The Art of Navigating
Hypertext \cite{3,19,25} is becoming a popular approach to many computer applications, especially those dealing with the on-line presentation of large amounts of loosely structured information such as on-line documentation or computer-aided learning. However, many of which are in the user interface area. One concern concerning hypertext that remain to be resolved, their individual interaction history can be used to increase the sense of context in the information space.

\textit{Users often become lost when exploring hypertext networks, but}}

\textit{jakob nielsen}
the user interface area. One of the major usability problems with hypertext is the user's risk of disorientation while navigating the information space. For example, our studies [23] showed that 56 percent of the readers of a document written in one of the most popular commercial hypertext systems agreed fully or partly with the statement I was often confused about 'where I was.'

To investigate a number of user interface options in hypertext systems we designed a prototype system in the form of a hypertext report on events at the 1987 hypertext workshop. This system was implemented on an Apple Macintosh with HyperCard as the programming system. To get a feel for our hypertext system, the reader is encouraged to review Figures 1 to 10 which contain screen dumps of a session with the system and thus constitute a kind of printed demonstration or guided tour. Hypertext is a very dynamic form of non-sequential writing: a directed graph, where each node contains some amount of text or other information. The nodes are connected by directed links. In most hypertext systems, a node may have several out-going links, each of which is then associated with some smaller part of the node called an anchor. When users activate an anchor, they follow the associated link to its destination node, thus navigating the hypertext network. Users backtrack by following the links they have used in navigation in the reverse direction. Landmarks are nodes which are especially prominent in the network, for example by being directly accessible from many (or all) other nodes.

Many other computer techniques may match this definition at least partly, but true hypertext should also make users feel that they can move freely through the information according to their own needs. This feeling is hard to define precisely but certainly implies short response times and low cognitive load when navigating.

### HYPERTEXT

Hypertext is non-sequential writing: a directed graph, where each node contains some amount of text or other information. The nodes are connected by directed links. In most hypertext systems, a node may have several out-going links, each of which is then associated with some smaller part of the node called an anchor. When users activate an anchor, they follow the associated link to its destination node, thus navigating the hypertext network. Users backtrack by following the links they have used in navigation in the reverse direction. Landmarks are nodes which are especially prominent in the network, for example by being directly accessible from many (or all) other nodes.

Many other computer techniques may match this definition at least partly, but true hypertext should also make users feel that they can move freely through the information according to their own needs. This feeling is hard to define precisely but certainly implies short response times and low cognitive load when navigating.

### NAVIGATION FACILITIES

This hypertext system has two navigational dimensions: A linear dimension used to move back and forth among the text pages within a given node, and a non-linear dimension used for hypertext jumps. To reinforce users' understanding of these two dimensions, two different animation techniques are used when shifting from one screen to another. Unfortunately, animation is the one aspect of our user interface which is impossible to show well in a set of printed figures, so it

---

**Version 1 of the hypertext system described in this article is available on a Macintosh format CD-ROM on the BMUG PD-ROM vol. 1 [1988] [path stacks/Hyper Media : HyperText Stacks Folder : HyperTEXT '87 Trip Report], published by Discovery Systems, 7001 Discovery Blvd., Dublin, Ohio 43017. U.S.A. Readers with electronic mail access can download Version 1 of the system (the version described in this article) from the Info-Mac archives as hypercard/hypertext-report-part1.hqx and ...part2.hqx (the Info-Mac archives are available by using FTP (account anonymous, any password) in the info-mac directory on sumex-aim.stanford.edu [36.44.0.6]). One can also get a copy by sending electronic mail to listserv@rice.rice.edu or listserv@rice.bitnet with the following two lines in the body of the message:

```
MACARCH GET HYPERCARD/HYPERTEXT-REPORT-PART1.Hqx
MACARCH GET HYPERCARD/HYPERTEXT-REPORT-PART2.Hqx
```

These two files should be concatenated and converted by the StuffIt utility which is available in many shareware collections.

© 1990 ACM 0001-0782/90/0300-0298 $1.50
is not included in the figures in this article.

Movement between pages within a node is seen as a linear left/right dimension, corresponding to the orientation of the scroll bars at the bottom of the screen and to the way printed books are read in Western society. A change to a new page along this dimension is visualized by an animated right or left wipe, using built-in visual effects from HyperCard which look quite like the turning of a page.

