# Image Browsers:
# Taxonomy, Guidelines, and Informal Specifications

Catherine Plaisant, David Carr and Ben Shneiderman

## 1. INTRODUCTION

The one-dimensional scroll bar has become a well-established fixture in all contemporary graphic user interfaces. While there is a common core of functionality, there are substantial differences in features, visual feedback, and operation. Users quickly accommodate to the differences and research on improved scroll bars is limited [Shneiderman 92] [Chimera 92].

When two-dimensional browsing is needed in paint or draw programs, geographic information systems, medical image systems, or advanced window managers, designers have merely used two one-dimensional scroll bars or they have made ad hoc designs for a two-dimensional scroll bar. However, the complexity of two-dimensional and higher dimensional browsing suggests that more careful analysis, design, and evaluation might lead to significant improvements.

We present a task taxonomy for image browsing, suggest design features and guidelines, assess existing strategies, and introduce an informal specification technique to describe the browsers.

### 1.1 Motivation for image browsing

A one-dimensional scroll bar is helpful in word processors when the document length is greater than one screen. Without a scroll bar users must rely on their memory of the current position and some numeric command for jumping within the document (e.g. 173,193p would print or display lines 173 through 193). The scroll bar enables users to move incrementally and by jumps within a document or large menu and provides feedback to indicate the current position of the screen within the document or menu. The visual feedback probably reduces memory and cognitive load for many tasks and simplifies planning.

Two-dimensional browsing is a requirement in painting or drawing programs when the image is larger than the screen. Building on user familiarity with one-dimensional scroll bars, many designers simply use two one-dimensional scroll bars to provide independent control over the horizontal and vertical directions (panning) (Figure 2). This is effective if users frequently move in a single direction by small (less that one screen) increments.

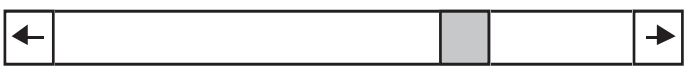


Figure 1: One dimensional scrollbar

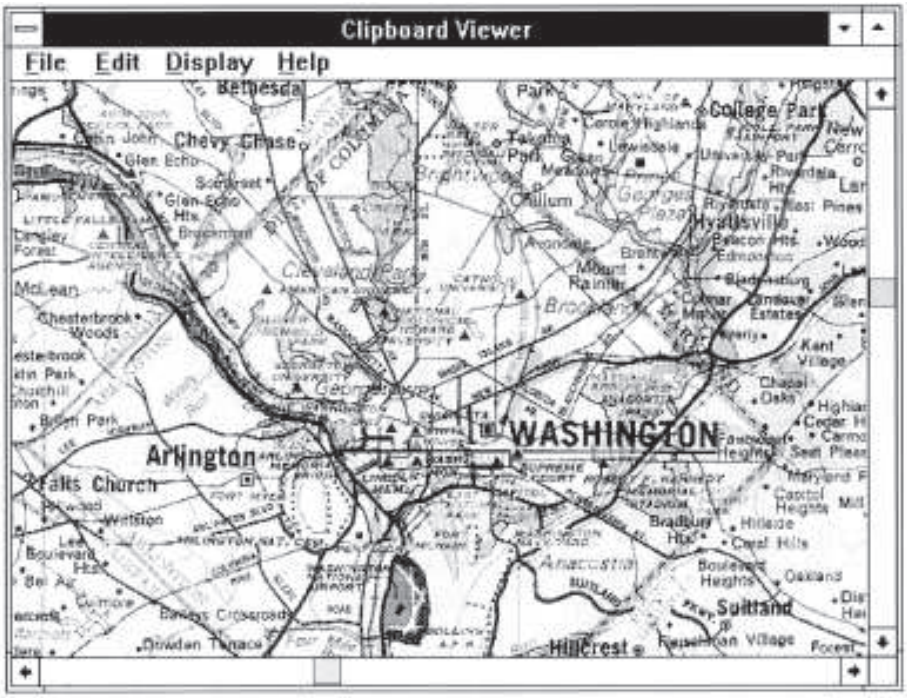



Figure 2: Detail only browser with two scrollbars

This solution becomes inadequate if the image is much larger than a screen, redisplay times are long, overviews are needed, zooming is desirable, diagonal panning is required, or multiple detail views are needed. These requirements occur in drawing programs and many other applications. In Geographic Information Systems (GIS), users may be browsing a world map and then seek detail views of a country, county, or city. The overview provided by the world map is a helpful - possibly necessary - feature to support the users' view of the details. The ratio of overview to detail view, the zooming factor, may be 1 to 10, 1 to 10,000, or even 1 to 10,000,000. GIS tasks may include following a river, border, or highway (diagonal panning), comparison of two harbors (multiple detail views), or simultaneous viewing of highways and population density maps (two related maps).

Medical images are increasingly stored and viewed within computer systems. Doctors may need to see a full spine X-ray and close-ups of vertebra pairs (an overview and multiple detail views) or to exam a tissue boundary on a microscopic slide (follow a feature on a detail view).

Managers for power distribution, telephone networks, computer systems, railroads or chemical plants, typically monitor a large system with an overview diagram and then focus on detail views when anomalies occur. Some problems are solved with local information only, but other problems require multiple detail views or an understanding of the "big picture" in the overview.

Our exploration of existing 2D browsers led us to identify a large number of features and a wide variety of tasks performed with the browsers. During the early stages of our exploration we found it difficult to even discuss our findings because of the lack of an adequate method to describe browser features. This led us to expand a sketching technique called DMsketch and being developed in our laboratory. DMsketch allows us to informally specify:

(1) the most significant graphical elements of the browsers,

(2) the interrelation between these elements, and

(3) the main possible user actions.

After describing examples of the most common browsers we present a taxonomy of 2D-browser tasks and techniques and guidelines to match the task to be accomplished with browser features and techniques.

### 1.2 Previous work

Papers can be found describing particular types of image browsers [e.g. Furnas 86; Sarkar and Brown 92] or describing applications including 2D browsers [Funke, Neal, and Paul 93] but very few studies have compared browser user interfaces or provided guidelines for their design or use. Overviews have been said to help people keep track of their location. Shneiderman [Shneiderman 92] has identified two dimensional browsers as a case of multiple window coordination and gives three examples. Windowing versus scrolling has been studied, mainly in the context of large text documents. Leung presents four techniques (two screens, split screen, bifocal display and fisheye view) with examples and some rough guidelines [Leung 89]. Beard [Beard and Walker 90] gives several examples of the use of 2D browsing in radiology, hypertext and other information systems. They describe an experiment which evaluates the usefulness of roaming and zooming techniques compared to simple scroll bars. The display of a miniature overview was again found helpful, and zooming and roaming were found superior to scrolling only. Schaffer et al. demonstrated the benefit of fisheye views over a crude zoom and replace interface for hierarchically clustered networks [Schaffer et al., 94]. In the context of the remote real time exploration of a pathology sample we have shown [Plaisant, Carr and Hasegawa 92] that an intermediate view (between the overview and the detailed view) becomes useful when the zooming factor is larger than 1:20. At higher ratios the overview alone could even become an hindrance to appropriate image exploration.

Related literature can found on the browsing of series of images (e.g. a radiology workstation will need to provide means to inspect dozens of images showing time series or different viewing angles) or of large image databases (showing thumbnails or performing graphical searches). The work we present here focuses on the tools to explore a selected image.

Geographical Information Systems often come to mind as typical image browser examples. Nevertheless, very little is done in commercially available systems to assist the browsing of maps. Those systems are mostly used to generate and print complex maps that are then studied on paper. Some systems offer complex macro facilities that allows users or developers to custom build the user interface of browsers but no guidelines or good sample designs are provided. As might be expected, our interviews of GIS users and trade show visits indicate that these facilities are rarely used. A common trend for GISs is to rely on the interface of widely available CAD/CAM systems which were designed for drawing tasks and not for map applications.

