Chapter 7
Interactions

In the previous chapters, we looked at some visual representations with which the user can interact to modify the general view. For example, in a three-dimensional view, the user may rotate the image, to be able to reveal rear objects that could otherwise be occluded by others in the foreground. Basically, actions like geometric transformations of the view, filtering the input data, or even changing the type of representation allow the user to make the best of the visual representations; sometimes this is fundamental to carrying out explorative analysis on a collection of data. In this chapter, we'll take a closer look at the mechanism of interaction and show how this can be useful with numerous practical examples.

7.1 The Problem of Information Overload

In the book's introduction, we touched on how the quantity of information to be processed has grown to such a high level over the past decade that by now we just can't do without computers, handhelds, cell phones, and anything else that can be used not only for communicating with others, but also for memorizing and organizing our ideas, information, photos, letters, reports, etc. Information visualization seeks to meet the needs of those users who, with computers, make use of graphical interfaces that take advantage of humans' notable perceptual ability of vision systems for visually exploring data at various levels of abstraction. In the previous chapters, we reviewed some techniques that can be useful for representing certain data types and meet specific user goals. Unfortunately, all of the techniques encounter the same physical problem when a certain amount of data is exceeded: the lack of space for representing all of the data intended for visualization.

Interactive tools can come to our aid. In particular, information visualization systems appear to be most useful when a user can modify the input data, change the visual mapping, or manipulate the view generated. Interactions facilitate data exploration and may uncover relationships that could remain hidden in a static view. The techniques that we are going to examine, regardless of their simplicity or sophistication, have a common objective: that of providing a global overview of the collection of data and, at the same time, letting users analyse specific details or parts that they may judge as relevant to their goal.

In 1996 Ben Shneiderman wrote an article [52] that defined, for the first time, a taxonomy of the possible tasks one may achieve with the graphical interfaces that make use of visual representations. The article became famous among researchers who deal with this discipline, thanks to a guideline, genial in its simplicity, of designing a interactive information visualization system, defined by Shneiderman as the "information visualization's mantra":[1]

First, **overview**,
then, **zoom** and **filtering**,
finally, **details on demand**.

The mantra clearly indicates how an information visualization system can support users in the process of searching for information. It is necessary to provide a global overview of the entire collection of data, so that users gain an understanding of the entire dataset, than users may filter the data to focus on a specific part of particular interest. Finally, all the details of a particular instance of data ought to be visible, should the user require them.

7.2 Types of Interactive Visual Representations

In Chapter 2, we saw a model of the visualization process, which allows the user to intervene at every stage of the process (Fig. 2.1). Depending on the type of interaction a user can perform, we can specify the following types of representation:

- **Static representations** don't allow users to perform any type of interaction, and only a single, unmodifiable view is generated.
- **Manipulable representations** allow users to manipulate the process that generates the view, via zooming, rotation, panning, etc.
- **Transformable representations** allow users to manipulate, in the preprocessing phase, the input data of the representations, for example through data filtering. These manipulations usually influence and modify the images that are generated.

In the following sections, we will examine some of the most interesting techniques for the manipulation and transformation of visual representations.

[1] The mantra is a form of spiritual practice, used in some religions, involving the continuous repetition of a word or certain number of phrases, with the aim of attaining a particular effect on a mental level. The most well-known mantra is the *Om* mantra practiced in Buddhism.

7.3 Manipulable Representations

A process of interactive visual representation can be defined as *manipulable* when the user can intervene and manipulate the view generated. The most common techniques apply geometric transformations, such as zooming onto a particular part of the view or rotating an image in a three-dimensional view. There are several techniques for view manipulation other than zooming and rotation; the most popular can be grouped into three categories:

- scrolling,
- overview + details,
- focus + context.

We are about to examine each of these categories.

7.3.1 Scrolling

Scrolling is a very common technique that we regularly use when working with a computer. It consists of visualizing only the part of the view that can be contained in the physical area on the screen and, through appropriate movement bars called *scrollbars*, allows the user to move the visible parts. This technique has the advantage of being acquired and consolidated in all of the basic window systems but has a defect of hiding the global vision of the entire view, which can be a problem when there is a need to contextualize the visual part within the entire collection of data.

7.3.2 Overview + Details

The basic idea of the *overview + details* technique is to show a detailed part of the view on the screen (exactly as scrolling does) while providing an global, less detailed, view of the entire representation. The name is derived from this approach, which shows the overall structure of the content to help users make first impressions, and understand how the entire collection is organized. The details, on the other hand, let the user "drill down" from that view into the details as they need to, keeping both levels visible for quick iteration. We show an example of this technique in Fig. 7.1, in which the details of a graph are visualized in the window, which maintains an overview in a window in the top right corner with an indication of the area represented in detail. Through zooming and panning operations, the user can widen or narrow the area visualized in detail and move (by operating the mouse on the rectangle with the red borders) the zone that displays the details.

This technique is applied in some applications that use the magnifying glass metaphor to show the details of a zone of the view. In the Microsoft Windows characters map (Fig. 7.2), for instance, the user can see an enlarged view of each letter

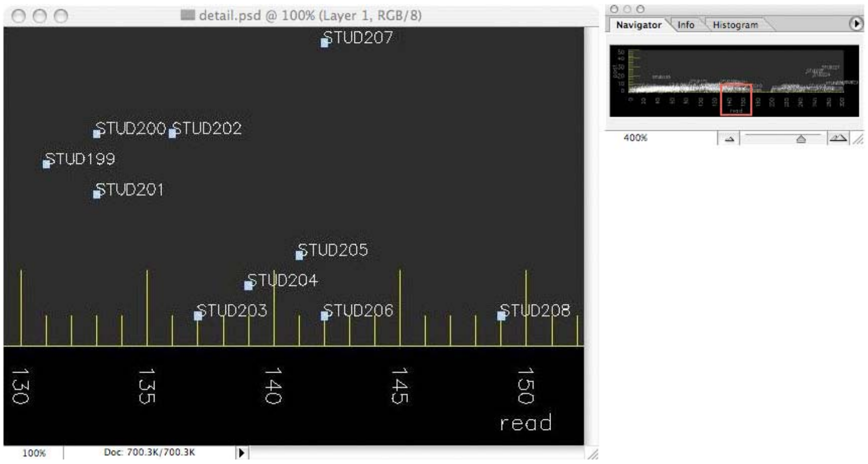


Fig. 7.1 Example of the application of the Overview + Details technique.

by passing the mouse pointer over a character, allowing the user to easily distinguish it among the other characters in a very contained space.

