Inter-Office Memorandum

To	Tim Townsend	Date	7 November 1977
From	Bob Metcalfe, Charles Simonyi	Location	Palo Alto
Subject	Scrolling and Panning	Organization	SDD/SD/SA

XEROX

This memo responds to your request for a characterization of the performance implications of horizontal scrolling, now called panning. In this memo we use the term scrolling to mean vertical movement and panning to mean horizontal movement, both of a page image on a display. The purpose of scrolling and panning is, of course, to enable viewing of a page which is too large for the display, perhaps because a larger, more legible font has been employed for user comfort.

In your request you mentioned what we now call an EDP page consisting of 60 lines of 132 fixed pitch characters in a single font. Our response treats the more general Diamond page case first and then examines possible performance improvements for the EDP case.

There are three page representations relevant to this discussion: (1) piece tables, (2) line descriptions, and (3) bit maps. Page edits are performed on piece tables. Piece tables are converted to line descriptions enroute to various other representations, one of which is the bit map maintained for the display.

Given a new piece table or one upon which an edit has just been performed, we estimate that 1 second of D0 processing will be required to compute the bit map viewed by a user on his display, just as specified in the requirements. It should be noted that a lower delay will be perceived because the bit map is displayed as it is updated.

Because scrolling and panning imply no modification of the piece table and often keep much of the same bit map in view, recomputing the bit map can frequently and to great advantage be avoided using the so-called BITBLT operation to move the bit map unaltered. We estimate that the display can be updated in .1 seconds when a scroll or pan operation keeps the display's view inside the existing bit map. Because memory is expensive, however, the amount of bit map maintained is only as large as that viewed. Therefore, a scroll or pan operation commonly results in a view which requires that new bit map be computed. In the worst case an entirely new bit map is required. In the common case only a small portion of the existing bit map is scrolled or panned out of view and only a comparably small segment of new bit map need be computed. Here is where the difference between scrolling and panning is felt.

Owing to the left-to-right and top-to-bottom nature of pages and owing to our efficient representation of pages which exploits this nature, the bit map recomputations of incremental scrolling are small relative to complete bit map recomputation. We estimate that scrolling updates require between .1 seconds and 1 second, depending of course on the fraction of the viewed page which remains in view after scrolling. Panning, however, does not benefit as does scrolling from the nature of pages and our representation of them. Using current Diamond algorithms, as the view of a page moves to the right or left, past the maintained bit map, total recomputation of the bit map is required.

Four alternatives have been considered for improving panning response. The first is to save the state of bit map computations at their boundaries so that they can be extended left and right incrementally. We now judge this alternative to be impractical.

A second alternative is to compute and maintain a large bit map even for a small display so that panning would require no bit map recomputation, only BITBLT. Scrolling cannot be handled this way because documents grow indefinitely long; a reasonable upper bound on width, however, is feasible. This is an attractive alternative except for the observation that a larger display is likely to be inexpensive relative to the bit map memory required.

A third alternative is to maintain, rather than a complete bit map, a complete line description of pages to be viewed on a small display. A line description requires only 30% of the computation required for a piece table to generate bit map and occupies between 25% and 50% of the space of its equivalent bit map. By maintaining page line descriptions, panning updates could be reduced to .3 seconds from 1 second at half the memory cost of a full page bit map. This alternative looks attractive, but requires further study.

A fourth alternative is to adopt a page representation specific to the EDP page. By removing font and spacing generality, complete bit map recomputations could, we estimate, be reduced straightforwardly to .2 seconds. Some penalty would of course be paid in developing this special representation, in coding various conversions, and in document load and store conversion delays.

Yet another alternative is to do nothing and have panning take 1 second to complete. This speed may prove to be acceptable, especially because a lower delay may be perceived.

It is our conclusion that panning is feasible and can be made to perform acceptably. We recommend further study, of course, but suggest at this time that wider displays be employed or, if not, that the line description approach in the third alternative be pursued.

c: David Liddle
Wendell Shultz
Jerry Szelong