# Visualization Methods for Dynamic Queries Databases

*Erik Wistrand*
Department of Computing Science
Göteborg University
S-412 94 Göteborg, Sweden
wistrand@cs.chalmers.se

## ABSTRACT

Modern database systems allow storage and retrieval of large amounts of multidimensional data. This has created a need for fast and flexible visualization methods, aiding the search process. Interactive data display, using dynamic queries has previously been shown to provide the means of easy retrieval of complex data and intuitive exploration of trends and anomalies.

Three visualization methods for database retrieval, using dynamic queries, are described and evaluated in an empirical study: scatter plots, cluster visualizations and geographic maps. An effective method of cluster layout is described and used as a tool in the cluster visualization. Criteria for describing effective visualizations are discussed and used in the study, such as compact data display, detection of anomalies, trends and similarities. The visualizations methods are compared for effectiveness in different retrieval situations. Based on the empirical study a classification and usage recommendation of the visualization methods is presented.

## KEYWORDS:

Visualization, cluster, map, scatter plot, starfield display, database query, dynamic queries, visualization criteria

## INTRODUCTION

Any large information space confronts the user with a multitude of exploration possibilities. Existing methods for exploration are traditional database query languages as SQL (Chamberlain et. al 1976), statistical tools such as SAS, and dynamic queries (Ahlberg et al., 1992, Shneiderman, 1992). All of these tools may incorporate visualizations of various kinds. The purpose of this paper is to provide an evaluation of interactive scatter plots (starfields), geographic maps and cluster display. The result should be possible to use as a visualization guide to different database retrieval situations, such as fast identifications of anomalies or finding trends.

## Databases

### Database models

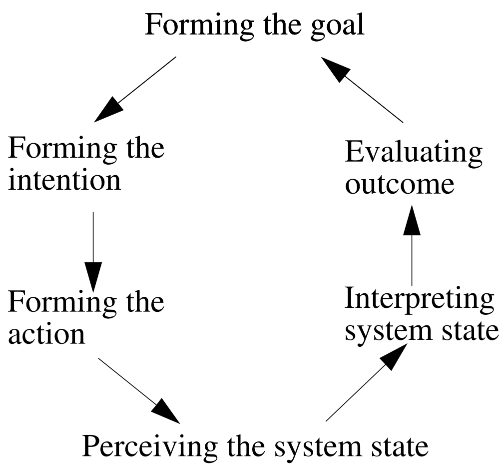
Modern databases containing large amounts of information demands both efficient storage structures, fast search algorithms and a data model users find logic for their application. Examples of database models includes the relational, hierarchical and network models (Elmasri & Navathe, 1989). The relational model is popular in many applications because of the efficient and nonprocedural query languages currently implemented for this database model on many different computer platforms.

### Querying a database

A database management system, DBMS, allows users to store, manage and retrieve data. The DBMS can also support functions for creating visual front ends to the database.

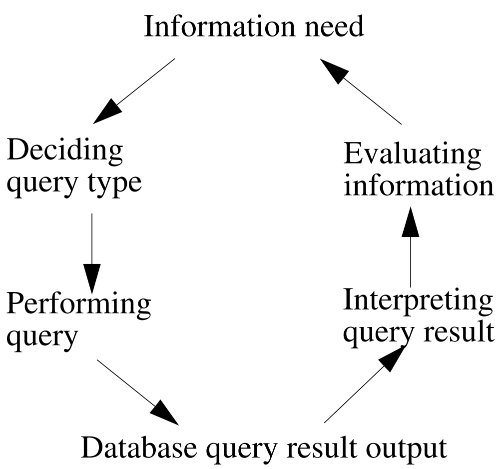
One of the main purposes of a database is the possibility to retrieve information from the stored data. To extract the information, users select a set of *criteria*, creating a database *query*. This query is processed by the database system and a query result is returned and then interpreted by the users.

As a general model for the interaction between humans and a system, Norman (1988) proposed the ”seven stages of action”, describing the relation between the human’s goal and the system’s state as a cycle.




*Norman’s seven stages of action.*

Applied on a DBMS, the seven stages can be used to describe the query process, the goal being the need for information and the system state the DBMS’s representation of a database query result.




