Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays

Christopher Ahlberg and Ben Shneiderman*
Department of Computer Science,
Human-Computer Interaction Laboratory & Institute for Systems Research
University of Maryland, College Park, MD 20742
email: ahlberg@cs.chalmers.se, ben @cs.umd.edu

ABSTRACT
This paper offers new principles for visual information seeking (VIS). A key concept is to support browsing, which is distinguished from familiar query composition and information retrieval because of its emphasis on rapid filtering to reduce result sets, progressive refinement of search parameters, continuous reformulation of goals, and visual scanning to identify results. VIS principles developed include: dynamic query filters (query parameters are rapidly adjusted with sliders, buttons, maps, etc.), starfield displays (two-dimensional scatterplots to structure result sets and zooming to reduce clutter), and tight coupling (interrelating query components to preserve display invariants and support progressive refinement combined with an emphasis on using search output to foster search input). A FilmFinder prototype using a movie database demonstrates these principles in a VIS environment.

KEYWORDS: database query, dynamic queries, information seeking, tight coupling, starfield displays

INTRODUCTION
In studying visual information seeking (VIS) systems for expert and first time users, we have found several user interface design principles that consistently lead to high levels of satisfaction. This paper defines these principles and presents a novel VIS system, the FilmFinder.

The exploration of large information spaces has remained a challenging task even as parallel hardware architectures, high-bandwidth network connections, large high-speed disks, and modern database management systems have proliferated. Indeed, these advances have left many users with the feeling that they are falling further behind and cannot cope with the flood of information [3, 18]. Now, the user interface design principles for VIS have the potential to reduce our anxiety about the flood, find needles in haystacks, support exploratory browsing to develop intuition, find patterns and exceptions, and even make browsing fun.

The key to these principles is understanding the enormous capacity for human visual information processing. By presenting information visually and allowing dynamic user control through direct manipulation principles, it is possible to traverse large information spaces and facilitate comprehension with reduced anxiety [14,16]. In a few tenths of a second, humans can recognize features in mega-pixel displays, recall related images, and identify anomalies. Current displays of textual and numeric information can be extended to incorporate spatial displays in which related information is clustered in 2-dimensional or higher spaces. This use of proximity coding, plus color coding, size coding, animated presentations, and user-controlled selections enable users to explore large information spaces rapidly and reliably.

KEY CONCEPTS
The principles of direct manipulation were a good starting point for design of visual information seeking applications [16]:
- visual representation of the world of action including both the objects and actions
- rapid, incremental and reversible actions
- selection by pointing (not typing)
- immediate and continuous display of results

However, when designing systems especially for information seeking tasks [11], additional principles are needed. A key VIS principle is to support browsing, which is distinguished from familiar concepts of query composition and information retrieval because of its emphasis on rapid filtering to reduce result sets, progressive refinement of search parameters, continuous reformulation of goals, and visual scanning to identify results. These goals are supported by the VIS designs developed in this paper:
- dynamic query filters: query parameters are rapidly adjusted with sliders, buttons, etc.
- starfield display: result sets are continuously available and support viewing of hundreds or thousands of items

* Current address: Dept. of Computer Science, Chalmers University of Technology, S-412 96 Göteborg, Sweden