Hypertext jumps are seen as being orthogonal to the left/right page turning and are visualized as an in/out dimension using an animated iris which opens for an anchored jump and closes for return jumps. The opening iris gives users the impression of diving deeper into the hyperspace when they take a hypertext jump, and the closing iris for return jumps gives the inverse feeling of pulling back again.

We should note that both navigational dimensions are bi-directional, left/right or in/out, and that this bi-directionality is emphasized by the choice of user interface animation techniques.

The Homogeneity Problem
To help users navigate the hyperspace we must help them understand and recognize their present location. This is partly done by facilities for utilizing the interaction history as will be discussed here. Recognizing locations is also facilitated by the dramatically varying graphical designs used for different elements of the system such as e.g. the front cover (Figure 1), the main text (Figure 4), and the literature references (Figure 7).

There is no risk that users will confuse these different parts of the information space.

In an earlier version of the system we also relied on more subtle differences in graphics design such as varying background patterns for different kinds of information. This kind of design difference can be seen by comparing Figures 2 and 3, but the differences were more subtle in other parts of the earlier system where we also used different background patterns for screen designs which were mainly taken up with running text. During user testing of this earlier version it turned out that nobody actually noticed these subtle design differences, so they were removed in the current version of the system. One reason users do not notice different background patterns may be that they are used to the indiscriminate assignment of colors to various screen elements which is allowed as the primary interface customization in many current systems. Therefore, they do not associate screen colors with semantic importance.

The differences in graphical design are intended to reduce the homogeneity problem in on-line text, which basically is that on-line text always looks the same. Text on a computer screen is nicely formatted, normally using a single typefont. Furthermore, it is always displayed on the same screen, so on-line text is extremely homogeneous, which again makes it harder for users to recognize information and get a sense of location. On-line text does not have the variety which traditional text has, due to variations in typefont, book size, color, etc., or even the basic differences in physical looks between, say a real book, a newspaper, and a note written on a napkin in the cafeteria during a lunch break. In this system we have used different fonts to a small extent (see Figure 12) but mostly we have relied on the graphical screen design to introduce a heterogeneous feel. In hypertexts containing more nodes it might not always be possible to rely on distinct graphical design only, so certain subtle forms of variation might still be needed.

FIGURE 1
This is the opening screen of the hypertext system. It includes buttons for getting help and other general information and an option to reset the recorded user history. This screen serves as the "landmark" of the hypertext structure and is accessible from throughout the system by clicking on the Front cover icon.

The most prominent graphic on this screen is a picture of a book which is intended to convey a general sense of the nature of the data in this information base. The user clicks on the name of the author of the book, a new screen will be displayed containing a digitized photo of the author. This facility was helpful in the iterative design of this system since people who had downloaded earlier versions from various computer network services could easily recognize the author at conferences to pass on their comments.

The upper right hand corner of the screen displays a timestamp informing us about the time since we last visited this node in the hypertext system. Since this is the front cover of the book, it also tells us the time since we last used this system.

In this demonstration example, we will assume that we click on the title of the book or on the book cover itself. This will display Figure 2 since the picture of the book itself is the anchor for opening the book. User studies indicate that the use of a book cover and its title as a hypertext anchor is fairly unintuitive for many users. They quickly understand clicking on icons and framed words (like "Quit" in this screen design) but often have to be encouraged to just click on a general graphics picture having no specific target indicated. In this case, a special problem is that the affordance ([9], [24]) of a book cover is to be pulled up while the mechanics of mouse-operated pointers require a push down. Because of this observation, the help screen (not shown here) was changed to explicitly state that the book was opened by clicking on it.

March 1990 Volume 33 Number 3 Communications of the ACM 299
One might argue that homogeneity could be desirable because it emphasizes the book metaphor and because readers can assimilate information faster when they encounter a familiar format. Of course, this is true to some extent but we would actually want to avoid the book metaphor in our future hypertext designs because it seems to limit the conceptual models of the search potential of hypertext and non-linear navigation of the information space. Furthermore, the available empirical evidence suggests that users behave very differently when reading hypertexts than when reading printed texts [22] and that this is even more true for experienced hypertext users who leave their book habits behind [14]. Therefore, it is unnecessary to keep the book metaphor except for walk-up-and-use situations where immediate transfer of past skills is needed. Of course, homogeneity should be retained to the extent that the same information should be presented in the same way to ensure consistent interfaces. For example, we used the same graphical language to encode the overview diagrams both at the global level (Figure 2) and at the local level (Figure 3).

