnified local view after a mouse button is pressed, Cockburn and Savage use a 'falling' speed, and Igarashi & Hinckley [14] place a limit on the maximum time-derivative of zoom, with similar effect. The falling rate was calculated using trial and error – if the rate was too fast, the user felt motion sickness and lost their place in the document, whereas it being too small led to a sluggish interface. This can be represented as a straightforward switch to a particular parameterization of the *A* matrix, which can be tuned to give an appropriate exponential decay in velocity or zoom.

Related problems include rapid zooming in and out when making a rapid change of direction [14]. In the state-space representation, dealing with these issues becomes a

matter of tuning the dynamics of the system by changing the *A* matrix, to make, for example, the time-constants associated with the zoom level larger than that of the speed, for regimes where speed is dropping.

Gutwin [12], Igarashi & Hinckley [14] and Wallace [25] report the *hunting effect* problem when users overshoot the target due to the system zooming in as the user slows, the user then rapidly adjusts behaviour to compensate, which causes the system to zoom out again. One approach to this would be to switch to a 'diving' control mode if $dz/dt < z_{thresh}$, where $a=0$, preventing zooming increases, unless a major change in velocity, occurs, which would switch the control mode back to velocity control.

5 Example application – document browser for a PDA

The document viewer was designed to use automatic zooming to browse PDF, PS and DOC files which had been converted to a image (PNG) file. BMP or PNG (Portable Network Graphics) files are more efficient, and have low rendering time. This increases the speed and smoothness of the browser, the implementation of which was simple but very efficient and smooth (although text tended to flicker during zooming because it was treated as a flat image). Equations (15) to (18) (previous section) show the formula used to calculate the relationship between the user's hand motion (tilting PDA) and the zoom level from the document.

For comparison we show trajectories of users using traditional scroll bars on the Pocket PC and a touch-screen based SDAZ implementation (Figure 3) for browsing a long document on PDA (Figure. 1b). The touch-screen based SDAZ and tilt-controlled SDAZ both use the same state-space model. The results in Figure 3 highlight the different navigation styles of the different interfaces, with the scroll bar approach using a number of rapid translations through the document to find a paragraph in bottom of the document, and no use of zooming for an overview, while the two SDAZ implementations had smoother navigation, which also included smooth changes in zoom level.

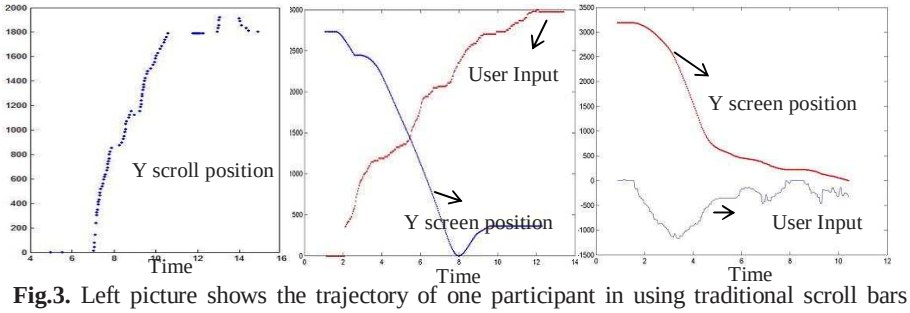

Fig.3. Left picture shows the trajectory of one participant in using traditional scroll bars in browsing the long document, so *y* displacement is as long as the document. Middle picture shows the trajectory of the same participant in touch screen based SDAZ in browsing the long.

Users found the touch screen-based mechanism intuitive and easy to use for browsing. Figure 4 presents the system's inputs in three SDAZ applications to find the same paragraph used in scroll bar browser for tilt-based and touch screen controlled SDAZ.

Also this figure presents an example run with tilt-based SDAZ, with augmented velocity control, as described in section 4.2.3, to browse the document to find 7 main headings. For comparison, the central plots in Figure 4 show tilt-based SDAZ without augmented velocity control on the same task, where fluctuations indicate that controlling the zoom level was difficult, and hunting behaviour appears when users tried to land on the targets (e.g. *t*=20,40, 85, in middle figures).

6 User feedback

We asked five users from our research lab to work with the document browser using tilt-based SDAZ and touch screen-controlled SDAZ with and without augmented velocity control. Users who did the experiment without augmented velocity control suggested that adding a control option or a switch to control the zoom-level with velocity and tilting angles will make the system more comfortable to use. Most of them proposed if they could control level of zoom by tapping on the screen or pressing a key on PDA, the application would be easier to use.