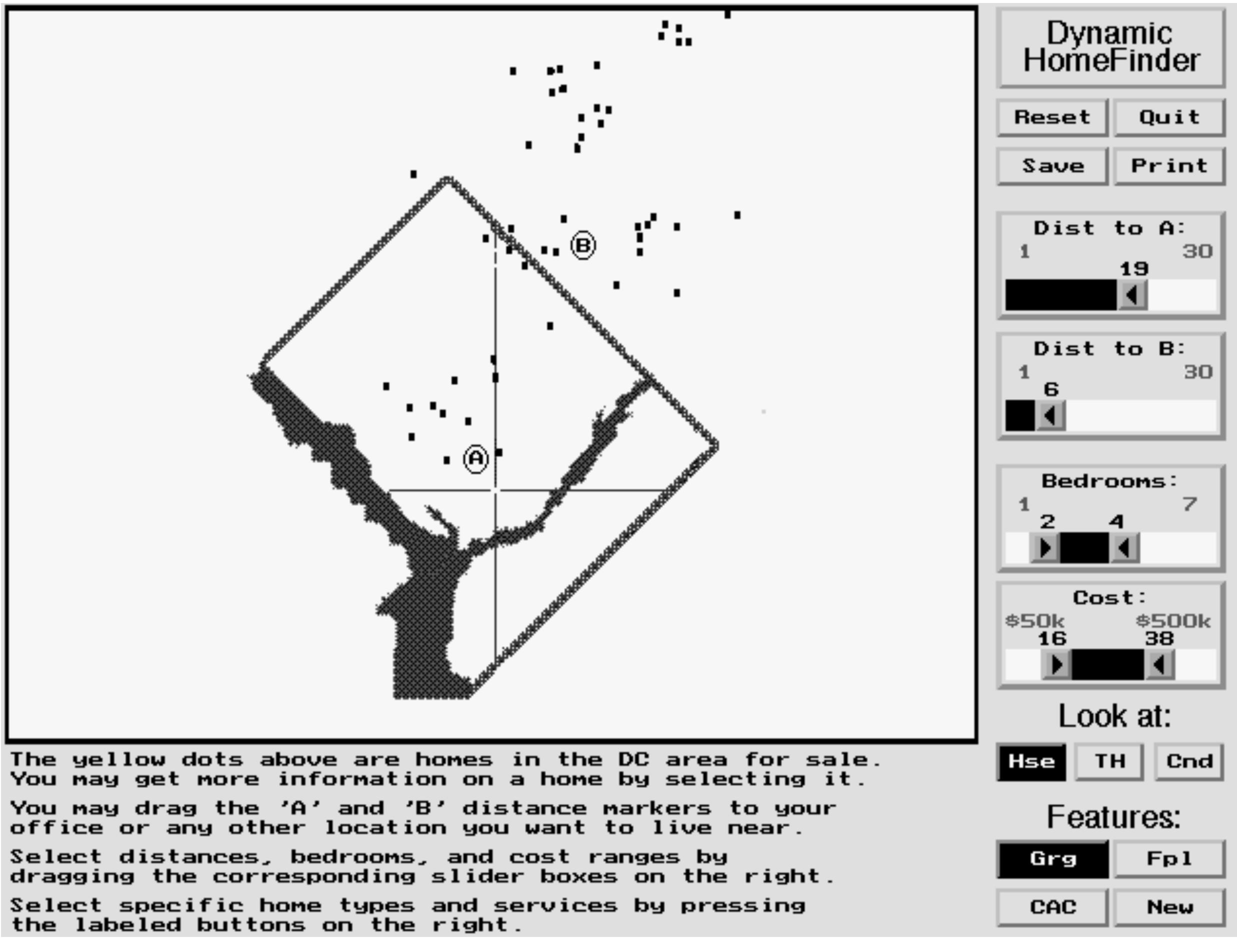


Figure 1: In the Dynamic HomeFinder query system each point satisfies the query described by the sliders for location, cost, number of bedrooms, home type (house, townhouse, or condominium), and buttons (Garage, Fireplace, Central Air Conditioning, or New construction). The points of light can be selected to generate a detailed description.

- tight coupling: query components are interrelated in ways that preserve display invariants and support progressive refinement. Specifically, outputs of queries can be easily used as input to produce other queries.

Dynamic Query Filters

Our early work on dynamic queries [2, 6, 20] demonstrated dramatic performance improvements and high levels of user satisfaction. By allowing rapid, incremental and reversible changes to query parameters, often simply by dragging a slider, users were able to explore and gain feedback from displays in a few tenths of a second. For example, the Dynamic HomeFinder enabled users to adjust upper and lower bounds on home prices and see points of light on a map indicating available properties (Figure 1). This allowed users to immediately identify high or low cost communities, or find low cost homes in high-priced communities. Users could similarly adjust a slider to indicate number of bedrooms, and select toggles to indicate desire for garage, central air-conditioning, fireplace, etc.

Each of these query components (sliders, buttons, etc.) acted as a filter, reducing the number of items left in the result set. The effects were combined with simple AND logic, accounting for most naturally occurring queries. In situations where OR logic was required, users were usually quite satisfied, or actually preferred, generating a sequence of queries. This approach allowed users to see the size of the ORed components rather than merely the union of the result sets.

The work reported in this paper advances dynamic queries by demonstrating the efficacy of selection of items in alphanumeric lists with the Alphaslider [1, 12]. This query component allows users to select one item from a list of 10,000 or more, with a simple selection tool that takes little screen space, avoids use of the keyboard, and prevents typing errors.