### 1.3 A few definitions

*Detail view*

A detail view (also called the local view) shows only a part of the image, usually magnified. In a world atlas application a view of Paris would be a detail view. The level of detail required depends on the task to be performed.

*Coordinated pair of views*

This pair of views (detail and overview) allows viewing on the detail view while keeping a sense of context on the overview. A hierarchy of views is usually provided by several coordinated pairs where the detail view of one pair becomes the overview of another pair.

*Zooming Factor (ZF):*

Level of magnification between 2 views. (ZF>=0)

*Global view*

The global view gives a view of the entire universe that could be explored. One purpose of the global view is to give a sense of what information will be in the image - and what is not. For example a world atlas global view might show a projected map of the earth, telling us that in fact this "world" doesn't include the moon or the galaxy. To get a real sense of how much information can be obtained from the image users need to see the global view AND a maximum-detail view. The maximum zooming factor is therefore related to the density of the image.

The global view is often used as an overview in a coordinated pair. But if the maximum zooming factor is too large, an intermediate view will have to be used as overview. For example because the zooming factor between the map of the world and the map of Paris is very large, one might use an intermediate map of Europe or France as an overview for the map of Paris (Figure 3 and 9).

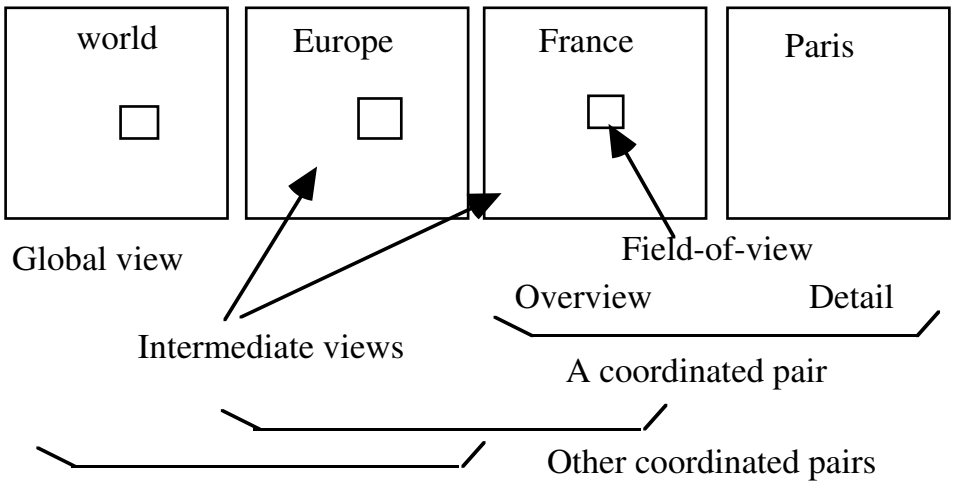



Figure 3: Definitions

*Intermediate views:*

Intermediate views are views showing the image between the global view and the smallest detail view.

*Field-of-view:*

In a coordinated pair the field-of-view indicates on the overview the location and shape of the coordinated detail view.

## 2. BROWSER SKETCH: INFORMAL SPECIFICATION OF IMAGE BROWSERS

This technique descends from both Hudson and Mohamed's graphical specification of layout constraints in the OPUS system [Hudson and Mohamed 90] and DMsketch developed by Plaisant and Shneiderman [Shneiderman 92].

Hudson and Mohamed introduced the idea of graphically representing a constraint on the layout of a user interface. They used an arrow to represent the presence of a constraint. The actual constraint is a hidden equation. The layout designer can view the equation by pointing at the constraint arrow. However, we believe that even if potentially more powerful equations can be developed, they do not convey meaning as clearly and quickly as a few specialized graphics. Also, this method does not have a way to specify that an area in one window will be viewed in greater detail in another.

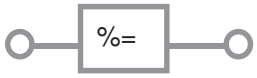
DMsketch (Direct Manipulation Sketch: "the printed equivalent of the 20-second demo") has been used as a technique to help designers exchange and record ideas more quickly and clearly than a formal specification language. DMsketch originally included icons for single clicking, double clicking, dragging, and so on. But this would be too low-level for our purposes so we extended DMsketch to show the major differentiating characteristics of the browsers. We are not so much concerned with the details of interface operation at the keystroke level as we are concerned with the relationships between windows.

### 2.1 The conventions used

We introduce our notation by defining a few primitives. Throughout this paper we will add new primitives and define composite objects as they become necessary to describe browsers.

Movement constraint operator

The movement constraint operator specifies that the object at its tail is movable. If the arrow is horizontal, it is movable in the horizontal direction. If vertical, then it is vertically moveable. The movement of the tail object is limited to be within the context of the object at the head of the arrow. An object without a movement constraint operator attached is by default not moveable.

Proportional size constraint operator 


The second primitive is the proportional size constraint. This operator joins two objects by its circle end points. The proportional size constraint forces the two joined objects to maintain the same

relative size. For example, it could join to a line whose maximum length is four to a line whose maximum length is two. If the larger line is shortened to two, then the smaller line would automatically be shortened to one. This constraint operates both ways. So, changing the shorter line would change the longer as well. This operator is not confined to lines. Two dimensional objects and movement constraint operators may also be joined.

We characterize the existence of such bi-directional links between elements of the user interface by the concept of tight coupling. This concept was first described in [Ahlberg and Shneiderman 94], in the context of an information seeking interface.

Fields-of-view

A B C D E F

The field-of-view encloses an area of an image and is displayed on the window which contains that image. It defines a clipping rectangle for the image in its underlying (or world coordinate) representation. This means that images enclosed by a field-of-view do not automatically become coarser as they are magnified. This would only happen after the maximum resolution of the underlying representation was reached. The contents of the field-of-view are projected into an new window. This window is identified by an arrow pointing from the source field-of-view to the destination window.

Shown above are a few variations of the field-of-view. The leftmost image (A) represents a generic field-of-view. The second field-of-view (B) shows that it was constructed by defining two points that represent the corners of the field-of-view. This rectangle is typically defined by a “mouse down, drag, mouse up” operation. The third field-of-view (C) is similar to the second, except that the first point defines the center instead of one corner. Field-of-view (D) is always the same size and is defined by one point. The image for field-of-view (E) shows that there are several different magnifications available. The last field-of-view (F) has a shape which matches the destination window.

Fitted projection

This symbol is used to show that the image within the field-of-view is projected to a window pointed at by the arrow.

## 2.2 Example of composite object

Standard coordination for fixed window browsers


In order to simplify the specification, one can define composite objects, give them their own symbol, and use it in subsequent specifications. For our first example of the notation, we will define a composite object. This object specifies a standard coordination between an overview and detail view of fixed sizes (Figure 4). At this point, we add the convention that unless otherwise specified all objects presented will be of fixed size. The left window is the source view and contains some image. A field-of-view in the source view can be moved both horizontally and vertically. The image enclosed by the field-of-view is projected on a second window which has scroll bars. The horizontal and vertical scroll bars are linked to the field-of-view by movement constraints . Thus, moving the field-of-view will not only change the image in the second window, but will change the scroll bar indicator position as well. In addition, moving a scroll bar will change the position of the field-of-view and modify the projection displayed in the second window. This particular coordination will be used frequently when specifying browsers. It is our recommended standard coordination for fixed window browsers and it has its own symbol (shown above). Note that if the windows were resizeable the shape of the field of view would have to coupled to the shape of the detail view.