**Overview Diagrams**

Overview diagrams have been used as navigational aids in several earlier hypertext systems. In this system we use two levels of overview diagrams to provide users with both a coarse-grained sense of location in the global information space and a more fine-grained sense of location in the local neighborhood of the current node. This approach works well in the current system which is fairly small, since its information space is divided into only 95 screens containing a text which took just 9 pages in the printed version [18].

For larger hypertext structures, it is an open question whether two levels of overview diagrams will be sufficient. One approach could be to use even more levels according to the principle that there should be “one, two, or enough” of any feature in a computer system. But in big hypertexts, using “enough” levels of diagrams will introduce navigation problems in itself as users move among the diagrams. Alternative designs might be to use fisheye views [8] or to show only the top and bottom-level diagrams and let users interpolate between them.

The overview diagrams in this system have been generated manually, even though some hypertext systems such as NoteCards [11] provide computer-generated diagrams. One reason for this decision was to speed up the implementation of the prototype interface, and another reason was the desire to have two levels of overview diagrams. It is not clear how one could automatically generate multi-level diagrams and how one should identify the important links to be included in the global diagram, even though some research has been done on nested hypertext structures and inheritance mechanisms [6]. One would probably not want all hypertext links from the local diagrams to propagate to the global diagrams, so some mechanism for using weights or relevance measures might be considered to prune the edges of the graph on the higher levels.

It is likely that big hypertexts will be the most useful ones and it is therefore important to address the issues of overview in large information spaces. We are currently working on methods for assigning relevance to links based on an information retrieval measure of similarity [7] between the two linked nodes as well as an estimate of the user’s current interest. However, it is yet too early to judge the usability of such methods for real work. If such metrics can be tuned to correspond to users’ actual intentions it would be possible to filter out links with ratings below some cutoff point. We are also experimenting with graphical methods to differentiate between anchors based on their estimated relevance to the user. Our limited experience indicates that users like this feature since it provides a prospective view on the links. A primitive example of such graphical differ-

---

**Figure 2**
This is the global overview diagram of the contents of the hypertext structure. It serves as a two dimensional table of contents. The 2-D structure was chosen since there is by definition no linear structure of a hypertext. There is no “first” chapter. The layout of the items is intended to suggest their relation; i.e. “Systems” and “Applications” are related, while “Definition of hypertext” is a subissue of “Research issues.”

“Definition of hypertext” is represented directly on the overview map because it is an especially important node in the hypertext network. Users can access the definition node directly throughout the system since the screens of the main text (Figure 4 etc.) include this global overview diagram as an active set of hypertext anchors.

The lines between the items are also intended to give an indication of how they are related since they show that “Hypertext ’87 workshop” is the central topic while “CSG ’86 Trip Report” is a disconnected topic. In fact, there are hypertext links from some of the nodes in the Hypertext ’87 report to some of the nodes in the CSG ’86 report but these links were judged to be of minor importance and are therefore not represented in this overview diagram. Checkmarks indicate the nodes which we have visited in previous use of the system (in this case “Research issues”).

---

![Diagram of overview structure](image-url)
entiation in the current system is shown in Figure 4 where one of the buttons has been made prominent. The differentiation in Figure 4 was statically specified by the author, however, rather than being dynamically tailored for the individual reader as we are trying to do in our current experiments.

Our system is a single user system which only keeps an individual interaction history, but one might speculate whether collective interaction histories could also be useful. For example, having the system keep statistics on how multiple users have moved through the hypertext might provide a means for automatically generating useful overview diagrams. If many users have been observed to have navigated between two specific nodes, it is likely that there should be a line between those nodes on one of the higher levels of overview diagram. And if almost no users have traversed certain links then they would be candidates for pruning. In this way the hypertext structure would constantly evolve towards a "survival of the fittest" nodes and links. Furthermore, it might be possible to use various clustering algorithms and multi-dimensional scaling techniques [28] to lay out the diagrams automatically.

Backtracking

Earlier studies of the usability of videotex systems [17] showed that the backtrack facility was one of the most important navigation facilities, especially for novice users. Users frequently relied on backtrack to save them when they were in any kind of situation which they could not handle. Because of this result, a return arrow is always available in this system to take users back to the locations in the hypertext where they have jumped from.

The system has an unlimited backtrack capability within any one session, meaning that sufficiently many repeated clicks on the return arrow will take one back to the first screen (Figure 1). Since the first screen has a kind of landmark stature in the navigation space, the system also provides a direct hypertext jump to it from the pages of the main text.