Starfield Display

In our early work on dynamic queries the output was based on a naturally occurring spatial display. For example, the chemical table of elements was used with color highlighting of chemical names to indicate inclusion in the result set. In the Dynamic HomeFinder, points of light on a map of Washington, DC indicated properties that matched the query components. One step in the direction of generality was to build a version of the HomeFinder that had textual output as might be found in the tuples of a relational database display. As the query components were adjusted, the display remained stable, but when the user let go of the mouse button, the screen was refreshed with the correct result set.

To further support the widespread application of dynamic queries it seemed necessary to find other approaches to visual information display [4, 5, 7, 17]. Points of light are convenient because they are small yet highly visible, could be color coded, are selectable objects, and can be displayed rapidly. But if a natural map did not exist for an application, such as a set of documents, photos, songs, etc., could we create one that would be suitable? While we need to try further examples, our initial answer is affirmative. For many situations we have been able to create meaningful two-dimensional displays by selecting ordinal attributes of the items and use them as the axes. This starfield approach is a scatterplot with additional features to support selection and zooming. Our intuitions about what choices are most effective is still rough, but there is hope that we can formalize our decisions.

For example, in a database of documents, the year the document was written might be the horizontal axis while length in words might be the vertical axis. Large old documents might be at the upper left while short new documents might be at the lower right. Other attributes such as an author assigned importance value, number of co-authors, or number of references could also be used. In a database of people, the axes might be the age, number of years of education, salary, number of children, or other demographic variables.

Tight Coupling

The principle of tight coupling of interface components began to emerge in some direct manipulation graphic user interfaces. For example, if a user saves a document, the SAVE menu item becomes grayed out until a change is made. Tight coupling helps reveal the software state, and often constrains the user from making erroneous or useless actions.

A more complex example of tight coupling is the interrelationship between the text in a word processor, the position of the thumb in the scroll bar, and the page number displayed on the window border. Moving the thumb causes the text to scroll and the page number to be updated. We could write a logical proposition describing the relationship among these three display components. Such a statement would begin by indicating that when the top of the document is displayed, the thumb is at the top of the scroll bar and the page indicator is set at 1. Good program design would ensure the preservation of the display invariants. However, some word processors may fail to preserve this invariant when sections of the document are deleted or when the document is reformatted with a larger font. To compensate, some word processors may include a repaginate command, or update the thumb position only when it is moved. These errors violate the principle of tight coupling.

Tight coupling also applies to components of a query facility. In a well-designed facility, users should be able to see the impact of each selection while forming a query. For example, if a user specifies that they want films before 1935, then only certain actors or directors are selectable. This is to prevent users' from specifying null sets, e.g. films made before 1935 and directed by Francis Ford Coppola.

Another aspect of tight coupling is the linkage of *output-is-input* to support efficient use of screen space by eliminating the distinction between commands/queries/input and results/tables/output. In short, every output is also a candidate for input. This principle first emerged in our 1983 hypertext work [9] in which the notion of embedded menus replaced the earlier designs that had a paragraph of output followed by a menu to select further information. It seemed more logical to have highlighted words in the text and simply allow users to select those words with arrow keys, a mouse, or a touchscreen. The outputs-are-inputs principle reduced screen clutter by eliminating redundancy, and focused users' attention to a single location for gathering information and for applying an action to get more information.

This principle was applied in the chemical table of elements in which each element could be selected causing the sliders to be set to that element's values [2], in our health statistic map in which a state could be selected to retrieve its detailed data [13], and in the HomeFinder in which a point of light could be selected to retrieve detailed information or a photo, if it were available.

That database output can be used as input can be compared to functionality in spreadsheets where there is no such thing as input cells or output cells, or the Query by Example system [21] where input can be treated as output and vice versa. It has been referred to as a notion of Equal Opportunity [15]. In information retrieval systems this is useful as users can easily explore branches of a search and follow their associations as they come along -- associative database searching.