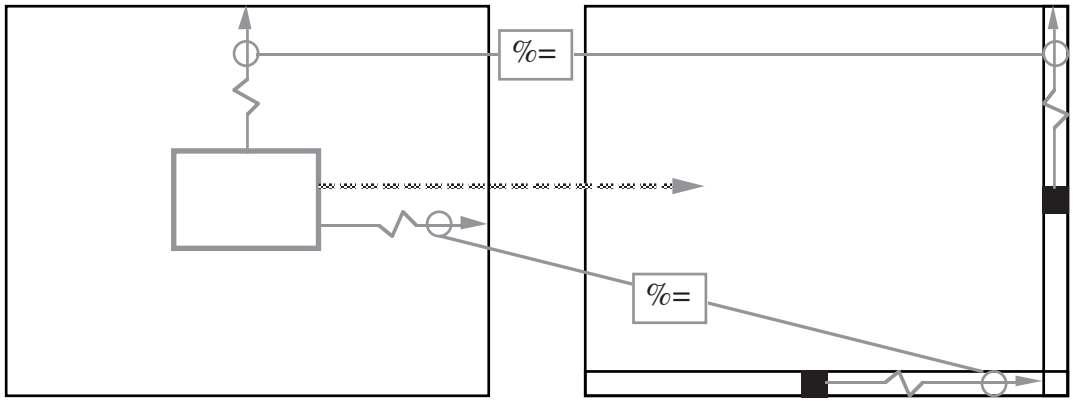



Figure 4: Specification of our standard coordination between two fixed size windows.

## 2.3 Commands

At this point the specification provides only a rudimentary way of describing commands by showing the “before” and “after” state of the interface (see Figure 6 for a Zoom command)

## 3. A MULTITUDE OF BROWSERS

A review of existing systems reveals the great diversity in designs for two-dimensional browsing. We introduce some of the classic techniques and show variations by using Browser Sketch.

### 3.1 Detail only browser

A single window is shown, which can be panned both horizontally and vertically over the detailed view of the image (Figure 2 and 5). This is the most common method in systems such as X windows, Microsoft Windows, and the Macintosh user interface. It is an easy to implement technique, but, it is only satisfying when the zooming factor is relatively small or seeing the global view is unnecessary. For example with a ZF = 2, a quarter of the image can be seen at once; therefore, not much navigation is required to see everything. However, with large zooming factors navigation is difficult. Imagine having a map of Europe at street level detail. If you are trying to find your way around Paris, it might take some time to realize that the view is actually one of Brussels.

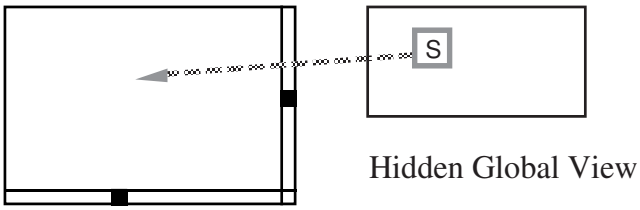



Figure 5: Single view browser.

### 3.2 One window with zoom and replace

A global view is presented so that the entire image can be seen. The user then marks a rectangular area and that area is magnified and replaces the original image. It is easy to implement and uses the screen space efficiently . It allows users to work on the detail view with all the screen space, handling navigation separately, but the context switch can be disorienting. This method is common to many CAD/CAM and GIS systems.

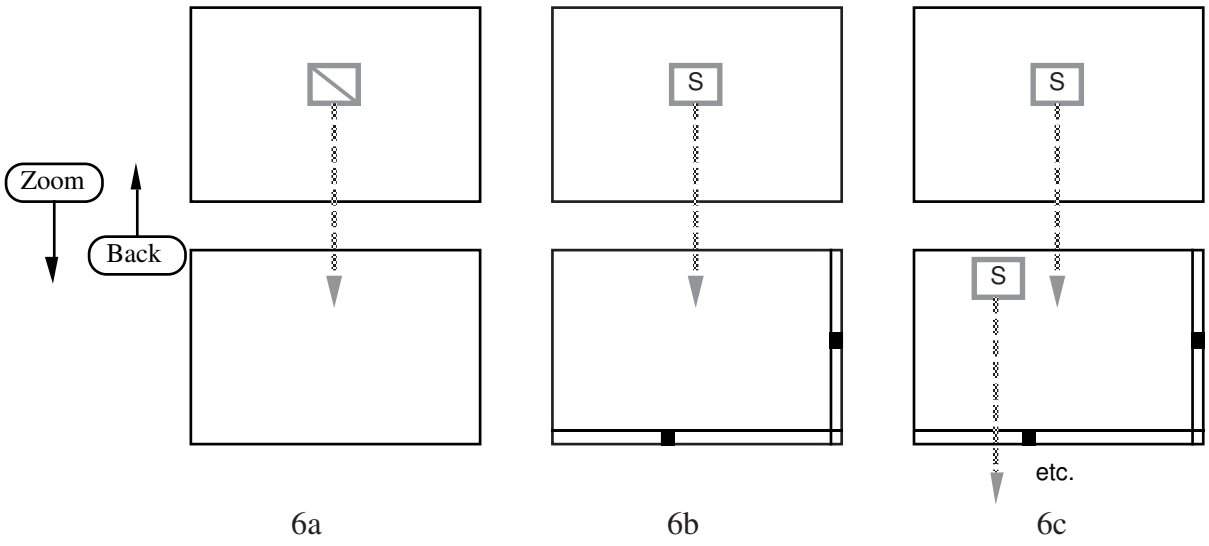



Figure 6: Three variations of the zoom-and-replace browser.

The leftmost browser 6a is the simplest zoom-and-replace browser. It has a major drawback in that if the zoomed view is not quite what the user want then the user must go back and zoom again to adjust the view. The middle browser 6b solves this problem by allowing the user to scroll the detailed view. The third browser 6c adds additional levels of magnification.

Of course the 2 first methods proposed can already be combined (global view first, zoom and replace then scroll (i.e. Pagemaker); or scroll first, with option to zoom out to global view or in to more details (i.e. MacPaint 2.0).

### 3.3 Single coordinated pair (detail and overview)

Many 2D browsers combine displays of the overview and a local magnified view. They are variations on our standard coordinated pair specified in Figure 4. The most common screen layout reserves a small part of the screen for the global view (7a), but both windows can be of equal size (7b) or the overview can occupy most of the screen while a small window gives details.

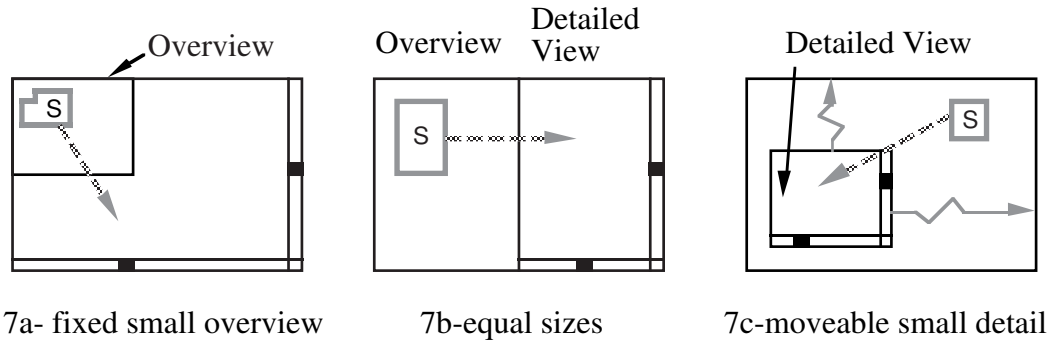



Figure 7: Three examples of single coordinated pairs

### 3.4 Tiled multilevel browser

Figure 8 and 9 show a three level browser with global, intermediate, and detailed views. The field-of-view for the global view defines the intermediate view. Moving the global field-of-view or scrolling the intermediate view updates both the intermediate view and the global field-of-view location. This is the standard coordination defined in figure 6. Similarly, the intermediate view and the detailed view use a standard coordination.