The backtrack facility in this system is strictly path-following, meaning that nodes are revisited in exactly the reverse order in which the user originally visited them. This is implemented quite simply as a stack, and every time a node is visited, it is pushed onto the stack. The history list in the system (Figure 10) works in much the same way, except that nodes which are revisited as the result of a backtrack action are placed once more on the history list but removed from the stack. The history list, therefore, includes all the nodes the user has visited in the sequential order of the time of those visits. Even though hypertext is non-sequential, each user linearizes the information in time as part of the process of moving through it.

Because of the principle of following the user's path through the information space exactly, the same node may occur several times in the backtrack stack or on the history list. An alternative would be to do like HyperCard's built-in "Recent" history list (shown in Figure 11) and have each node occur only once. We have chosen the path-following principle to enable users to rely on their memory of their own navigation behavior when interpreting the history list or the result of backtracking. An out-of-context history list like Figure 11 is considerably harder to interpret than a path-following one since it removes information about the transitions between nodes: Users often remember information of the nature "after I was at A, I went to B."

In an earlier version of our hypertext system, we had the facility for reader-added annotations shown in Figure 12 which used structure-oriented backtrack. The idea was that the system could utilize its knowledge about the structure of the hypertext to provide a direct backtrack jump to the location which the user was likely to want to return to, instead of forcing the user to backtrack one step at a time through a number of intermediate screens. It turned out in user testing, however, that users were not able to understand this new kind of backtrack which was inconsistent with the general backtrack in the rest of the system, so we removed it from later versions of the system. It would probably have been possible for users to learn the structure-oriented backtrack if it had been used in more parts of the system instead of just in one isolated part, and it
In many of its screens Glasgow Online does have a back-track facility but it is inconsistently labeled and not always present, so subjects felt more comfortable using the landmark facility for the front screen. From this we conclude that backtrack facilities need to be simple and consistent, so that users can always rely on them as a lifeline to get out of trouble.

INTERACTION HISTORY
This system relies heavily on the user's individual interaction history to provide a sense of context. The backtrack and history list facilities discussed previously are examples of interaction techniques that are defined relative to the actual user's previous personal use of the system.

Timestamps
All user movements between screens are timestamped. When a user returns to a location in the hypertext, the system displays the time which has passed since that user's previous visit to that location. When users visit a location where they have never been before, the system informs them about this fact instead of showing the whole notion of nonsequential text is flawed. If one needs to make certain buttons very large and graphically attractive in order to induce readers to select them first, why bother giving any options at all? The reason is that there are several different classes of readers. Some readers may be experts in the domain of the information base and will know how to navigate it to find the information of specific interest to them. They can certainly do so even though the author has made certain anchors prominent, since readers retain complete freedom of movement. Other readers, however, may be novices who are more in need of guidance instead of forcing people to read in certain ways, our hypertext design allows individual readers to customize their reading to their individual needs and learning styles in yet another way of providing individualized interaction. But since novices do not yet know the structure of the information space and may have difficulties in understanding the meaning of the terminology, we feel that it is reasonable for the author to provide them with hints.
"Hypertext" is non-sequentially linked pieces of text or other information. If the focus of such a system is on non-textual types of information, the term hypermedia is often used instead. Practically the only such hypertext is often radioed documents, so the footnote, so termed "footnote".

The graphic design of this screen is intended to emphasize its landmark status and to differentiate it from the text of the other nodes. The definition is accessible from all the screens of the main text through its anchor in the global overview diagram. Just before the screen dump shown here was taken, we clicked on the hypertext anchor "non-textual types" to get the small pop-up annotation giving examples of these types. The pop-up will go away when we click on it or when we leave the screen to go elsewhere. Hypertext anchors for pop-ups are marked on the screen by a raised asterisk which is intended to invoke the notion of a footnote from traditional books. The asterisk is a different notation from the boxes used for anchors for hypertext links which jump the user to a new screen, since that helps users predict the actions of the system in response to their clicks. Version 1 and 2 of this system had the same box notation for both kinds of link, and users found it very frustrating not knowing in advance whether a click would just pop up a small annotation or take them to an entirely different context.