Tight coupling has several aspects:

- comprehensible and consistent affordances to guide users (highlighted words or areas, explicit handles, scrollbars, etc.).
- rapid, incremental, and reversible interactions among components.
- constraints on permissible operations to preserve *display invariants* (logical propositions relating the components, e.g. that the scroll bar thumb position constantly reflects the position in the document) and prevent errors.
- *continuous display* to always show the users some portion of the information space that they are exploring. They begin by seeing a typical result set or

item, which helps to orient them to what is possible in this information seeking environment. This seems more effective than starting with a blank screen or a form to fill in.
- *progressive refinement*, in which users can alter the parameters to get other results [18]. If the users see that there are too many items in the result set, they can reformulate their goal and seek a more restrictive value for one of the attributes.
- allow users to select *details on demand* [9]. This is the heart of hypermedia, but it applies to most designs. Instead of older query facilities which required alternation between query composition and result interpretation, our designs show results and invite further selections if details are needed. In the Dynamic HomeFinder, homes were shown as simple points of light until the user selected one of the points to get the details. This principle reduces clutter while the query is being shaped and allows users to get further information when they need it.

FILMFINDER DESIGN

To test these principles of visual information, we created a tool for exploring a film database, the FilmFinder. Watching a film is often a social activity so this tool was designed to encourage discussions and make the decision process easier for groups of viewers. Existing tools for learning about films include encyclopedias, such as *Leonard Maltin's Movie and Video Guide* [10]. They typically provide an alphabetic organization, with additional indexes, but these are difficult to use. Recently, computer-based encyclopedias such as Microsoft's Cinemania have appeared on the market. Although some of them employ novel approaches such as hypertext links they still do not provide users with an overview of the data. They employ a traditional approach to database queries with commands or form fill-in, and then provide a textual response in the form of scrolling lists. If the users are unhappy with the result, they compose a new query and wait for the new result. Progressive refinement can take dozens of steps and many minutes.

Before designing the tool, informal interviews were conducted with video store clerks and film aficionados. The FilmFinder [Color plate 1] tries to overcome search problems by applying dynamic queries, a starfield display, and tight coupling among components. Dynamic queries were applied by having a double box range selector to specify film length in minutes, by having buttons for ratings (G, PG, PG-13, R), large color coded buttons for film categories (drama, action, comedy, etc.), and our novel Alphasliders for film titles, actors, actresses, and directors.

The query result in the FilmFinder is continuously represented in a starfield display [Color plate 1]. The X-axis represents time and the Y-axis a measure of popularity. The FilmFinder allows users to zoom into a particular part of the time-popularity space [Color plate 2]. As users zoom in the colored spots representing films grow larger, giving the impression of flying in closer to the films. The labels on the axes are also automatically updated as zooming occurs. When fewer than 25 films are visible, their titles are automatically displayed.

To obtain more information about a particular element of the query results, users click on that element, getting desired details-on-demand [Color plate 3]. An information card which provides more information about attributes such as actors, actresses, director and language, is displayed. In a traditional retrieval system users would obtain more information by starting a new query. In the FilmFinder users can select highlighted attributes on the information card and thereby set the value of the corresponding Alphaslider to the value of that attribute. This forms the starting point for the next query and allows graceful and rapid exploration with no fear of error messages.

Tight coupling is strongly supported in the FilmFinder. When users select categories of movies using the category toggles, the starfield display and the query ranges of the Alphasliders are immediately updated [Color plate 2]. This effectively eliminates empty and invalid query results. The same is possible when users zoom into a particular part in the search space -- only those films that appear during that range of years and are in the particular popularity range will be part of the Alphaslider query range. The Alphasliders can even affect each other, selecting Ingmar Bergman on the Director slider would set the Actor slider to only operate over those actors who appear in Ingmar Bergman's movies. This interaction between the query widgets, and the possibility to use query results as input, creates a tightly coupled environment where information can be explored in a rapid, safe, and comprehensible manner.

FILMFINDER SCENARIO

Tools like the FilmFinder might be found in video stores, libraries, and homes - they might even come as a part of standard television sets. Imagine the Johnson family sitting down at their TV Saturday night to watch a movie that they all like. They turn on the TV and are presented with a FilmFinder's starfield visualization and a number of controls [Color plate 1]. All controls are operable by a wireless mouse and no keyboard is needed.

The family members don't have a specific film in mind, but they know that they want to see something popular and recent. After some discussion they agree on the categories for a search: drama, mystery, comedy, or an action film. To cut down the number of films further they select an actor that they all like, in this case Sean Connery. Observe in [Color plate 2] how the category toggles have been manipulated - and the Alphaslider indexes updated to contain appropriate values, the visualization has zoomed into the correct part of the information space, and Sean Connery has been selected with the Actor slider.