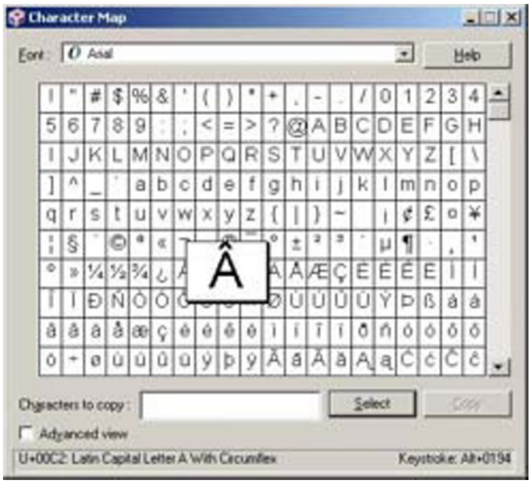
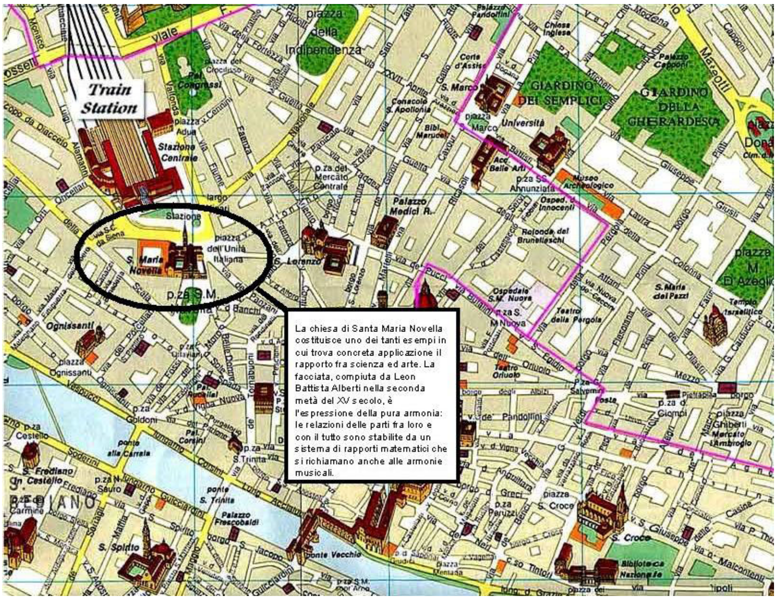


Fig. 7.2 Map of the characters used by Microsoft Windows to assist the user in the choice of a special character. © Microsoft.

The magnifying glass metaphor are best applied in applications that require the maximization of space for the overview portion. Figure 7.3 illustrates an application that shows a tourist map of Florence. To dedicate all of the available space to the visualization of the map, an appropriate lens can be placed over the map and moved by using the mouse. When the lens is placed over a building of touristic interest, a



La chiesa di Santa Maria Novella costituisce uno dei tanti esempi in cui trova concreta applicazione il rapporto fra scienza ed arte. La facciata, compiuta da Leon Battista Alberti nella seconda metà del XV secolo, è l'espressione della pura armonia: le relazioni delle parti fra loro e con il tutto sono stabilite da un sistema di rapporti matematici che si richiamano anche alle armonie musicali.

Fig. 7.3 Tourist map of Florence. By moving the lens over the building of touristic interest, it is possible to see an area in which the details of the building are illustrated.

small section of text becomes visible featuring a detailed description of the selected building.

Other examples of applications that use the overview + details technique are the *zoomable user interfaces* (ZUIs), which use zooming as the main method for exploring items of information that are too numerous to be displayed on a single screen. ZUIs have been investigated for several years at the *Human–Computer Interaction Lab* (HCIL) of the University of Maryland, which has also made two toolkits for building such interfaces available, Jazz and Piccolo[2] [3, 2]. The ZUIs display the graphical representation on a virtual desktop instead of in a window. The virtual desktop is rather wide and has a very high resolution. A portion of this huge virtual desktop is seen on the display through a virtual camera that the user can pan across the surface in two dimensions, and smoothly zoom into objects of interest, for more detailed information, and zoom out for an overview.

An example of an application developed using the Piccolo ZUI toolkit is PhotoMesa,[3] a zoomable image browser. It provides a zoomable environment for users to view the images contained in multiple directories of the computer, and allows users to surf and browse through simple navigational commands to smoothly zoom in and out (see a screenshot in Fig. 7.4).

[2] http://www.cs.umd.edu/hcil/piccolo.

[3] http://www.windsorinterfaces.com/photomesa.shtml.

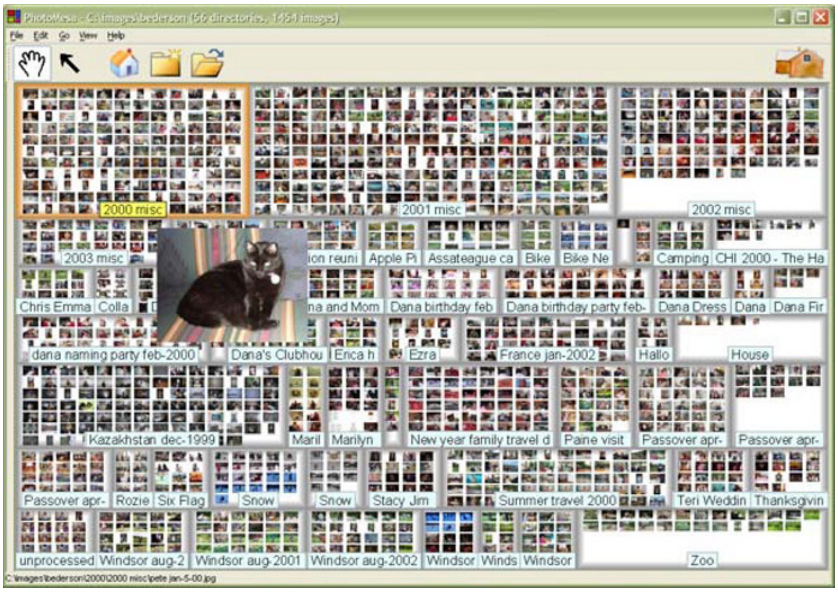


Fig. 7.4 PhotoMesa zoomable image browser. Image reproduced with the permission of Ben Bederson, Windsor Interfaces, Inc.