After we clicked on the anchor for the pop-up annotation, the system automatically put a checkmark in front of it to help us remember that we have already seen this annotation if we return to this page later. The system has also put in a checkmark at the anchor marked "Frank Halasz," since we have also seen the destination for that link (it is the screen in Figure 4). So let us click on "HyperCard" instead to go to Figure 6. Figure 4 in the form of a reference to "Frank Halasz." This kind of double reference is quite common in hypertext, since there is a good chance that if A is relevant to B, then B is relevant to A. However, users sometimes do not know which of the references in B would refer to A, even if they just came from A, so they may happen to activate a jump leading back to A instead of a jump leading them onwards. Checkmarking the anchor leading back alleviates this problem.

One problem with putting footprints on the maps and other hypertext anchors is that almost all parts of the structure will get marked eventually, if the user uses the system extensively. Perhaps a notation similar to that of the experimental Color EMACS text editor [15] could be used where changes to the file are shown in a contrasting color that "decays" over time to gradually match the color of the original text. In a similar way, footprints in a hypertext might erode gradually so that only current footprints were displayed prominently while older footprints were displayed in ever lighter shades of gray or not at all.

One problem with putting footprints on the maps and other hypertext anchors is that almost all parts of the structure will get marked eventually, if the user uses the system extensively. Perhaps a notation similar to that of the experimental Color EMACS text editor [15] could be used where changes to the file are shown in a contrasting color that "decays" over time to gradually match the color of the original text. In a similar way, footprints in a hypertext might erode gradually so that only current footprints were displayed prominently while older footprints were displayed in ever lighter shades of gray or not at all.

Another checkmark issue is whether the system should use one notation for the checkmarks placed at anchors which the user has actually activated, and another for checkmarks placed at anchors which are linked to destinations already seen by the user but
have not been explicitly activated by that user. Our system uses a single notation to simplify both the implementation and the number of interface conventions which the user needs to learn.

**CONTEXT-IN-THE-LARGE VS. CONTEXT-IN-THE-SMALL**

Most of the issues discussed under navigation and interaction history have to do with providing the user with a sense of context and location in the entire hypertext structure. We will call this kind of context for context-in-the-large, but Figure 9 shows that there can also be problems with context-in-the-small in reading a hypertext. When the computer display is as small as in our system, users can only see a very small part of the information at any one time. This means that they can very easily lose track of how the text they are currently reading is related to the immediately preceding or following text, since that text will often not be visible. In reading the text in Figure 9, we understand that we are seeing some kind of enumeration of items, but we have no idea of the author's intention with the list or what kind of list it is.

The previous designs of this hypertext system did not include the dynamic overview diagrams on each screen and thus had room for about twice as much text on each screen. This still did not solve the context-in-the-small problem, but it was much better. In the current design we have added lots of facilities to help with the context-in-the-large problem, such as the overview diagrams and timestamping and icons for the history list and direct access to the front cover. It is likely that this redesign represents a poor trade-off given the constraints of the small screen: The loss of context-in-the-small could well damage overall usability more than the added context-in-the-large improves it. We did the redesign anyway, since we wanted to experiment with these new features which will probably show their usefulness on the larger displays which will be much more common in the near future.

Figure 9 also shows another case of the problems with context-in-the-small. It was necessary to move the entire description of the third list element to a new page to avoid having a figure and the text explaining it split over different pages, which would have had an even worse impact on readability. The result, however, is to distribute the issues discussed over several pages, which means that the reader cannot get an overview of these issues and how they are related. In another part of this hypertext system shown in Figure 8, we have tried to solve a similar problem by having a list of items represented as anchors for hypertext pop-ups each containing the full text for one issue.

Finally, the scroll bars at the bottom of the pages of the main text address a contextual problem on a level somewhat in between the in-the-small and the in-the-large levels since they provide readers with a sense of relative location within the individual hypertext node and an indication of the size of that node as illustrated in Figure 6. In general, we should remember that users need context not just in-the-large but also in-the-small when reading hypertexts, and that different mechanisms are usually required to provide the two kinds of context.

**IMPLEMENTATION ISSUES**

The hypertext system described in this article was implemented in HyperCard using the HyperTalk [1] programming language. Our emphasis in designing and implementing the system was on rapidly demonstrating our ideas for hypertext user interface features and not on producing a generally useful system. As a matter of fact, the system can only display the one hypertext document it was built to support.

HyperTalk is an object-oriented [27] programming language, but not with full generality. For example, HyperTalk does not allow the programmer to define subclasses of objects such as buttons, and this led to problems in the code implementing the automatic checkmarking of anchors leading to previously visited nodes. We wanted to look only at buttons which were hypertext anchors, and not at the buttons in the scroll bars, for example. We had to do this by inspecting the
When we arrive at this node, the system puts checkmarks at the hypertext anchors for “Frank Halasz” and “Classification of HT systems” since we have visited the destinations for those links already.