Now the number of films in the starfield has been cut down from about 2000 to 25 and the Johnsons decide to look further at *The Murder on the Orient Express*. They select it with their remote control and are presented with an information card [Color plate 3]. The description and image remind the Johnsons that they have already seen this film, so the information card can now become the tool to further refine their search.

Mr. Johnson sees Anthony Perkins' name and decides he wants to see a movie starring Anthony Perkins, while Mrs. Johnson wants to see a movie with Ingrid Bergman. To resolve the disagreement they select both actors in the information card, and the selection is reflected in the Alphaslider settings. When the information card is closed, the query result is updated and the Johnsons are presented with one movie with both Anthony Perkins and Ingrid Bergman which they decide to watch [Color plate 4].

FUTURE WORK

The dynamic queries approach has much to recommend it, but it must be extended to deal with larger databases, more varied kinds of information, and a greater range of query types. Current dynamic queries do not satisfy the demands of relational completeness, but they offer other features that depend on spatial output that are not available in existing relational databases. It appears productive to combine the strengths of both approaches.

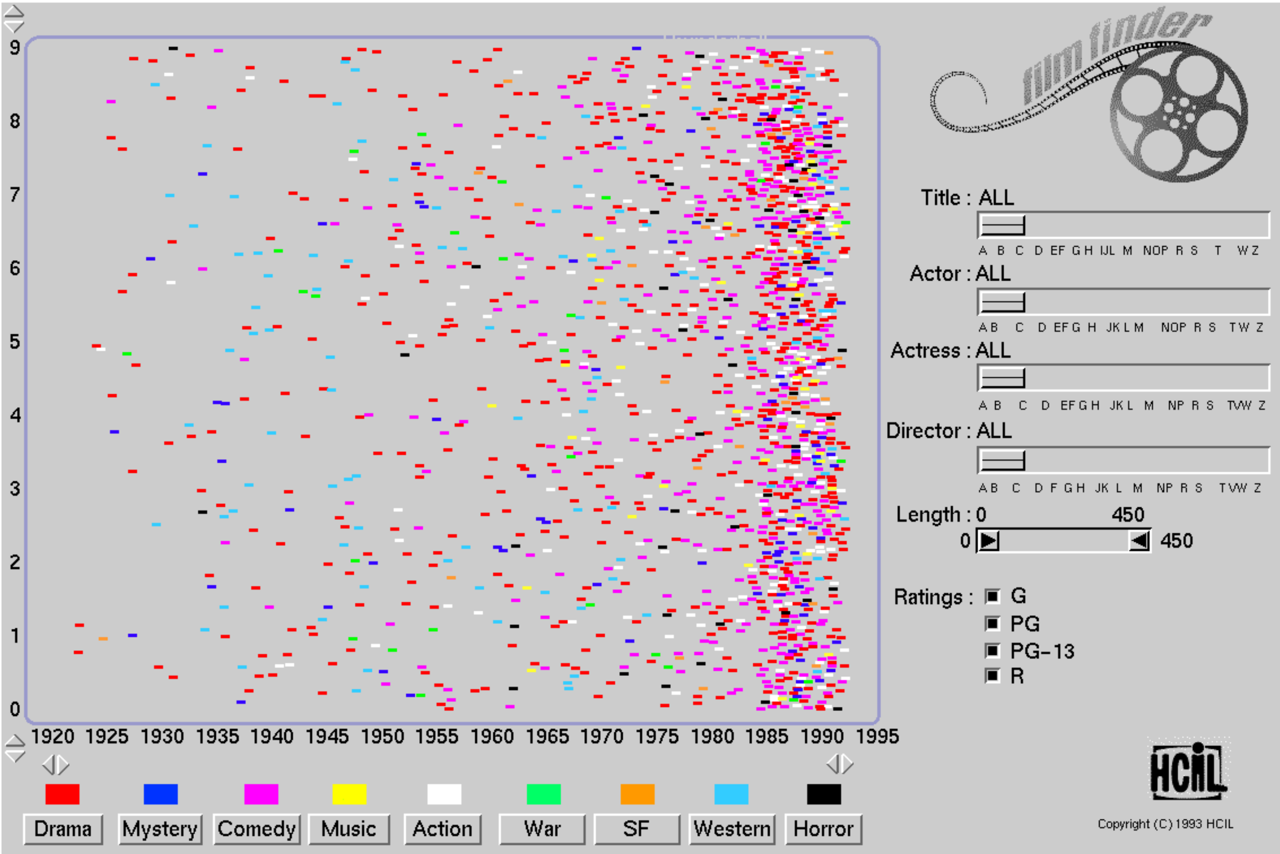
When searching films - as well as for other information - it would be desirable to incorporate fuzzy searching to find similar films. To include such functionality in the FilmFinder would probably be desirable - but first algorithms must be devised and more importantly, the issue of to what extent the mechanisms should be user controlled must be examined.

