E

Elastic Interfaces for Visual Data Browsing

Wolfgang Hürst
Albert-Ludwigs-Universität Freiburg, Germany

INTRODUCTION

In this article, we discuss the concept of elastic interfaces, which was originally introduced by Masui, Kashiwagi, and Borden (1995) a decade ago for the manipulation of discrete, time-independent data. It gained recent attraction again by our own work in which we adapted and extended it in order to use it in a couple of other applications, most importantly in the context of continuous, time-dependent documents (Hürst & Götz, 2004; Hürst, Götz, & Lauer, 2004). The basic idea of an elastic interface is illustrated in Figure 1. Normally, objects are moved by dragging them directly to the target position (direct positioning). With elastic interfaces, the object follows the cursor or mouse pointer on its way to the target position with a speed s that is a function of the distance d between the cursor and the object. They are called elastic because the behavior can be explained by the rubber-band metaphor, in which the connection between the cursor and the object is seen as a rubber band: The more the band is stretched, the stronger the force between the object and the cursor gets, which makes the object move faster. Once the object and cursor come closer to each other, the pressure on the rubber band decreases, thus slowing down the object's movement.

In the next section we describe when and why elastic interfaces are commonly used and review related approaches. Afterward, we illustrate different scenarios and applications in which elastic interfaces have been used successfully for visual data browsing, that is, for skimming and navigating through visual data. First, we review the work done by Masui (1998) and Masui et al. (1995) in the context of discrete, time-independent data. Then we describe our own work, which applies the concept of elastic interfaces to continuous, time-dependent media streams. In addition, we discuss specific aspects considering the integration of such an elastic behavior into common GUIs (graphical user interfaces) and introduce a new interface design that is especially useful in context with multimedia-document skimming.

BACKGROUND

Direct positioning is usually the approach of choice when an object has to be placed at a specific target position. However, elastic interfaces have advantages in situations in which the main goal is not to move the object itself, but in which its movements are mapped to the motion of another object. The

Figure 1. Illustration of the concept of elastic interfaces

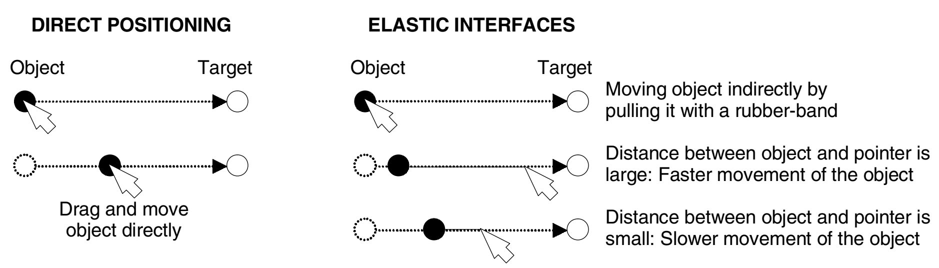
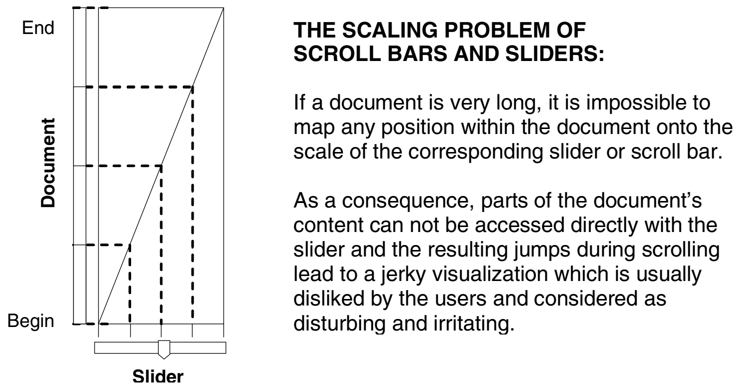


Figure 2. Illustration of the scaling problem of scroll bars and slider



most typical examples for such a case are scroll bar and slider interfaces, for which the dragging of the scroll bar or slider thumb to a target position is mapped to the corresponding movements within an associated document. One common problem with scroll bars and sliders is that a random document length has to be matched to their scale, which is limited by window size and screen resolution (the scaling problem; compare Figure 2). Hence, if the document is very long, specific parts of the file are not accessible directly because being able to access any random position of the document would require movements of the slider thumb on a subpixel level. This is impossible with direct manipulation since a pixel is the smallest unit to display (and thus to manipulate) on the screen. In addition, the movement of the document's content during scrolling becomes rather jerky, which is usually considered irritating and disturbing by users. This is where elastic interfaces come into play: Since the scrolling speed is indirectly manipulated based on the mapping of the distances between cursor and thumb to a corresponding speed, navigation becomes independent of the scroll bar's or slider's scale and thus independent of the actual length of the document. If the function for the distance-to-speed mapping is chosen appropriately, subpixel movements of the thumb and thus slow scrolling on a finer scale can be simulated.

Other solutions to solve the scaling problem have been proposed in the past, mainly as extensions or replacements of slider interfaces. The basic functionality of a slider is to select a single value or entry by moving the slider thumb along the slider bar, which usually represents an interval of values. A typical selection task is, for example, the modification of the three values of an RGB color by three different sliders, one for each component. If visual feedback is given in real-time, sliders can also be used for navigation either in a continuous, time-dependent media file, such as a video clip, or to modify the currently visible part of a static, time-independent document whose borders expand beyond the size of its window (similar to the usage of a scroll bar). In both cases, again, the user drags the thumb along the bar in order to select a single value. In the first case, this value is a specific point in time (or the corresponding frame of the video), and in the second case, it is a specific position in the document (and the task is to position the corresponding content within the visible area of the screen).

Most approaches that try to avoid the scaling problem have been proposed either for selection tasks or for scrolling interfaces that enable navigation in static, time-independent data. The most well known is probably the Alphaslider introduced by Ahlberg and Shneiderman (1994). Here, the thumb of a slider or a scroll bar is split into three different areas, each of which allows for navigation at a different granularity level. Ayatsuka, Rekimoto, and Matsuoka (1998) proposed the Popup Vernier in which the user is able to switch between different scrolling resolutions by using additional buttons or keys. Instead of relying on different granularities for