More recently, Apple adopted a similar interaction modality in the user interface of the iPhone, which enables the user to zoom in and out of web pages and photos by placing two fingers on the screen and sliding them farther apart or closer together.

7.3.3 Focus + Context

Another family of view manipulation techniques is called *focus + context*. This consists of simultaneously providing the user with detailed (focus) and contextual (context) information in the same area, without using two separate views. The goal is to dedicate the whole space in the screen to the detailed view, which is what interests the user but, at the same time, it strives to retain the context within which the details are positioned. This can be done either through the distortion of the view or through the elimination of the details in the peripheral zones.

The distortion techniques literally create a distortion of the image generated by the view, to dedicate a large part of the screen to the details of interest to the user. Some notable works, which we find very often in literature, are the *bifocal views*, the *perspective wall*, and the *fisheye view*, whereas *hyperbolic browser* and *SpaceTree* use a technique that eliminates the details of the peripheral zone.

7.3.3.1 Bifocal View

Over the last 15 years, the technology of computer monitors and displays has evolved (although not at the same frenzied speed as other computer components, such as the processor and memory). In fact, in the early 1990s, the most common monitor resolution was 640 × 480 pixels; today it is possible to avail of monitors with a 1600 × 1200 pixels resolution at a reasonable cost. However, despite this increase in the number of pixels and, consequently, the visible area, the available space is not always sufficient for containing more complex representations.

Robert Spence, as far back as 1982, was the first to propose a solution by means of a view distortion technique [53]. If an image is too large to be visualized in the computer screen, then a distorted vision can be provided to contain it in the visible area of the screen. The problem, however, is how and in what way to apply the distortion.

As we have mentioned many times, every visual representation should allow the user to observe the details of a part of the view and, at the same time, retain an idea of the global view. Following this principle, distortion techniques combine a part of the screen (central) in which the detailed information (the *focus*) is presented and the peripheral parts are distorted (through a transformation function), allowing the image to be contained in the screen and simultaneously providing the context in which the detail is placed (the *context*). In particular, the *bifocal display* applies a transformation outside the two vertical axes, as shown in Fig. 7.5.

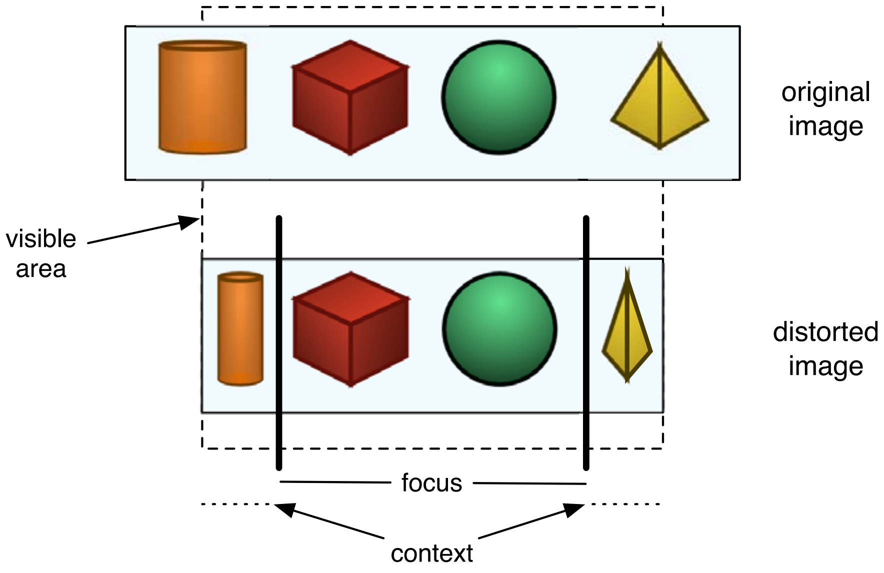


Fig. 7.5 Bifocal display.

7.3.3.2 Perspective Wall

The *perspective wall* [40] is an application derived from the bifocal display, but it uses a three-dimensional perspective for the context. The front wall is used to display the focus of the data while the two side walls show contextual information with a decreasing magnification level from the front wall (see Fig. 7.6). This intuitive distortion of the layout provides efficient use of screen space. The technique was developed at the Xerox Palo Alto Research Center and is currently used by Business Objects[4] (the same company that produces TableLens) for *TimeWall* software. Further mapping on the space has been added to the perspective wall. For example, in Fig. 7.6, the icons represent movies, arranged horizontally in chronological order and vertically by distributors. Color is used to distinguish the genre (action, comedy, drama, etc.). In this case, we assume that the users are particularly interested in the release date and the distributor of the movie.

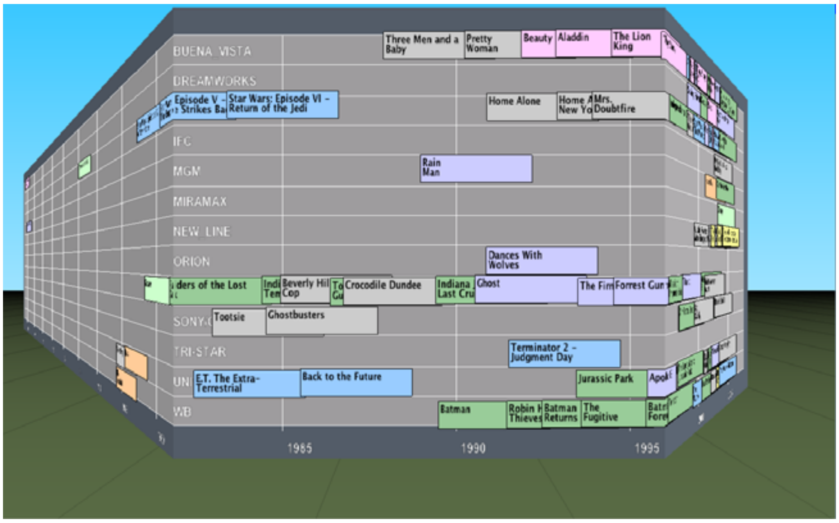


Fig. 7.6 Perspective wall. Image generated by ©Business Objects TimeWall software.