When browsing the information space by zooming, it is important that this is done smoothly so users get a feeling of flying through the data. New algorithms and data structures are necessary to support the smooth flying through the data. A natural extension would be to add a third dimension so that some films would appear closer than others.

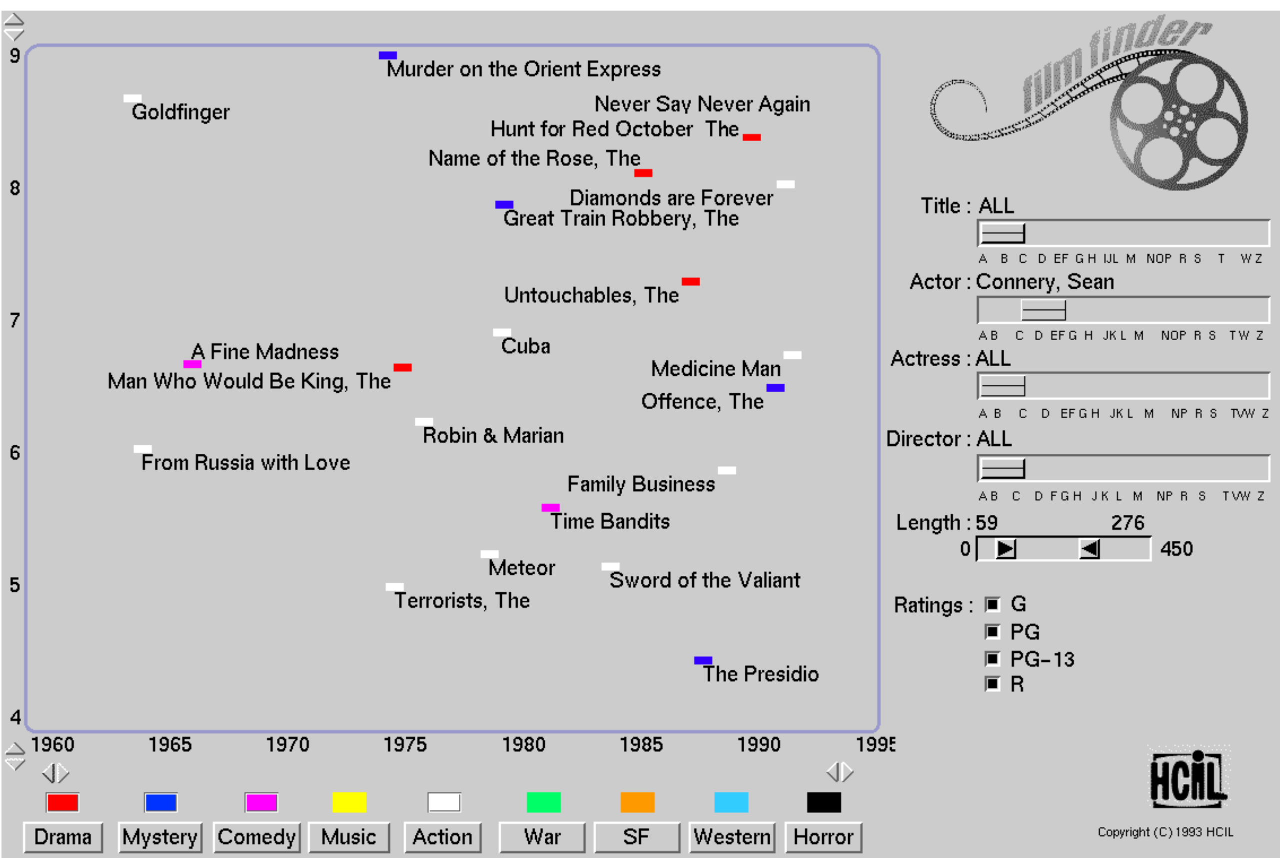
The tight coupling among query components in the FilmFinder was helpful - but there may be cases when such interrelationships not are desirable. Formal specification of the logical propositions for display invariants is a useful direction, because it could lead to proofs of correctness and advanced tools to build such interfaces rapidly.

