Tilt-Based Automatic Zooming and Scaling in Mobile Devices – A State-Space Implementation

Parisa Eslambolchilar[1] and 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.

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.

4.1 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.

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)

$$\dot{X} = f(x) + g(u) \qquad (1)$$
$$\dot{Y} = h(x)$$

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)

$$\dot{x}(t) = Ax(t) + Bu(t) \qquad (2)$$
$$y(t) = Cx(t) + Du(t)$$

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:

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.