This screen is part of the node describing Apple’s HyperCard system as can be seen from the highlighting of that node in the local overview diagram and also by the two headings inside the picture of the book page. The bold faced heading describes the location in the global overview diagram while the underlined heading describes the location in the local overview diagram. These two headings are automatically concatenated by the system to form a text string referring to a node in the history list shown in Figure 10.

In a preliminary design of these headings the picture of the book page was smaller and the headings were placed outside the book page to indicate that the headings were a kind of “computer-generated state information” and not part of the running text. Even though the early design placed the headings in the same physical location on the screen as the current design, it did not work since test users did not view the text outside the book page as being a label for the contents of the book page. This result could be explained by the gestalt law of closure which states that things which are within the same closed region are seen as corresponding, so the solution was simply to move the headings within the closure of a larger book page.

The little black and white boxes below the book page form a scroll bar which indicates two things: Our relative location within the current node and the size of the node. The length of the scroll bar is proportional to the number of pages in the node, so the scroll bar in Figure 4 is longer than the scroll bar in this figure because the node visited in Figure 4 is bigger than node we visit here. Proportional scroll bars make it possible to always have the “thumb” of the scroll bar (the black box) be the same size, corresponding to the fact that the book pages have a fixed size. The relative size of the thumb compared to the entire scroll bar indicates the proportion of the total text which the user is currently seeing.

The black box indicates our relative position within the node in an analog format. An alternative would have been a digital format (e.g. “page 1 of 2”) but the scroll bar design allows us also to use it as an anchor for direct hypertext jumps to the other pages within the node: When the user clicks in the scroll bar, the system jumps to the page in the node which has the relative location corresponding to the click.

Here, we see that the current node is quite small and we are on its first page, so we might decide to look at the rest of the node by clicking on the right arrow button. The next page looks similar to the except that it has a left arrow button instead of a right arrow button, so it is not shown here. The next page has a hypertext reference to “Van Nevar Bush” which we follow to get a reference to this famous author.

message handler associated with each button at runtime to see whether its text contained the word “jump” which was a procedure call used only in the code for hypertext jumps. Furthermore, since HyperTalk does not have instance variables for button objects, we had no clean way to get a reference to the node which the button linked to. Again, we had to go and inspect the button code to grab the destination address from the parameter list for the call of the “jump” procedure.

Many other “dirty” programming techniques were used in the implementation of this hypertext system [29], but the end result was that we were able to get a working system in a very short time. It should be remembered that the goal of the project was to get a running system for user interface experiments and not to study the programming techniques which one would have to use for a commercial hypertext product.

For the development of this prototype hypertext system, first 12 hours were spent writing the actual text which forms the information base used by the system. Then 17 hours were spent on software development to establish features for moving through the information space, and finally 11 hours were spent reorganizing the basic text into objects fitted to the software and adding hypertext links. After this original implementation, an additional 3 hours were spent on adding scroll bars to the interface. This feature was added quickly and shows that it was possible to modify the prototype according to user feedback.

For the third version of the prototype, major modi-
As indicated by the drastically different graphic design, this screen is part of the literature reference section of the information space. The left and right arrows are for moving alphabetically through the reference list. The alphabetic order of authors' last names is an access mechanism with very low semantic meaning, so to utilize hypertext principles, the books on the bookshelf have been designed as hypertext anchors to help users discover literature references. For instance, by clicking on the book marked "Poetry" we can jump to a literature reference about the use of hypertext for teaching poetry.

Some of the book buttons are linked to multiple destinations, one of which is chosen by random. For example, the book marked "Classics" will jump to one of the references considered to be classics in the hypertext field. This randomness was inspired by the "Something Else" feature in the Drexel Disk [13] and should only be used in situations, where users are exploring the information space in a browsing mode. In most situations, predictability would of course be the preferred interaction characteristic, and we actually had one user complain that he had spent almost an hour experimenting with the book buttons in a vain attempt to discover the algorithm used to select their destination. This user was a computer scientist, however, and it is likely that more ordinary users would ascribe the system's behavior to "magic" [28] and not worry too much about it. We do not have enough evidence from our usability studies to decide this design issue for certain since a more detailed study of effects of randomness in user interfaces would be required. Based on this limited experience, our intuition suggests that it would actually be better to go for a predictable design.