*Norman’s seven stages applied on a database query.*

An effective DBMS then needs to have functionality which makes the stages in Norman’s model as effective as possible - good feedback to keep context between stages, an interface which makes possible user actions visible and a result presentation which allows fast and correct interpretation.

## Database query types

When doing a query to a database, users needs to decide what type of information they need to extract. A database represents information internally by storing the complete set of elements, their types and the relations between elements. The user then interprets this data as information. Depending on the purpose of the query, data can be used to extract different information.

An *information object*, either a set of elements or a pattern formed by elements, is the possible result of a database query.

A set of elements represents the criteria used to select them, while a pattern is an abstraction created by the user to understand how selected sets of points relate to each other.

The object types of elements and patterns form the database query classes *detecting elements* and *detecting patterns*. Detecting elements often concerns detecting the properties of individual elements while detecting patterns concerns detecting the combined properties, or gestalt, of a large set of elements.

In this paper four types of queries will be considered: two query types concering patterns detection and two query types concering element detection:

*Detecting elements:*
1. Finding specific elements
2. Detecting anomalies

*Detecting patterns:*
3. Detecting trends
4. Finding similarities

The examples of query types: finding specific elements, trends, anomalies or similarities, were chosen as they are common queries in large databases.

*Finding specific elements*
One of the most common query types in traditional database usage, is finding specific elements given a set of match criteria. This query type requires users to have a clear definition of the match criteria and produces a result consisting of the matching points in the database.

The result can be either a list of single values or a complete relation in a relational database. A list of values is mainly intended for direct user interpretation, while a relation may be used for further processing in the database system.

Such a query is easily expressed in SQL, as in the following example query, asking for all persons (Name) aged more than 30 years, having a salary above $2,000 per month, in a relation R.

```
SELECT Name
FROM
    R
WHERE
    Salary > 2000 AND
    Age < 30
```

*Detecting trends*
Trends in a data set are a the way subsets change their values, as functions of a set of the attributes in the database. Users select this set of attributes, perform the query, and interprets the result as a trend depending on the selected attributes.

In SQL, simple trend detection can be done manually by performing a query and applying a built-in function for analyzing the result. In this example a set of employees are selected and ordered by age. Users can then inspect the result and look for any possible regularity in the employee list.

```
SELECT(Name, Age, Salary)
FROM
    R1,R2
WHERE
    Age > 30 AND
    Salary > 1000
ORDER BY
    AGE
```

More complex trends are difficult to perceive if the number of attributes or matches are large.

*Detecting anomalies*
The anomalies or outliers in a data set are single points which have few values in common with any other point in the set. The anomalies may represent errors in data input or potential points of interest.

When searching for anomalies, users normally have little or no information about the criteria describing the anomalies - the goal of the query is to find both the anomalies themselves and why they do not match the rest of the set.

Detecting anomalies in SQL without knowing in what way they are different from other points in the set, is difficult, if not impossible. One possibility for smaller data sets can be manual inspection and interpretation of the text results from multiple test queries, but that requires extensive user familiarity with the data.

*Finding similarities*
Having two or more subsets of the data space, the criteria best describing all of subsets can used as a measure of similarity between sets. Users may need to find other similar subsets to extend their search query, or to make decisions depending on the properties describing a given set of subsets.

Finding similarities in SQL is a task analogous to detecting anomalies, involving the same problems, thus making it a difficult task.

## Query systems

*Text based query languages*
Text based query languages as SQL and related languages (Elmasri & Navathe 1989) are well-known and have been implemented on most platforms. SQL is based on the relational database model and has semantics which closely resemble the relational operators such as JOIN, SELECT and PROJECT (Elmasri & Navathe, 1989). Both input and output in SQL represents a database relation which can be used for further processing.

Query languages such as SQL need a well-specified set of criteria to produce results from queries, making it difficult for users browsing an unfamiliar database