7.3.3.3 Fisheye View

Created from a proposal by George Furnas, the *fisheye view* [23] introduces a visualization technique of the focus + context type, inspired by the human visual system. When observing an object, the human visual system simultaneously perceives the

[4] Business Objects S.A. is a global business intelligence (BI) software company recently acquired by SAP AG. In 2007, Business Objects acquired Inxight Software, the original producer of TableLens and TimeWall software. http://www.businessobjects.com.

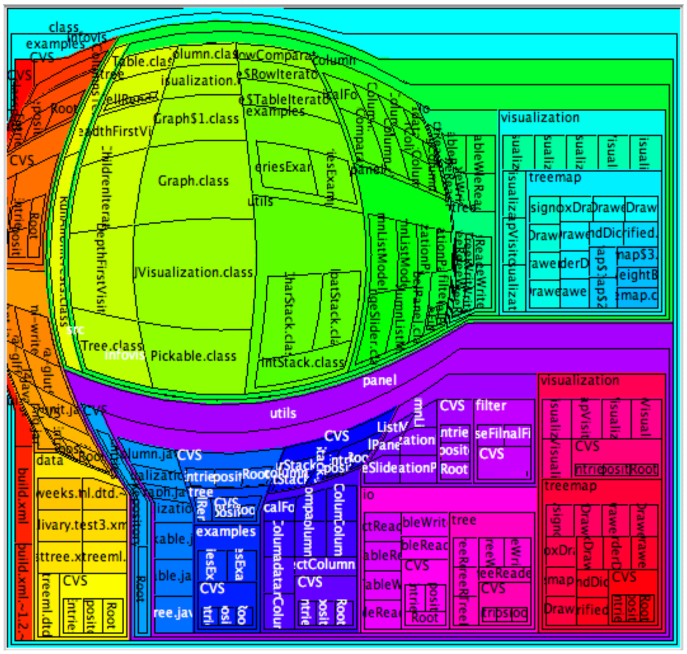


Fig. 7.7 Fisheye distorted treemap.

object itself and the immediate nearby area. We focus on a particular zone, but the area surrounding remains perceptible to our attention, with a detail that fades increasingly the further it gets from the image's focal point. The basic principle of this technique is precisely that of representing the most relevant information in a focal point, with the maximum detail, while the peripheral information is presented with lesser detail. A level of relevance is defined for each element; it is calculated on the importance of the information and its distance from the center of the focus. Figure 7.7 shows a treemap in which a fisheye distortion has been applied through a magnifying glass-type effect.

Researchers have tried to maximize the use of this technique in all ways, proposing its use in numerous contexts. One of these is the *fisheye menus*, defined at the HCIL Lab at the University of Maryland [1]. The goal of this application is to facilitate the user in choosing options presented through drop-down menus, which are ubiquitous in all graphical user interfaces. The fisheye menus dynamically change the dimension of the font of the drop-down menu, so that the elements near the cursor appear in a normal font, while the other, distant elements still remain visible, but with a font size that decreases gradually the farther the element is from the cursor. Figure 7.8 gives an example that requires the user to choose among 256 nations represented in the menu. The version on the left presents a normal drop-down menu, while that on the right is a version realized with a fisheye view. The researchers who created this type of menu hope for widespread use in commercial applications.

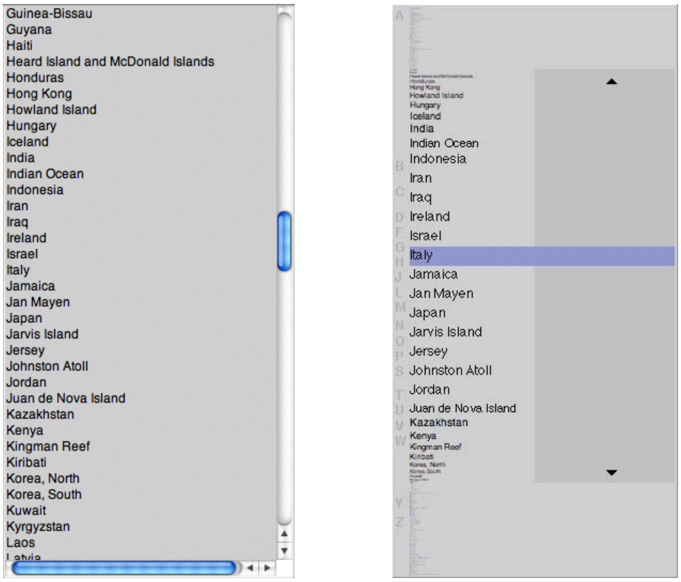


Fig. 7.8 Example of a drop-down menu with scrollbar (left) and the fisheye version (right). Image reproduced with the permission of Ben Bederson, University of Maryland, Human–Computer Interaction Lab.

However, the common drop-down menu is consolidated in almost any graphical interface, and the difficulty of using the lens of the fisheye version (in fact, a small movement of the mouse provokes a notable change to elements in the focus) makes its adoption in commonly used interfaces very challenging.

7.3.3.4 Hyperbolic browser

Another focus + context type application that has become “historical” was developed by researchers of the Xerox PARC Lab at the beginning of the 1990s and is called *hyperbolic browser* (also known as *hyperbolic tree*) [37]. It deals with an interactive technique that aims to represent data structures of very large trees. Currently, this technique is used in the *StarTree* software, produced by Business Objects.

The root node is initially represented at the center of the image, while the child nodes are positioned in a radial arrangement, with lesser detail. Peripheral nodes are also displayed in an oval region, to a level of detail that still provides the context of the central node (see the example in Fig. 7.9). The interesting aspect is that the user can explore the tree by clicking on the nodes and moving them with the mouse.