REFERENCES

1. Ahlberg, C. and Shneiderman, B., 1993. The Alphaslider: A rapid and compact selector, *Proc. ACM CHI'94 Conference*.
2. Ahlberg, C., Williamson, C., and Shneiderman, B., 1992. Dynamic queries for information exploration: An implementation and evaluation, *Proc. ACM CHI'92 Conference*, 619-626.
3. Borgman, C. L., 1986. Why are online catalogs hard to use? Lessons learned from information-retrieval studies, *Journal of the American Society for Information Science 37*, 6, 387-400.
4. Buja, A., McDonald, J. A., Michalak, J., Stuetzle, W., 1991. Interactive data visualization using focusing and linking, *Proc. IEEE Visualization '91*, 156-163.
5. Egenhofer, M., 1990. Manipulating the graphical representation of query results in Geographic Information Systems, *1990 IEEE Workshop on Visual Languages*, IEEE Computer Society Press, Los Alamitos, CA, 119-124.
6. Eick, Steven, 1993. Data visualization sliders, AT&T Bell Laboratories Report, Napervillie, IL.
7. Feiner, S. and Beshers, C., 1990. Worlds within worlds: Metaphors for exploring n-dimensional virtual worlds, *Proc. User Interface Software and Technology '90*, ACM, New York, NY, 76-83.
8. Koved, L. and Shneiderman, B. 1986. Embedded menus: Selecting items in context, *Comm. of the ACM 29*, 4 (April 1986), 312-318.
9. Kreitzberg, Charles B., Details on Demand: Hypertext models for coping with information overload, in Dillon, Martin (Editor), *Interfaces for Information Retrieval and Online Systems,* Greenwood Press, New York, 1991, 169-176.
10. Maltin, L., 1993. *Leonard Maltin's Movie and Video Guide*, Penguin Books, New York.
11. Marchionini, G., 1993. *Information Seeking*, Cambridge Univ. Press, Cambridge, UK.
12. Osada, M., Liao, H., Shneiderman, B., Alphaslider: searching textual lists with sliders, *Proc. of the Ninth Annual Japanese Conf. on Human Interface*, Oct. 1993.
13. Plaisant, C., 1993. Dynamic queries on a health statistics atlas, Forthcoming Technical Report, Human-Computer Interaction Laboratory, University of Maryland, College Park, MD.
14. Robertson, G. G., Card, S. K., and Mackinlay, J. D., 1993. Information visualization using 3-D interactive animation, *Comm. of the ACM 36*, 4, 56-71.
15. Runciman, C. and Thimbleby, H., 1986. Equal opportunity interactive systems, *Int'l Journal of Man-Machine Studies 25* (4), 439-51.
16. Shneiderman, B., *Designing the User Interface: Strategies for Effective Human-Computer Interaction: Second Edition*, Addison-Wesley Publ. Co., Reading, MA (1992), 573 pages.
17. Singers, R., Endres, L., 1993, Metaphoric abstraction, the starfield and complex systems, in preparation, Johnson Controls, Milwaukee, WI.17. Welty, C., 1985. Correcting user errors in SQL, *Int'l Journal of Man-Machine Studies 22*, 463-477.
18. Welty, C., 1985. Correcting user errors in SQL, *Int' Journal of Man-Machine Studies 22,* 463-477.
19. Williams, M., 1984. What make RABBIT run? *Int'l Journal of Man-Machine Studies 21*, 333-352.
20. Williamson, C. and Shneiderman, B., 1992. The Dynamic HomeFinder: Evaluating dynamic queries in a real-estate information exploration system, *Proc. ACM SIGIR Conference*, 339-346.
21. Zloof M. Query-by-Example, *National Computer Conference,* AFIPS Press (1975), 431-437.



Ahlberg & Shneiderman, Color plate 1. The FilmFinder.



Ahlberg & Shneiderman, Color plate 2. Categories have been selected, the displayed is zoomed