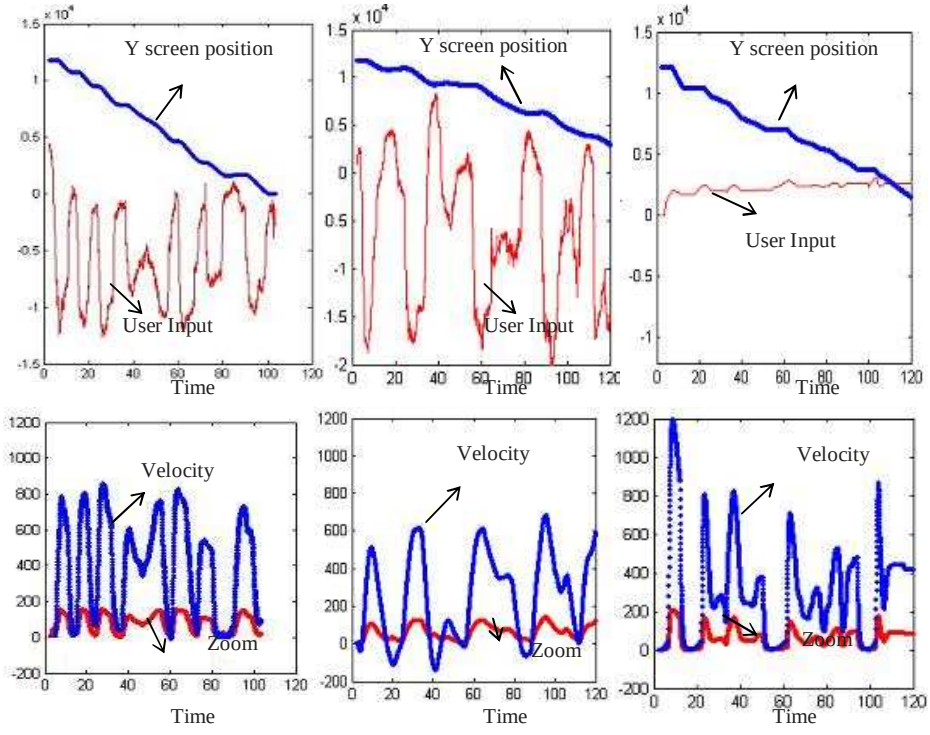

Fig.4. Left picture tilt-based SDAZ with augmented velocity control, middle picture tilt-based SDAZ without augmented control and right picture touch-screen controlled SDAZ.

In contrast, users who did their experiments with augmented velocity control were satisfied with the application in both tilt-based and touch screen-controlled modes. Some users complained that with tilt input, they had to tilt the device to angles which caused irritating reflections from the PocketPC screen. Users in both groups, with and without augmented control, commented that if they were involved with other tasks, (like answering the phone, working with PC, etc.) they would prefer the touch screen-controlled SDAZ because they imagined it would be difficult to stay in the desired position in the document, with a tilt-based SDAZ. Although this was beyond the scope of our initial experiments, a key factor in the usefulness of tilt-based SDAZ will be the ease with which the user can toggle tilt-control on and off, during tasks.

7 Conclusions

We have presented a state-space, dynamic systems representation of the dynamic coupling involved in speed-dependent automatic zooming. We demonstrated the applicability of the approach by implementing a speed-dependent zooming interface for a text browsing system on a PDA instrumented with an accelerometer, and with stylus control. We illustrated the behaviour of the different interfaces by plotting their trajectories in phase space and as time-series.

Initial informal user evaluation of the implementation of SDAZ on a Pocket PC is positive, and users felt that this provided an intuitive solution to the problem of large documents and small displays. The tilt-controlled version can be used in a single-handed manner, without obscuring the screen, but because in the implementation tested, there was no toggle for tilt-control, users felt more comfortable with the stylus-controlled version.

This approach has the potential to provide a very general framework for development, analysis and optimisation of interfaces which induce complex, but convenient coupling among multiple states, in order to cope with few degrees of freedom in input. It opens up the dynamics of the ‘look and feel’ of mobile applications based on continuous control metaphors, to analysis and design techniques from automatic and manual control theory [15, 23].

References

[1] K. Åström, J., Wittenmark, B., *Computer controlled systems, Theory and Design*, 3rd ed. NJ: Prentice Hall, 1997.

[2] R. C. Barrett, E. J. Sleker, J. D. Rutledge, and R. S. Olyha, "The Negative Inertia: A dynamic pointing function," in CHI'95, conference companion, 1995, pp. 316-317.

[3] D. Beard and J. Walker, "Navigational techniques to improve the display of large two-dimensional spaces," *Behav. Inf. Tech*, vol. 9(6), pp. 451-466, 1990.

[4] B. Bederson and J. Meyer, "Implementing a Zooming User Interface: Experience Building Pad++," *Journal of Software - Practice and Experience*, vol. 28(10), pp. 1101-1135, 1998.

[5] R. Blanch, Y. Guiard, M. Beaudouin-Lafon, "Semantic Pointing: Improving target acquisition with Control-Display ratio adaptation," *Proc. of ACM Conference on Human Factors in Computing Systems*, pp. 519- 526, 2003.

[6] W. L. Brogan, *Modern Control Theory*, 3rd ed: NJ: Prentice Hall, 1991.