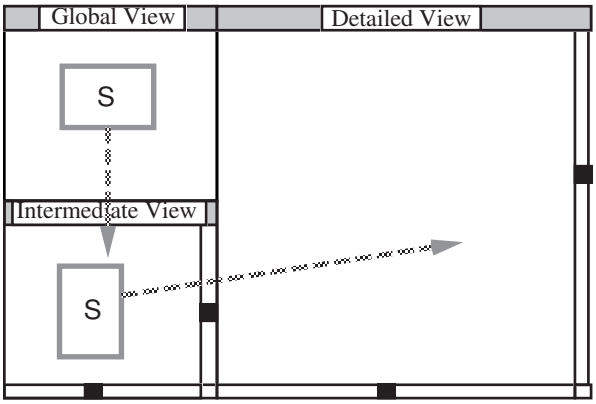



Figure 8: Specification of a three level browser with global, intermediate, and detailed views.

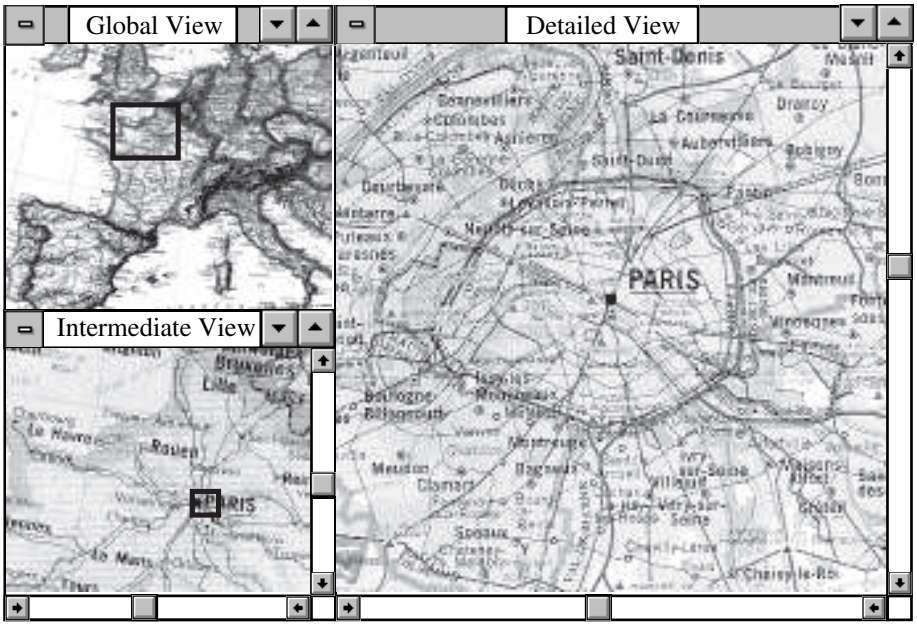



Figure 9: Example of three level browser with global, intermediate, and detailed views.

### 3.5 Free zoom and multiple overlap

This is the typical design chosen by application running on fast platforms with large screens (figure 10). Users are free (but required) to specify, move, reshape and delete every window as they want. Any side by side comparison is possible.

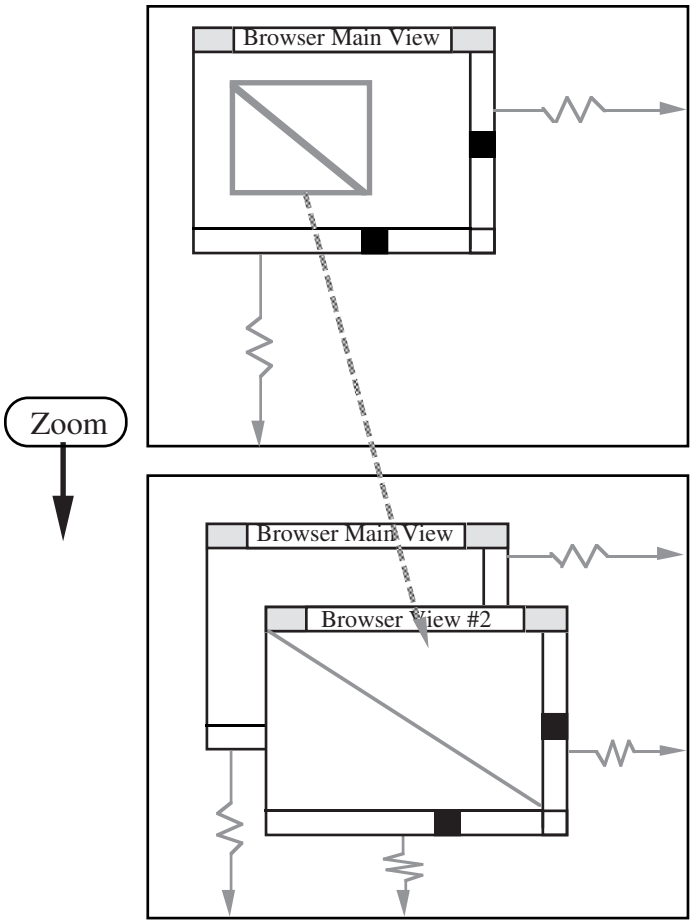



Figure 10: Specification of zooming in a overlapped window browser.

The user marks an area in a current map view (top frame) and marks the boundaries for a new window (bottom frame). The system then creates the window and projects the marked area into the new window which overlaps the source window. Both windows are linked to an undisplayed global view (not shown). Since there is no coordination between the views, the user has two independent browsers at different magnification.

This design has more flexibility but more specification and window management is required. Managing the display takes a significant amount of time and windows are constantly obscuring each other. All actions are initiated from the overview of the whole image which is always presented first.

### 3.6 Bifocal view browser

A variant of the classic overview/detail browser is called the bifocal browser [Spence 82]. It uses a magnifying glass metaphor. A zoomed image is placed on top of where the magnified object is located, therefore always covering the neighboring objects.

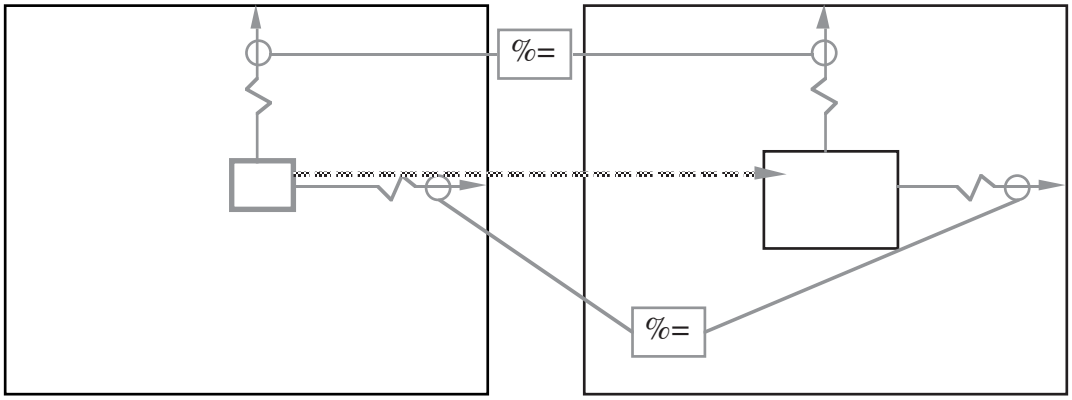
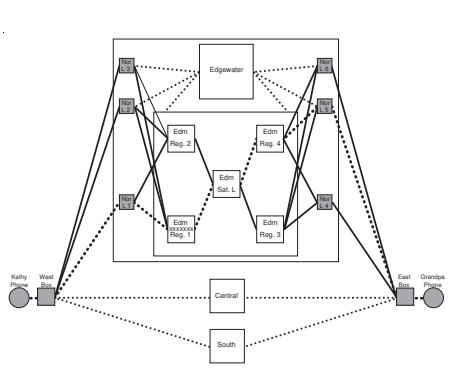



Figure 11: Bifocal browser (or magnifying glass browser)

### 3.7 Fisheye view

An interesting extension of the bifocal view is the fisheye view [Sarkar and Brown 92]. The image is distorted so that the center of interest is displayed at high magnification while the image is progressively compressed at other places (Figure 12). It uses a single view showing a distorted global view so no zooming or scrolling is required. Distortion can be severe, especially with large images. Image browsing is unecessary but users still have to specify the focus point(s) and magnification(s).