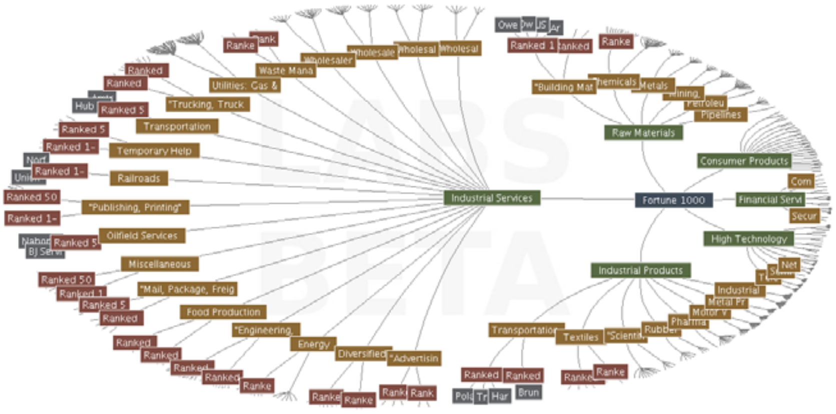


Fig. 7.9 Example of a hyperbolic browser. Image generated by ©Business Objects StarTree software.

Clicking on a peripheral node moves it to the foreground, at the center of the figure, and the entire visualization is reorganized with the new focus at the center.

7.3.3.5 SpaceTree

SpaceTree [26] is another example of representation of hierarchical data structures in which the focus + context principle is used for datasets too large to be visualized in the visible area of the screen. Developed by the researchers of the University of Maryland, SpaceTree, rather than visualizing the entire structure of the tree, lets users explore the tree interactively, dynamically rescaling the branches of the tree to best fit the available screen space. Users can navigate the tree by clicking on nodes to open branches, or by using the arrow keys to navigate among siblings, ancestors, and descendants (see Fig. 7.10). Closed branches (because of lack of space) are represented with a triangle. The shading of the triangle is proportional to the total number of nodes in the subtree, the height of the triangle represents the depth of the subtree, and the base is proportional to the average width (that is, the number of items divided by the depth). It includes, in addition, integrated search and filter functions. SpaceTree is interesting as the entire visible space is used to visualize only a part of the tree but, simultaneously, allows the user to explore parts of the tree and reorganize the visible area with an attractive animation, without losing the context. It is advisable to try the demo to grasp this application's full potential. [5]

[5] A demo of SpaceTree is available at the University of Maryland website; http://www.cs.umd.edu/hcil/spacetree.

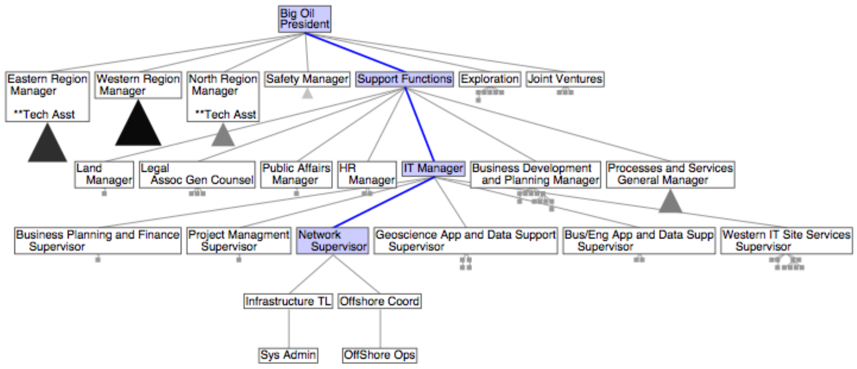

Big Oil President
Eastern Region Manager **Tech Asst
Western Region Manager
North Region Manager **Tech Asst
Safety Manager
Support Functions
Exploration
Joint Ventures
Land Manager
Legal Assoc Gen Counsel
Public Affairs Manager
HR Manager
IT Manager
Business Development and Planning Manager
Processes and Services General Manager
Business Planning and Finance Supervisor
Project Managment Supervisor
Network Supervisor
Geoscience App and Data Support Supervisor
Bus/Eng App and Data Supp Supervisor
Western IT Site Services Supervisor
Infrastructure TL
Offshore Coord
Sys Admin
OffShore Ops

Fig. 7.10 SpaceTree. Image reproduced with the permission of Ben Bederson, University of Maryland, Human–Computer Interaction Lab.

7.4 Transformable Representations

An interactive visual representation can be defined as *transformable* when the user can in some way intervene and manipulate the phases of preprocessing and/or visual mapping. This is very useful in all cases of explorative analysis, when the users are unfamiliar with the contents of a collection or have a limited understanding of how the data is structured, or when in search of interesting structural properties (see Section 1.2). The opportunity to interfere with input data or the mapping is very useful because it can help in the research and definition of the data properties, and may lead to interesting insights. We are going to examine some of the most common transformation techniques, which include *filtering, data reordering, dynamic queries, magic lens,* and *attribute explorer*.

7.4.1 Filtering Input Data

One of the simplest operations (and also the most obvious) that can be carried out is that of filtering the data that are used as input in a visualization. *Data filtering* is part of the preprocessing phase in the model of visualization and can be used for

1. eliminating any data items and attributes of the dataset that do not have to be taken into consideration by the visual representation because they are neither required nor relevant,
2. carrying out analysis focused on parts of the dataset; for example, to see how the visualization is modified by including or excluding some attributes or instances of the dataset.

The second option is the most interesting, as it allows an analyst to carry out explorative analysis on subcollections of data and draw general conclusions. For example, a possible application could be the analysis of the internal sales of a multinational company. In a hypothetical visual representation of the sales, an analyst could filter the input data relative to a certain branch of the company in order to compare the sales of this branch against those of other branches.

We have already seen an example of data filtering in Section 4.2.1, where we applied the brushing of the values on the parallel coordinates.

7.4.2 Data Reordering

Another operation that can be carried out in the preprocessing phase is the reordering of the entire dataset by the values of a certain attribute. Basically, it is what is achieved by using the TableLens application that we saw in Section 4.2.3. In this application, if we click on the heading of a column, the data are sorted by that column's values. The operation can be useful for understanding if there is a possible correlation between the sorted attribute and the dataset's other attributes.