[7] A. Cockburn, J. Looser, and J. Savage, "Around the World in Seconds with Speed-Dependent Automatic Zooming," in Demonstration in the Proceedings of the ACM User Interface Software and Technology (UIST Conference Supplement), Vancouver, Canada, 2003, pp. 35-36.

[8] A. Cockburn and J. Savage, "Comparing Speed-Dependent Automatic Zooming with Traditional Scroll, Pan, and Zoom Methods," in People and Computers XVII: British Computer Society Conference on Human Computer Interaction, Bath, England, 2003, pp. 87-102.

[9] G. W. Furnas, "Generalized Fisheye Views," in Proceedings of CHI'86, 1986, pp. 16-23.

[10] G. W. a. B. B. B. Furnas, "Space-Scale Diagrams: Understanding Multiscale Interfaces," *Proceedings of CHI'95*, 1995.

[11] Y. Guiard, F. Bourgeois, D. Mottet, and M. Beaudouin-Lafon, "Beyond the 10-Bit Barrier: Fitts' Law in Multiscale Electronic Worlds," in Proceedings of IHM-HCI, 2001, pp. 573-587.

[12] C. Gutwin, "Improving focus targeting in interactive fisheye views," *Proc. of ACM Conference on Human Factors in Computing Systems (CHI'02), Minneapolis*, pp. p267-274, 2002.

[13] B. Harrison, K. P. Fishkin, et al., "Squeeze Me, Hold Me, Tilt Me! An Exploration of Manipulative User Interfaces," *Proceedings of CHI'98.*, 1998.

[14] T. Igarashi and K. Hinckely, "Automatic speed-dependent zooming for browsing large documents," in 13th Annual Symposium on User Interface Software and Technology, ACM UIST, San Diego, CA, 2000, pp. 139-148.

[15] R. J. Jagacinski and J. M. Flach, *Control Theory for Humans*: Lawrence Erlbaum Associates, 2003.

[16] S. Jul and G. Furnas, "Critical Zones in Desert Fog: Aids to Multiscale Navigation," in Proceedings of UIST '98, 1998, pp. 97-106.

[17] J. D. Mackinlay, G. G. Robertson, and C. K. Card, "The Perspective Wall: Detail and Context Smoothly Integrated," in Proceedings of CHI'91, 1991, pp. 173-179.

[18] T. Masui, "LensBar - Visualization for Browsing and Filtering Large Lists of Data.," in Proceedings of InfoVis'98, 1998, pp. 113-120.

[19] T. Masui, K. Kashiwagi, and G. R. Borden, "Elastic graphical interfaces for precise data manipulation," in CHI'95, Conference Companion, 1995, pp. 143-144.

[20] J. Rekimoto, "Tilting Operations for Small Screen Interfaces," in Proceedings of UIST, 1996, pp. 167-168.

[21] G. G. Robertson and J. D. Mackinlay, "The Document Lens," in Proceedings of UIST'93, 1993, pp. 101-108.

[22] J. Savage, "Speed-dependent automatic zooming," University of Canterbury, 2002. http://www.cosc.canterbury.ac.nz/research/reports/HonsReps/2002/hons_0208.pdf.

[23] T. B. Sheridan and W. R. Ferrell, *Man-Machine Systems: Information, Control, and Decision Models of Human Performance*: MIT Press, 1974.

[24] J. J. Van Wijk, "Smooth and efficient zooming and panning," *T. Munzner, S. North (eds.), Proceedings IEEE Symposium on Information Visualization (InfoVis'2003)*, pp. 15-22, 2003.

[25] A. Wallace, "The Calibration and Optimisation of Speed-Dependent Automatic Zooming," University of Canterbury, Christchurch, New Zealand November 2003. http://www.cosc.canterbury.ac.nz/research/reports/HonsReps/2003/hons_0303.pdf.

[26] A. Wallace, J. Savage, and A. Cockburn, "Rapid Visual Flow: How Fast is Too Fast?," in Proceedings of the Fifth Australasian User Interface Conference (AUIC2004), Dunedin, New Zealand, 2004, pp. 117-122.

[27] J. Williamson and R. Murray-Smith, "Dynamics and probabilistic text entry," Department of Computing Science, Glasgow University DCS Technical Report TR-2003-147, June 2003.

[28] J. Williamson and R. Murray-Smith, "Granular synthesis for display of time-varying probability densities", International Workshop on Interactive Sonification (Human Interaction with Auditory Displays), eds. A. Hunt, Th. Hermann, Bielefeld, Germany, January 2004

[29] S. Zhai, B. A. Smith, and T. Selker, " Improving Browsing Performance: A Study of Four Input Devices for Scrolling and Pointing Tasks," in INTERACT, 1997, pp. 286-292.

Acknowledgements

The authors gratefully acknowledge the support of SFI BRG project *Continuous Gestural Interaction with Mobile devices*, Science Foundation Ireland grant 00/PI.1/C067, the *MAC network* - EC TMR grant HPRN-CT-1999-00107, and EPSRC grant GR/R98105/01.,