One way of doing so would be to use the Intermedia [29] technique of popping up a menu listing the possible destinations whenever the user activated an anchor linked to multiple destinations.

From the screen in this figure we could click on the "Read excerpt" button to see some quotes from the original article by Vannevar Bush. Following automated cross reference to other articles is one of the great benefits of hypertext, but we will not do so in this example for copyright reasons. (The copyright problem is exactly one of the worst socio-economic barriers to realizing the full potential of hypertext.) Instead we click several times on the backtrack ("return") arrow in the lower right corner of the screen, backtracking through the intervening screens until we return to the screen shown in Figure 4.

As indicated by the drastically different graphic design, this screen is part of the literature reference section of the information space. The left and right arrows are for moving alphabetically through the reference list. The alphabetic order of authors' last names is an access mechanism with very low semantic meaning, so to utilize hypertext principles, the books on the bookshelf have been designed as hypertext anchors to help users discover literature references. For instance, by clicking on the book marked "Poetry" we can jump to a literature reference about the use of hypertext for teaching poetry. Some of the book buttons are linked to multiple destinations, one of which is chosen by random. For example, the book marked "Classics" will jump to one of the references considered to be classics in the hypertext field. This randomness was inspired by the "Something Else" feature in the Drexel Disk [13] and should only be used in situations, where users are exploring the information space in a browsing mode. In most situations, predictability would of course be the preferred interaction characteristic, and we actually had one user complain that he had spent almost an hour experimenting with the book buttons in a vain attempt to discover the algorithm used to select their destination. This user was a computer scientist, however, and it is likely that more ordinary users would ascribe the system's behavior to "magic" [28] and not worry too much about it. We do not have enough evidence from our usability studies to decide this design issue for certain since a more detailed study of effects of randomness in user interfaces would be required. Based on this limited experience, our intuition suggests that it would actually be better to go for a predictable design.

One way of doing so would be to use the Intermedia [29] technique of popping up a menu listing the possible destinations whenever the user activated an anchor linked to multiple destinations.

From the screen in this figure we could click on the "Read excerpt" button to see some quotes from the original article by Vannevar Bush. Following automated cross reference to other articles is one of the great benefits of hypertext, but we will not do so in this example for copyright reasons. (The copyright problem is exactly one of the worst socio-economic barriers to realizing the full potential of hypertext.) Instead we click several times on the backtrack ("return") arrow in the lower right corner of the screen, backtracking through the intervening screens until we return to the screen shown in Figure 4.

We have backtracked to the screen shown in Figure 4 and from there we have read through the rest of the node, ending up on the page shown here. This part of the text contains a list of seven issues which are hypertext anchors for pop-up windows. We have already seen issues 2, 3, and 5 (as indicated by the checkmarks put in by the system) and here we have just clicked on the anchor for issue 7 to get the screen to look like this.

We can now read the full text describing issue 7 while still being able to see the list outlining the full set of issues. Because of the small screen, the pop-up unfortunately obscures the overview diagrams, but we can always click on it to make it go away.
2) The “docuverse” is the most interesting, but we are building “docu-islands” in the form of isolated, not cross-linked information structures. Systems are closed, incompatible, and without the possibility for data transfer. Instead linking information between systems should be part of an open system conforming to a standard.*

To get from Figure 8 to this screen we have wandered a little bit around the hyperspace and finally happened to click on “People” in the global overview diagram and then on “Andy van Dam” in the local overview diagram which resulted from that action. We have then read through some of the pages in the node until we came to this list of hypertext research issues. This design has some problems in-the-small since not all the issues can be listed on the same screen. All the text for “issue 3” had to be moved to the next page to avoid breaking it up and getting even worse context.

In addition to these three major version of the system, several subversions were released as part of the iterative design based on user tests. Each of these minor changes required only between 15 and 30 minutes to implement.

HyperCard proved to be very effective for our prototyping of graphical hypertext user interfaces. One reason for this was because the author was already an experienced HyperTalk programmer before starting this project. However, learning to implement user interfaces in HyperCard is very fast, compared with most other graphical and/or object-oriented programming systems, partly because it is very concrete in its use of traditional screen painting tools as the graphical programming language of the system. Many parts of an interface can be prototyped simply by painting them on the screen, and even the parts which are under dynamic program control behave in much the same way as one would directly operate the screen during non-programming use of HyperCard.