7.4.3 Dynamic Queries

Let's suppose that we have a database containing information with which we are unfamiliar. Also, suppose that the only tools at our disposal for retrieving the data are queries expressed in SQL language. This language is standard for querying a database, but, unfortunately, it is not the ideal tool for building a mental model of the entire dataset. It is difficult to use (one needs to know the syntax of the language); it doesn't tolerate imprecision (in other words, it only returns the records that satisfy determined criteria); moreover, it doesn't offer suggestions on how the query should be reformulated to have more (or less) results; even worse, it hides the context of the elements that are returned in output.

We'll try to explain the idea with a practical example. Imagine having a movie database, the structure could be similar to the following table:

Title	**Actor**	**Director**	**Year**	**Genre**
Monster-in-Law	Jennifer Lopez	Robert Luketic	2005	Comedy
Entrapment	Sean Connery	Jon Amiel	1999	Comedy
The Aviator	Leonardo DiCaprio	Martin Scorsese	2004	Drama
Murder on the Orient Express	Sean Connery	Sidney Lumet	1974	Thriller

We'd like to know which films were produced after 1975 that feature Sean Connery in the lead role. If we are using a relational database, we need to write an SQL query like the following:

```
SELECT  title  FROM film
WHERE actor='Sean Connery'
AND year > 1975
```

Certainly, the average user can't be expected to understand and insert this type of query. It's necessary to create a specific user interface that allows users to carry out database queries simply and without needing to be familiar with any particular query language. The difficulty lies in making users understand what the database contains. If a user doesn't necessarily want to make a specific database query but would like to first understand what the database contains and how it is structured, what type of graphical interface should we create?

Dynamics queries were conceived of precisely for helping users in this task. They can be considered a visual alternative to the use of SQL for database queries; dynamic queries are mainly required to generate graphical representations that change instantly according to how the user manipulates the query controls. A database query is divided into its components, which are graphically represented through manipulable graphical interface objects like buttons, sliders, check boxes, and other control widgets. The user can adjust one of these components and the visual representation will be immediately updated to reflect this change.

Let's look at an example, a prototype developed by the University of Maryland specifically for studying this type of interaction. Even if it is a little dated (going back to 1994), it effectively illustrates the approach used to carry out this type of explorative tasks. The prototype (see Fig. 7.11) proposes an alternative interface for exploring the film in a database containing 1,500 films. It initially represent the entire database through a *starfield display*, that is a colored scatterplot, with individual points representing films in the database. The color corresponds to the film genre (e.g. comedy, drama, musical), while the production year and the popularity index (which is the success attained by that film) are respectively mapped onto the x- and y-axes. In the *mantra* (Section 7.1), this corresponds to the overview of the entire collection of data. The user may find this first visual representation helpful for gaining a general idea of the films available in the entire movie outlet and make some decisions based on the year, genre and popularity of the film.

Then, using the cursors that are found in correspondence to the axes, the user can filter the results based on a range of data and popularity, but it doesn't end here: Thanks to the new component introduced by this application, called *alphaslider* (in Fig. 7.11 on the right), the user can directly select from thousands of titles, actors, actress, and directors. The alphaslider function is identical to that of the drop-down menus, but in a case such as FilmFinder, where one needs to select from hundreds or thousands of different elements, alphaslider offers the advantage of using only one line of text output. Therefore, it is very efficient in its use of screen space. Figure 7.12 displays the result of a possible filtering (second part of the *mantra*), where the now-reduced number of films also allows for the visualization of the labels with the name of the film. In this case, the user has chosen to filter the film based on the actor (Sean Connery), production year (between 1960 and 1995), popularity (at least 4), and duration (between 60 and 269 minutes). A click on one of the points of the scatterplot opens a new window (Fig. 7.13) containing all of the details of

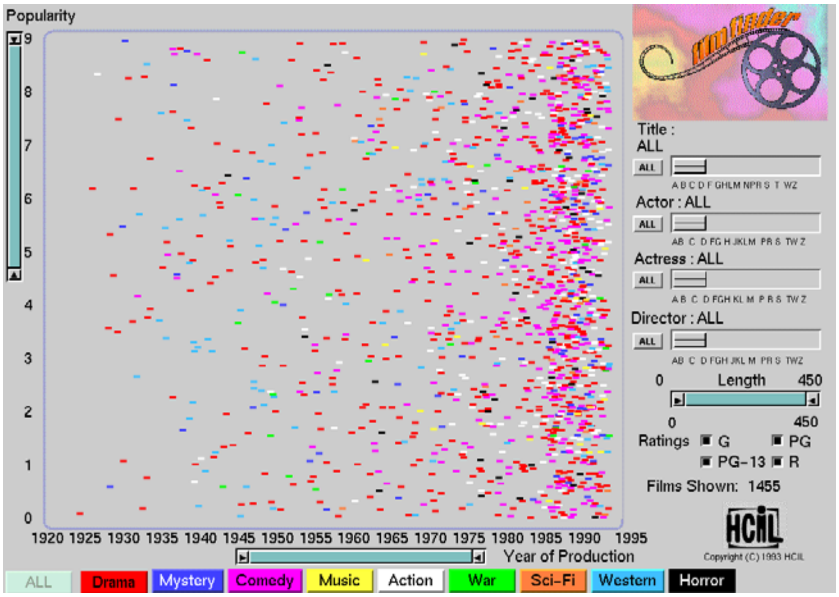
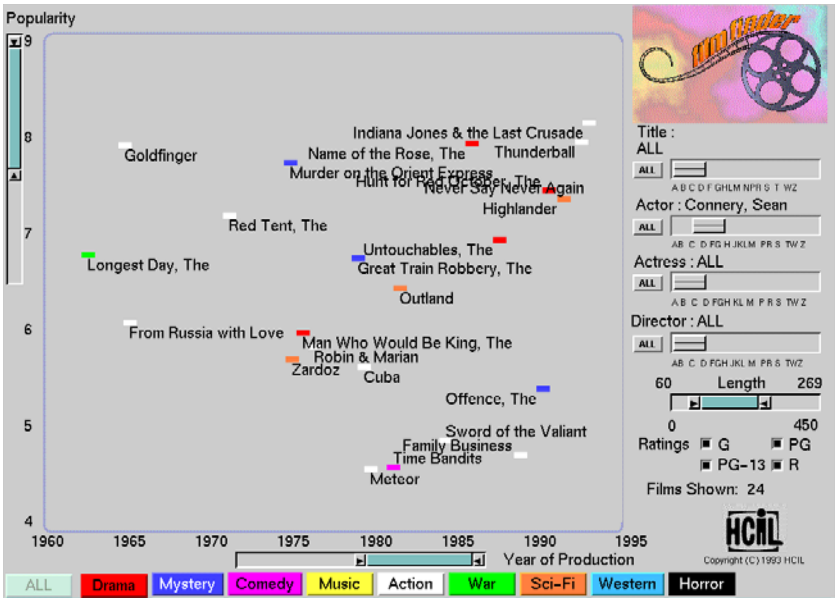