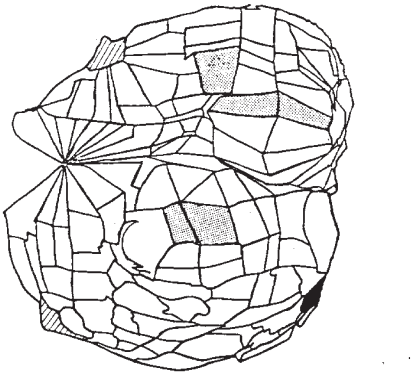



Figure 12a (from Schaffer et al. 94) Figure 12b

Figure 12: Examples of fisheye views for network diagrams and geographical information.

## 4. TASK TAXONOMY

We identified five classes of tasks to be accomplished with image browsers. Applications are often required to provide for different types of tasks but it is usually possible to identify in each application the most important task which is either performed repetitively or gets first priority because of safety requirements.

*Image generation (drawing, painting, scanning...)*

Users are drawing or painting a large image or diagram. The attention is on a small part of the image but there is often the need to step back to look at the entire image. With a painting program a painter might concentrate on the drawing of a face then look back at the whole view of a scene. With a CAD/CAM program a boat designer might spend an hour drawing the bow of a boat then check the overall shape of the hull. Here units and sizes are often important. When a large document is automatically digitized by a scanner, progress is shown on a view of the whole document, but the refining work will be done in the few areas of the image that need retouching. For such image generation tasks, a overview is important but most of the time is spent at a detail level. Users tend to be expert users.

*Open ended exploration*

With a surrogate travel program, a museum patron or a tourist can explore a remote city by navigating a map and accessing information on the local attractions. An adventure game player has to quickly move around the imaginary space to become familiar with it. The space to explore is unknown, so it's easy to get lost. The overview of the space being explored is not always complete or even available since it is explored for the first time. The navigation needs to be fast and the user interface quickly mastered.

*Diagnostic*

A special case of exploration is the diagnostic task. For example, with a pathology workstation, pathologists can explore a digitized sample of tissue at low or high resolution. Similarly a VLSI circuit specialist can explore a magnified view of a circuit. Most of the time is spent panning the image, looking for patterns, therefore panning speed is crucial, so is coverage as skipping part of the image can result in the wrong diagnosis. But a complete automatic scan can also lead to boredom and errors. Important locations can be saved for later review. Several browsers might be needed to compared cases.

*Navigation*

Users are navigating a more or less known environment. The typical question in mind is "how do I get there?" A Geographical Information System onboard a vehicle can be used to direct a package delivery truck to its destination, or a tourist to a museum. A global view is necessary to show the current position, provide context and point at destinations. Then the relevant information is presented at the minimum magnification level necessary to view the route. Zooming and panning occur

only occasionally.

*Monitoring*

Here users have to "keep an eye on everything" and always have information status on the entire system that they are monitoring. Examples include the management of a large network, the central monitoring of the security or temperature of a large set of buildings, or the monitoring of a production plant. When problems occur users have to be able to allocate some attention to local aspects while still watching the overview. Multiple views can be associated with a given problem that should be globally saved or retrieved since the number of windows can become very large. Window management is an important issue: an overlapping window can hide important changes in another window.

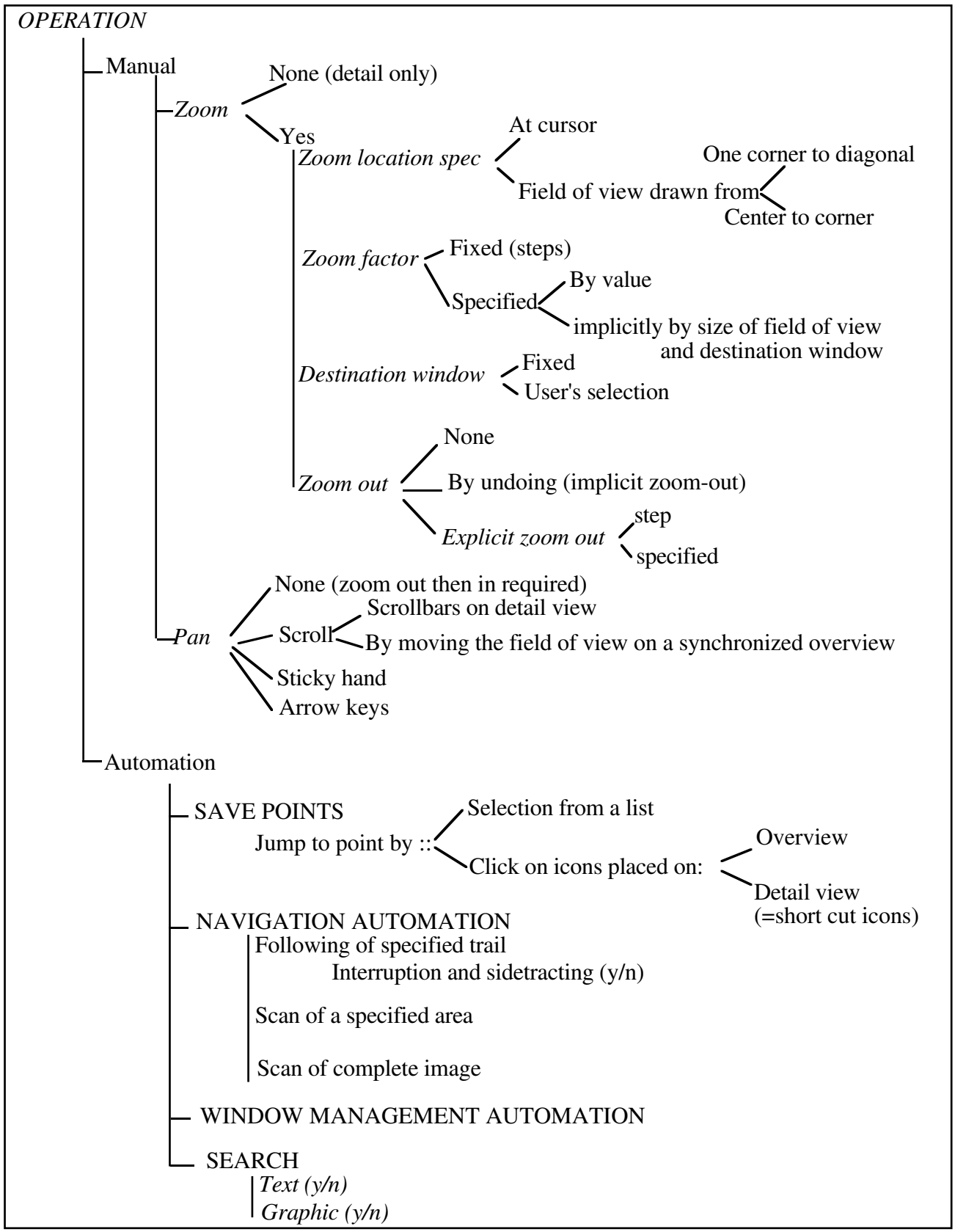
## 5. IMAGE BROWSERS TAXONOMY

We identified several points of comparison between image browsers and prepared a taxonomy of image browsers summarized in Figures 13 and 14. We classified all points of comparison into presentation aspects and operation aspects. In turn the presentation aspects were separated between static and dynamic, and the operation aspects between the manual and automated operations. Then for each aspect we list and classify all the techniques or features we found in image browsers.