CONCLUSIONS
The hypertext principle can be useful for many things.
In fact, this very article has some characteristics which would have made it suitable for presentation via a hypertext system, since it relies to a large extent on a non-linear discussion through the figures and captions and since it has many internal cross-references between the figure captions and the main text.

Hypertext, on the other hand, also has serious usability problems which must be solved before hypertext systems can see truly widespread use. This article has made as the result of comments from Uffe Lyngbaek, since it relies to a large extent on a hypertext system, since it has many internal cross-references between the figure captions and the main text.

Acknowledgements. Many people have made constructive comments on earlier versions of this hypertext system. Specific changes to the interface have been made as the result of comments from Uffe Lyngbaek, Cathy Marshall, and Ben Shneiderman, but they are of course not responsible for the way I have chosen to address their criticisms. Helpful comments on earlier versions of this article have been given by Peter Brown, James D. Foley, Christian Gram, Lynda Hardman, Uffe Lyngbaek, Kurt Narmark, John Schnizlein, Ben Shneiderman, and Patricia Wright as well as by anonymous Communications referees.

REFERENCES
This is an alternative history list generated by HyperCard's built-in Recent command. Each miniature represents one screen which we have visited in the current session (navigation behavior from earlier sessions is not saved) and each screen is only represented once in the list, no matter how many times we have visited it. You go to a screen by clicking on its miniature.

One reason we chose to use the textual history list in Figure 10 instead of this graphical history list is that it is very hard to distinguish the miniatures for screens which basically have the same graphical layout but contain different text.

An annotation page from version 1 of the hypertext system. The graphical design is intended to convey the impression that this is some short added material which is separate from the main report. A different typefont is used to indicate that the annotations will normally have been written by people other than the author of the main hypertext.

Since a single node in the hypertext could have several associated annotations, a user asking to read annotations would first be taken to a screen showing a menu of the annotations for the current node (not shown here). The user could then select to read an annotation based on the menu's listing of the author and subjects of the annotations.

This screen design has two backtrack arrows since there are presumably two different ways the user might want to return from this node (which has no further hypertext jumps available since the system did not provide a facility for adding annotations to the comments themselves). The standard return arrow performs the standard backtrack function of returning the user to the departure point for the previous hypertext jump which in this case is the menu of annotations. An extra arrow to the left of the standard return arrow is an "express return" which returns readers directly to the location from which they jumped to the annotation menu, thus providing a shortcut through the straight backtrack path. It turned out, however, that almost no test users understood this facility.

Comment by Jakob Nielsen

Please add your own comments freely to this stack.

You will get an empty comment-card to write on by clicking the button "Add your own comment" which appears in most areas of this information space.

At the moment this stack contains the following total number of comments: 13
Articles


CR Categories and Subject Descriptors: H.1.2 [Models and Principles]: User/Machine Systems—human factors; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—search process; H.4.3 [Information Systems Applications]: Types of Systems—hypertext; J.7 [Computers in Other Systems]—publishing

General Terms: Design, Experimentation, Human Factors

Additional Key Words and Phrases: Context, HyperCard, hypermedia, interaction history, user navigation

ABOUT THE AUTHOR:

JAKOB NIELSEN is assistant professor of user interface design at the Technical University of Denmark. His research interests include usability engineering and hypertext. Author’s Present Address: Jakob Nielsen, Technical University of Denmark, Department of Computer Science, Building 344, DK-2800 Lyngby, Copenhagen, Denmark. datJN@NEUVM1.bitnet

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

---

**CHI ’90**

*The Conference on: HUMAN FACTORS IN COMPUTING SYSTEMS*

**IS MORE THAN A CONFERENCE -- IT IS AN EVENT!!**

One that is considered a mandatory experience for everyone who designs interactive software and computing systems. CHI ’90, the seventh in this annual series, is internationally recognized as the premier forum for human-computer interaction.

CHI ’90 is sponsored by the Association for Computing Machinery’s Special Interest Group on Computer and Human Interaction, in cooperation with the IEEE Computer Society, ACM/SIGGRAPH, the Human Factors Society, the Cognitive Science Society, and six other international societies and special interest groups.

**APRIL 1-5, 1990**

**SEATTLE, WASHINGTON**

---

For further information, or an advance program contact:

CHI ’90 Conference
Attn: Toni MacHaffie, P.O. Box 5847, Beaverton, OR 97006
(503)-591-1981 * FAX: (503)-624-3934 * Email: machaffie.chi@xerox.com

---

Communications of the ACM
March 1990 Volume 33 Number 3