Fig. 7.11 A collective view of all of the available films in FilmFinder. Image reproduced with the permission of Ben Shneiderman, University of Maryland, Human–Computer Interaction Lab.

the film selected, including an image. It is interesting to note how the interaction approach adopted by FilmFinder (overview, filtering, and detail) corresponds to the indications of the *mantra* recommended by Ben Shneiderman.

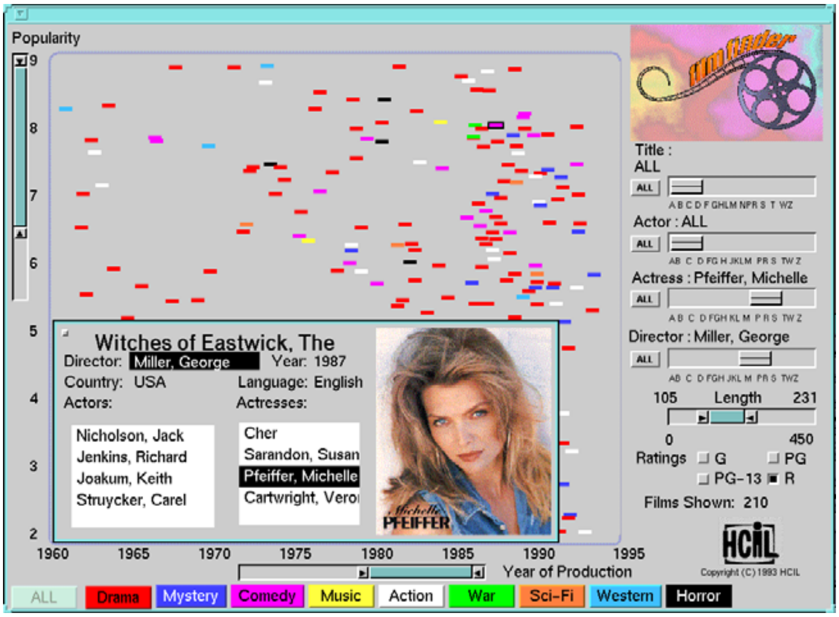
7.4.4 *Magic Lens*

Conceived at the famous Xerox PARC Laboratory, the *magic lens* also constitutes an alternative approach to the dynamic query, which is offered to the user through the metaphor of an "intelligent" lens placed over a visual representation [21]. In a common 2D visual representation, such as a scatterplot, two attributes can be mapped onto the axes, while it is necessary to introduce other mapping, such as to color or shapes, to represent further attributes. When there are several attributes, the magic lens can be very helpful, as it extends the scatterplots or other similar 2D representations without sacrificing their simplicity. Basically, it consists of placing a lens (a movable shaped region) over a graphical representation, that affects the appearance of structures viewed through it. The operation performed by the lens is usually a filtering on the data viewed through the lens. An example appears in Fig. 7.14. The user can make a choice on which attribute to perform the filtering on and which operation to apply (by using the buttons on the right) and can set the threshold for the filter (by adjusting the cursor placed over the lens). The graphical elements



Title : ALL
Actor : Connery, Sean
Actress : ALL
Director : ALL
60 Length 269
0 450
Ratings ■ G ■ PG ■ PG-13 ■ R
Films Shown: 24

Fig. 7.12 Films listed after the filtering operation in FilmFinder. Image reproduced with the permission of Ben Shneiderman, University of Maryland, Human–Computer Interaction Lab.



Witches of Eastwick, The
Director: Miller, George Year: 1987
Country: USA Language: English
Actors: Nicholson, Jack; Jenkins, Richard; Joakum, Keith; Struycker, Carel
Actresses: Cher; Sarandon, Susan; Pfeiffer, Michelle; Cartwright, Veroi

Title : ALL
Actor : ALL
Actress : Pfeiffer, Michelle
Director : Miller, George
105 Length 231
0 450
Ratings ☐ G ☐ PG ☐ PG-13 ■ R
Films Shown: 210

Fig. 7.13 Details of the selected film in FilmFinder. Image reproduced with the permission of Ben Shneiderman, University of Maryland, Human–Computer Interaction Lab.

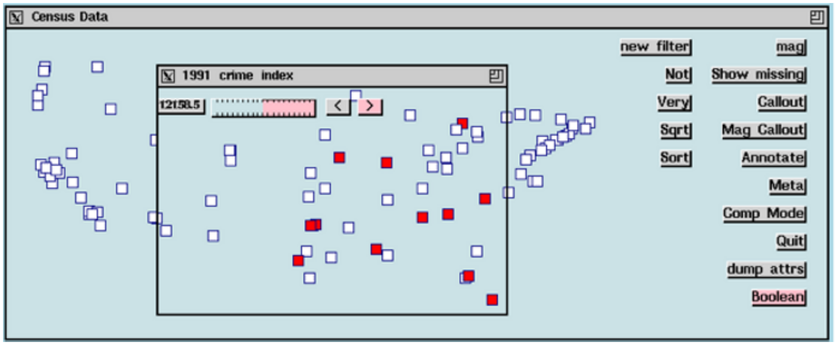


Fig. 7.14 Magic lens allows visual filtering of data in a scatterplot. Image reproduced with the permission of Eric Bier, Xerox PARC.

below the lens will change color according to the value of the filtered attribute. By placing overlapping lenses, one can carry out compositions of operations on more than one attribute.