### 5.1 Presentation (See Figure 13)

Single view browser: Browsers can dedicate all the screen space to a single view. The single view browsers are very efficient for limited panning. They are the most commonly used browsers when display space is scarce. They are appropriate when the task requires users to concentrate on only a part of the image that can be made to fit the screen, but inappropriate when users have to compare several distant parts of the image (using a fisheye with multiple foci can help in some cases). We identified three variations of the single view browser, the detail only browser, the zoom-and-replace browser, and the fisheye.

- The detail only browser does not support any zooming in or out, but only panning. As the Operations diagram shows (Figure 13), panning can be done by using scroll bars, sticky hands or arrow keys. This type of browser is the default provided by most windowing systems if the image to be viewed is larger than the window used for viewing. But it only seems to work well when the entire image is not much larger than the view. For example, it is quite acceptable to pan through an image which is four times the size of view. This browser is common for image generation since most of the work is done at the detail level. It is not appropriate for monitoring as some changes can remain unseen.
- As the size difference increases navigation becomes more difficult and the zoom-and-replace browser becomes more appropriate. Some zoom-and-replace browsers do not offer panning, which can be annoying since users have to zoom out and then in to make slight adjustments to the detail view. Another typical mistake for zoom-and-replace browsers that allow panning is to




13: Browser taxonomy for presentation aspects

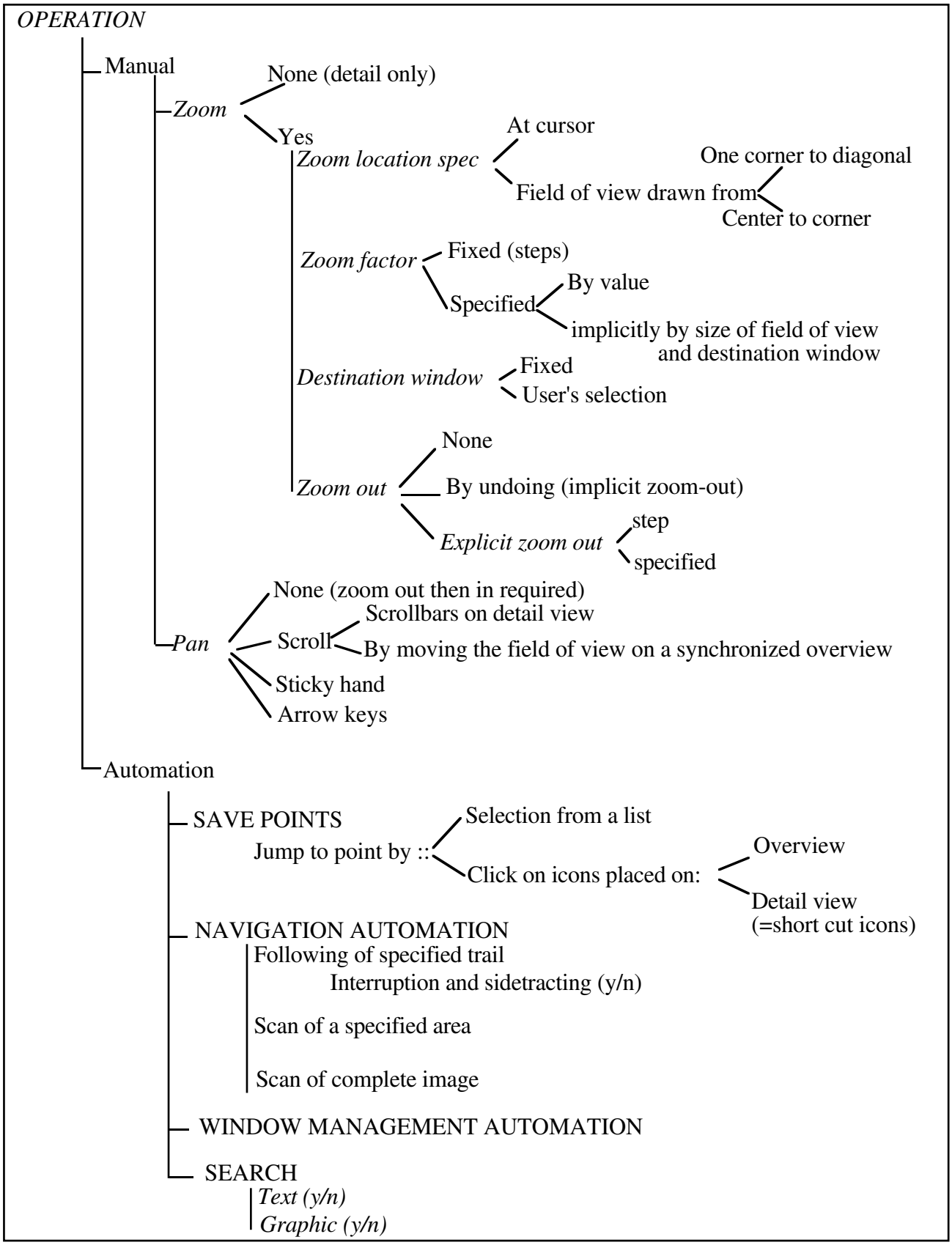



Figure 14: Browser taxonomy for operation aspects

forget to update the cursor position in the hidden overview during the panning of the detail view (which causes confusion when zooming out). Zoom-and-replace browsers make good image generation browsers and appropriate diagnostic tools if display update speed is sufficient. The numerous dynamic presentation aspects of the zoom-and-replace browser make it a large family of browsers. Some control of the zooming factor should be provided.

- Fisheye views give detail and context in a single view but, the fisheye browser severely distorts the image and requires constant reorientation. The distortion is a severe problem for applications where size and geometry are important (e.g. geographic information or CAD/CAM drawing). Fisheye views seem more appropriate for viewing abstract representations such as network diagrams which can be tailored for a user or a task but views do not change constantly for a given user or task. The transformations are complex and computationally demanding [Sarkar and Brown 92]. Schaffer et al. demonstrated the benefit of fisheye views for hierarchically clustered networks [Schaffer et al., 94]. The fisheye browser they developed was found to be superior to a zoom-and-replace browser. We were disturbed by the fact that the parameters of the zoom-and-replace interface were sub-optimum (e. g. too large of a zooming factor) but this experiment provides very valuable insights in the design of those two browsers. Designers should remember that fisheye views can be inappropriate when fidelity to standard layout is important. For example the map shown in Figure 12b was found to be useless for epidemiology. It is a fisheye view of San Francisco (size proportional to the 1980 white male population). Such cartograms were rejected by epidemiologists because it made it impossible to compare with the many other maps they are trained to memorize (e.g. maps of diseases.)

Three techniques can be used to modify the fisheye view: the image can be graphically distorted, or filtered (to remove unwanted objects away from the focus) or abstracted (replacing blocks by symbols). Fisheye views resemble domain specific layout programs allowing interactively generated custom layouts.

Dynamic aspects: An important presentation aspect which can be used when comparing browsers is the smoothness of the screen update when the image is panned or zoomed, and the nature of the update when it is zoomed.

*Quality of the update (slow/fast - continuous/jumpy)* A fast and smooth continuous image update (e.g. the Erdas Imagine GIS system) makes the navigation and exploration of images natural and simple even over relatively long distances. It allows users to concentrate on their tasks rather than managing a navigation tool. At one end of the spectrum are the “fly-over” interfaces only possible with hardware having sufficient throughput. At the other end slow and jumpy updates can be disorienting if not dizzying, in which case overviews become mandatory. But the smooth scrolling plus rapid and continuous zooming remains the secret of success for single view browsers.

*Nature of the update:(zoom /explode /distort- Zooming factor) :* Another aspect is whether or not a

zoomed image is similar to its representation in the field-of-view (or the unzoomed image). When an area is zoomed, it can be simply expanded (similar to a camera zoom) or it can be "exploded" to reveal the internal structure which was not apparent in the overview or unzoomed view (e.g. zooming on a network node represented as a plain rectangle might bring the diagram of the internal structure of that node). This "explosion" technique is regularly used for hierarchical or hierarchically clustered datasets. It simplifies the drawing of the overviews but can cause disorientation since the image is always changing[Schaffer et al. 94].

When an "explode" zoom is used designers need to consider what subset of information appears on the overview. This is especially important for monitoring applications where alarms should be visible on the overview. In addition to expansion and explosion, the zoomed image can also be distorted, as is the case in fisheye browsers. Here again techniques can be combined: when an object is selected for zooming the image can be expanded to only show the neighborhood of the object and the object itself exploded to show its interior structure.

*Zooming factor:* When the image is expanded the zooming factor is the level of magnification between two views. As shown in the operation aspects diagram the zooming factor can be fixed or specified. When the factor is fixed it is the designer's role to choose an appropriate factor or set of factor(s). This is a difficult task as a compromise must be made between speed of access to details and the preservation of context information at each level. No validated guidelines exist and designers have to rely on usability testing with real users and tasks to adjust the zooming factors to appropriate levels, and use intermediate views when necessary [Plaisant, 92].

Multiple view browsers: Displaying several views becomes a necessity when viewing details and context simultaneously is important, the distortion of fisheye view is not appropriate, parallel viewing is required for comparison, or when the display speed is insufficient to allow continuous zooming and panning. Designers too often rely on window managers' capabilities. Typically window managers handle the free overlapping and resizing of the windows. This strategy is easy to implement and permits any type of task to be performed, but we believe that it is more difficult to use. It has been shown [Bly and Rosenberg 86; Davis, Bury, and Darnell 85] that managing the overlapping windows can take considerable effort and time for the users. We believe that providing an automatic window management strategy should limit the need to move and resize windows incessantly. Many simple strategies (like the classic overview / detail pair ) are possible and researchers are investigating more complex strategies that are likely to be closely linked to specific tasks [Funke et al. 93]. The more elaborate strategies are likely to be task dependent and research is needed to provide guidelines for designers and to develop tools to specify and customize those window management strategies.

The standard overview and detail view coordinated pair (Figure 4) is easy to implement and answers many users' needs. It can be recommended for all tasks. The field-of-view should be the same shape as the window it represents (e.g. square for figure 7b, boot-shaped for 7a). The systems using this

technique can be compared by the screen space ratio between overview and detail view (SSROD). This ratio should be a function of the task to be performed. For example drawing or open-ended exploration requires a large detail view, while monitoring requires a large overview to guarantee that changes will be noticed. For navigation, detail view and overview are of similar importance.

In addition to the relative size of the overview and detail view, systems vary in the way the views are laid-out . Tiling the view windows frees the user from managing the views. Overlapping views gives more flexibility but forces the user to do more management. There are examples of systems where overlapping windows cannot be moved and block access to part of the overlapped image (e.g. an early version of Publishers-Paintbrush, and Figure 7a). Of course this should be avoided and small windows overlapping the larger view must be movable.

As discussed in the previous section the choice of the zooming factor is delicate. For coordinated pairs our experience suggests that magnification between an overview an a detail view should be less than 20. Once the zooming factor between screens gets to be 20:1 users have difficulty using the overview for navigation [Plaisant et al. 92]. Intermediate views should be added as necessary.

Hybrid browsers are common. For example a global view can be provided along side a second window which functions as a zoom-and-replace, the field-of-view of the global view always providing feedback about the size and location of the zoomed area. Another nice hybrid is the free overlapping of multiple chained pairs of coordinated views.

Coordination: Our recommended coordinated pair (Figure 4) includes a symmetric position constraint between field-of-view and detail view scroll bars. Moving the field-of-view updates the detail view. Similarly, panning the detail view or modifying it should update the overview and the location of the field-of-view. This is an example of bi-directional coordination (tight coupling) between two views. Browsers can be compared by the degree of coordination (or tight coupling) between view. The coordination can be non existent (no field-of-view), unidirectional (moving the field-of-view updates the detail view) or bi-directional (the field-of-view is updated when the detail view is scrolled or modified). We believe that there are many opportunities for beneficial coordination.

Global view: This view of the entire image is important in many tasks. For diagnostic tasks users might be required to explore the entire image (e.g. slides of possibly cancerous tissue sample). When presented with the pan-able detail view of a microscopic slide a pathologist may entirely miss a part of the image if the sample is made of two distinct parts. During a monitoring task alarms may be ignored if the global view is not visible at all times. In the world atlas example (Figure 3) children may be desperately looking for non-existing information about other planets if no global view gives them the scope of the atlas. Just as a table of contents is a requirement for a printed book, the global view is a requirement for a large image browser.

The global view shows the entire information space and allows quick access to any part. This

overview can be made simple and attractive for novice users or public access information systems. For experts the global view should be as detailed as permitted by the display. Dense global views provide experts with direct access to details that would otherwise require several zooming operations (even if these global views appear unreadable to others!).

### 5.2 Operation

In addition to classifying browsers by presentation, they can be classified by the type of operations they require and permit. We split operations into manual and automatic.

Manual operations: Browsers support two principal manual operations, zooming and panning. If zooming is needed the zoom location can be specified by the cursor location or by the drawing of the field-of-view on the overview. A fixed size rectangle is used for a fixed zooming factor or a user controlled rectangle for variable zooming. The new detail view can be placed either by the user or by the system. Zooming out can be implicit by undo or explicit by step or by zoom factor specification. The designer has to find the appropriate compromise between the danger of complexity and the advantages of flexibility. Browsers intended for public access or occasional users will benefit from simple designs (e.g. zoom at cursor locator and fixed zooming factor) while expert users will demand more control over the browsing. Implementing all possibilities in a system is likely to be unrealistic and designers should carefully study the tasks to be accomplished. For example if size is important or if measurements are to be done on the image, specifying the zooming factor by its value (e.g. 200%) is more important than giving control of the field-of-view.

We observed four different panning implementations. The most common is vertical and horizontal scroll bars. Another way is with a "sticky hand" which grabs the picture when the mouse button is pressed (first used in MacPaint). The picture then follows the cursor until the mouse button is released. The "sticky hand" metaphor is only appropriate when a real time update of the image is possible. Arrow keys are another method used for panning. Finally when an overview is present, panning of the detail view can be accomplished by moving the field-of-view indicator in the overview. Panning and zoom can be readjusted simultaneously by redrawing the field-of-view or adjusting its size, placement and even aspect ratio. Some general panning should always be available. In some rare cases detail view panning might have to be disabled to avoid confusion between similar parts of an image (e.g. when looking closely at a single vertebrae of a spine X-ray, panning to others vertebrae may bring confusion as to which vertebrae is currently on the screen. It might be more appropriate to only allow panning by moving the field of view.)