7.4.5 Attribute Explorer

A further approach to formulate dynamic queries is represented by a system designed by Bob Spence's group in London in 1994 [62]. It uses cursors and histograms to explore the visual representations of a dataset's attributes. Each attribute to be explored is displayed as a histogram, with the range of attribute values segmented along the horizontal axis, and each data point displayed as a "stacked block" within its segment. The blocks in different histograms corresponding to values of a particular instance of the dataset are linked in a way that if the user filters the attribute values in one histogram, this filtering operation is reflected in other histograms as well.

Let's suppose that the dataset to graphically represent contains a number of real estate properties for sale. The properties are characterized by numerous attributes: number of rooms, square meters, age of the property, presence of a yard, and number of bathrooms, to mention just a few. This type of information is best treated with a graphical approach in that often, people, who wish to buy real estate have a very vague idea of what they want to buy. Buyers prefer to be fully understanding of what is available before making a decision on which house to visit for a possible purchase.

Let's suppose we have a dataset such as that reported in the following table:

Ref. N°	m^2	Rooms	Price	Baths	Age
234a	85	3	320,000	1	12
29b	120	4	400,000	2	0
266r	75	2	270,000	1	5
322u	93	3	350,000	1	15
211e	110	4	380,000	1	12
209f	80	3	300,000	1	4
188a	80	3	280,000	1	0
190v	92	3	250,000	1	25

The attribute explorer representation is made up of a collection of five histograms, one for each dependent attribute listed in the previous table (Ref. N° is considered an independent attribute). For simplicity, Fig. 7.15 shows only the histograms relative to the "m^2" (square meters), "Price," "Rooms," and "Baths" attributes. The histogram already provides an idea of the distribution of the values in the scale. We notice, for instance, that three properties have about 80 square meters (as there are three stacked blocks for this value). However, the most interesting aspect of this type of representation consists of

1. allowing the user to filter the values of one or more attributes, by moving the cursor located at the bottom to correspond with the histogram's axes (the color of the blocks changes according to the filtered value, where white indicates that the property meets the applied constraint),
2. linking the histograms of the various attributes, so that the filtering on an attribute is automatically reported in the other histograms as well.

In Fig. 7.15, we can notice that the block of the property having reference number 322u is gray, since we have filtered out properties with a price superior to 300,000 in the histogram on price. The same block, corresponding to property 322u, is colored gray in the other histograms also. This way we have an insight on how many (and which) properties satisfy the filtering on price in terms of square meters, rooms, and baths.

When filtering is applied to multiple attributes simultaneously, we can use the color intensity of blocks to encode how many attributes meet the applied constraint. For instance, we can use a progressively darker shade as attributes fail more constraints.

For a better understanding of attribute explorer, we recommend watching the Bob Spence video.[6]

7.5 Conclusion

In this chapter, we have dealt with interactive visual representations, which we have divided into manipulable and transformable representations. The former allows the

[6] Bob Spence's videos are available at http://www.iis.ee.ic.ac.uk/~r.spence/videos.htm.

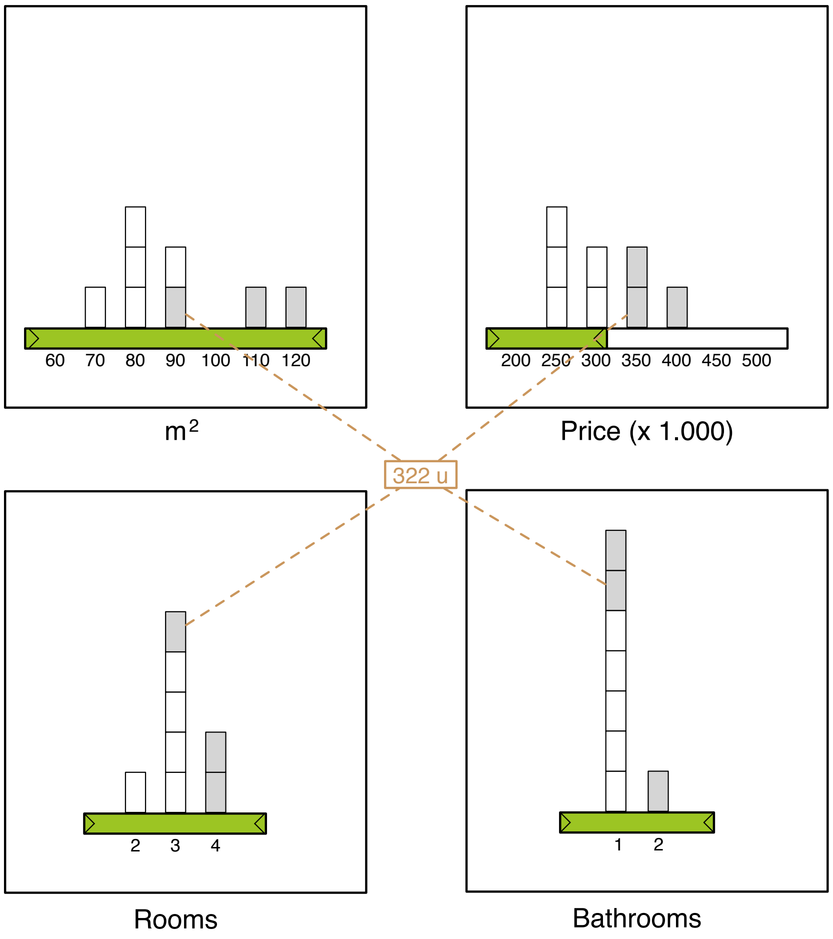


Fig. 7.15 In attribute explorer, the users can filter the values of one or more attributes by moving the cursors; the filtering on an attribute is automatically reported in the other histograms as well.

user to manipulate the view, while the latter allows the user to manipulate the source data and the mapping process. Both are very helpful in the process of explorative analysis, as they facilitate data exploration in large datasets and allow the user to get insights from the data.