Automation : If browsing involves too many actions it becomes difficult for users to concentrate on their task. Possible beneficial automations include: saving points for rapid return, automatic navigation, automatic window management, and text or feature search.

*Saved points:* Similar to setting bookmarks in text, marking points on an image can speed up image browsing. Locations of interest can be saved to allow rapid return to those saved points. Eventually,

written or spoken comments can be saved with the location. This process of saving and retrieving points can greatly speedup navigation and diagnostic tasks (for second opinion).

*Navigation automation:* Direct manipulation techniques can be used to automate some of the navigation. An area can be marked on the overview to be systematically explored. A trail can be drawn on an overview and followed automatically, with the possibility to stop, explore locally and resume the trail. Those techniques can be useful for diagnostics or even for simple exploration tasks. The areas already explored can also be shown on the overview to verify that all important parts of the image have been explored. Macro commands can be offered. For example, if a series of similar images have to be analyzed, users can mark the start point of a typical exploration sequence which is then executed with a single command.

*Window management automation:* When multiple views are used the placement of the windows can be automated. Fixed positioning of windows is of course an extreme example of such automation. The coordination of the field-of-view and the detail view is another example. But more elaborate strategies can affect the sizing and placement of windows. For example windows can be iconized when unused for a certain time, or resized according to their estimated level of interest [Funke et al., 93]. Synchronized windows can help compare multiple images which can be panned simultaneously and then closed simultaneously [Shneiderman, 92].

*Image search:* The automatic identification of features in images is a growing field of interest based on the large body of work in computer vision on feature extraction and on similarity measures. A lot of work has recently been devoted to image retrieval, but similar techniques could be used to navigate within a large single image. For example, a map can be searched for switching yards, a spine X-ray for the location of each vertebrae, etc. Of course, a simpler search can be done on the text found in the map. Such feature extraction might allow the browser to be adapted to the task using content information, multiple detail views can be created automatically or panning speed can be adjusted according to the presence of features of interest (e.g. the panning of a county map would be tailored to slow down when a switching yard is visible on the screen, or to jump from yard to yard).

As these examples illustrate, many automations are possible. Research is still needed to determine the benefits of such automations (or even in some cases to prototype and implement them). In general those automations are likely to be very task dependent and only found in specialized browsers. Another challenge is to devise techniques allowing users to specify the needed automations. This topics borders the more general topic of programming in the user interface.

## 6. TOWARD 3D-BROWSERS

So far 3-dimensional spaces generated and browsed on computer screens tend to be either small spaces (limited need for navigation), exploratory adventure games (being lost is a feature) or entirely based on some pseudo-natural navigating interfaces (going somewhere or bringing objects, fly through). Applications are not so much for drawing or constructing, but for viewing what was done with other 2D tools. Many 3D applications are really 2D navigation (on the ground) or 2D with

layers (in a building). There are more possible manual operations than just zoom and pan. More complex systems are like flight simulators which use the traditional flight instruments and tools for navigation.

The basic browser is a zoom-and-replace browser (also resembling the fisheye view with perspective). 3D browsing is based on the "natural" navigation skills of the users (approaching, turning around). Overviews are sometimes provided, either as classic 2D overviews (e.g., a user in virtual reality can grab a virtual street map), or in 3D. For the applications which require the overview to be complete and always visible (e.g. monitoring) 3D overviews have to be automaticaly rotated to periodically expose all faces or flattened into multiple 2D slices.

In comparison 2D image browsers provide much more functionalities than the traditional "natural" browsing of 2D printed images (e.g., multiple windows, coordinated views, saved points, automations.) Better 3D navigation and exploration tools are still to be invented as 3D computer "spaces" become more widespread

## 7. CONCLUSION

As this taxonomy suggests there are a large number of choices to make when designing or choosing an image browser. Improved design based on controlled experiments could improve speed, error rates, and subjective satisfaction. Guidelines are limited and very few have been validated. New automations should be prototyped and tested. Techniques allowing users to specify the needed automation should be investigated. The multiple view browsers will indirectly benefit from an increased attention to the design of window managers and of coordinated window placement strategies.

The many options, features, and parameters we have described show the complexity of image browser interfaces. We defined the terms to describe them and proposed an informal technique to specify them and discuss their design features. After this extensive analysis it might be appropriate to reiterate the goal which is to design the simplest tools that fit the task. In some cases it might mean avoiding a browser entirely! This study shows that browsing is rarely trivial. Before considering the details of an image browser, designers should consider larger screens (or even multiple screens) and denser representations that do not require zooming and panning. Pixels on the screen are precious and effort should be made to display as much information on the screen as the task and user population will permit. Elegance and readability are important for public access, while speed of use should be the goal for expert users who need less zooming, less panning, automated functions, and dense screens.

Image browser design is a lively topic. If zooming and panning cannot be avoided, the tasks and the user population should drive the selection of the browser characteristics. Usability testing remains a requirement because of the still small number of validated guidelines. Beyond the flat screen browsing, novel features for three dimensional browsers are still to be invented.

**Acknowledgments**

We want to thank all the members of the Human-Computer Interaction Laboratory for their helpful suggestions. Partial support for this research was provided by Johnson Controls and OCLC, and by NSF under grant NSF-D-CDR 8803012 and NSF-EEC 94-02384.

**References**

Beard, D. and Walker II, J., Navigational techniques to improve the display of large two-dimensional spaces. *Behaviour and Information Technology 9* , 6 (1990), 451-466.

Bly, S. and Rosenberg, J. A comparison of tiled and overlapped windows. *Proc. of CHI'86 , Human Factors in Computing Systems*, ACM, New York (1986), 101-106.

Chimera, R., Value bars: an information visualization and navigation tool for multiattribute listings. Demo summary in the *Proceedings of CHI'92*, *Human Factors in Computing Systems*, ACM, New York, (1992), 293-294.

Davies, S., Bury, K., and Darnell, M. An experimental comparison of windowed vs. non windowed operating system environment. *Proc. of the Human-Factors Society 29th Annual Meeting*, (1985), 250-254.

Funke, D., Neal , J. and Paul, R., An approach to intelligent automated window management, *International Journal of Man-Machine Studies 38*, (1993), 949-983.

Hudson, S. and Mohamed, S., Interactive specification of flexible user interface displays, *ACM Transactions on Information Systems 8*, 3 (July 1990), 269-288.

Leung, Y. K., Human-computer interface techniques for map based diagrams, in Salvendy, G. and Smith, M., Eds., *Designing and Using Human-Computer Interfaces and Knowledge Based Systems,* 361-368, Elsevier, Amsterdam, (1989).

Plaisant, C., Carr, D., and Hasegawa, H., When an intermediate view maters a 2D-browser experiment, University of Maryland, Center for Automation Research Technical Report, CAR-TR-645, (Oct. 1992).

Sarkar, M. and Brown, M., Graphical fisheyes views of graphs, *Proceedings of CHI'92, Human Factors in Computing Systems*, ACM, (1992), 83-92

Schaffer, D., Zuo, Z., Greenberg, S, Bartram, L, Dill, J. Dubs, S. and Roseman, M., Navigating hierarchically clustered networks through fisheye and full-zoom methods. To appear in *ACM Trans. on Information Systems.*, 1994.

Shneiderman, Ben, Designing the User Interface: Strategies for Effective Human-Computer Interaction: Second Edition, Addison-Wesley Publ. Co., Reading, MA (1992).

Spence, R. and Apperley, M. Database navigation: an office environment for the professional, *Behaviour and Information Technology 1,* 1.(1982), 43-54.

Address questions about this article to Catherine Plaisant, A.V. Williams Building, University of Maryland, College Park, MD 20742 (plaisant@cs.umd.edu),

or see hhtp://www.cs.umd.edu/projects/hcil/