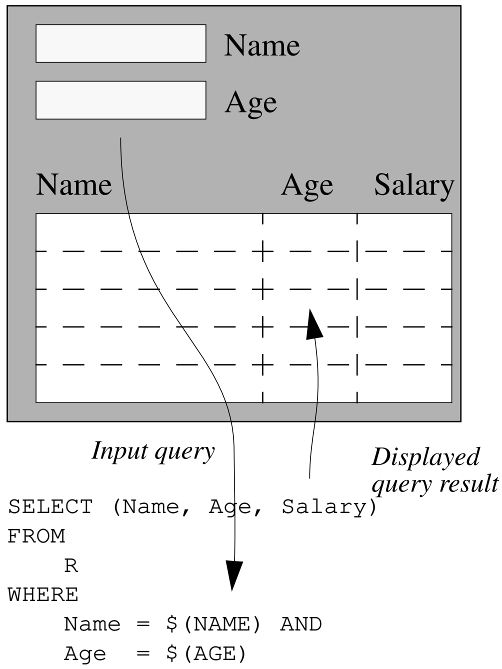
A common problem when searching for elements with text based query languages, is the all-or-nothing result - a query results in either an empty result or an overwhelming list of matching points, with little or no information on what to do to widen or narrow the query.

Simple pattern detection can be accomplished with built-in statistical functions and possible output to a graphic display. Finding anomalies and similarities is difficult and requires several user queries to manually compare the results from each query.

*Form fill-in*
A form fill-in query system is either an interface to a existing database query language or a direct interface to the database system. Form fill-in systems, running on top of query languages, function as interfaces to the basic system and allow users to design and use custom query interfaces to the database. Using forms, users are not forced to remember the sometimes obscure syntax or semantics of the query language. The creation of forms still needs knowledge of the underlying DBMS. If a user creates forms for use by the others, the other users may not be able modify these forms, fixing the types of possible queries to the ones supported by the pre-designed forms.

The same problems as in text-only query languages concerning detection of patterns, anomalies and similarities are present in form fill-in queries.

*A form for seeking persons. The light grey fields are used for entering data, the larger white for displaying output.*

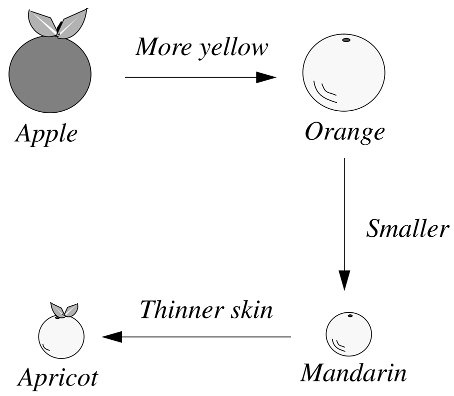



*SQL-like query attached to the form*

*Query-by-example*
The query by example method, or QBE method, (Zloof, 1975) presents users with a set of labeled entries for the attributes in the database. Users can then enter fixed values or expressions describing the values in each entry. This method lets users create search queries ”on the run” without the need of remembering any complex syntax/semantics of the basic query language. Finding specific elements is straight-forward by entering full descriptions. Detecting patterns can be done by guessing pattern expressions and testing them for validity. Still, detecting anomalies or similarities may be hard because of the rigid output format.

A QBE system is similar to a form fill-in system, but extends the form fill-in method to include expressions describing the separate attributes. Both form fill-in and QBE systems allows custom query forms which only uses a subset of the database's set of attributes.

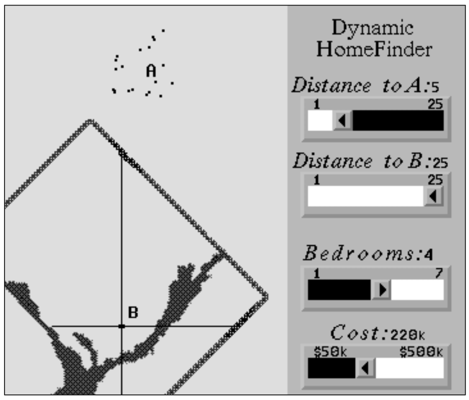
*Incremental modification*
In an incremental modification system, for example Rabbit (Williams, 1984), users are initially presented with an arbitrary item from the database. By modifying this item, a succession of items are selected, all of which may be further modified.




*Finding an apricot from an apple in an incremental modification system.*

This search method stimulates the exploration of the database and is a natural way to shrink a large set to a smaller and manageable subset. However, since then description of the subset lies in the path to the result it can be hard to remember exactly which criteria were used. This can make it difficult to determine any pattern or similarities in the subsets. If the items are displayed without context information, such as how much it differs from previous items or the possible ways to modify it, it is hard to tell an anomaly from a normal item, since items are displayed individually.

*Dynamic queries*
A dynamic queries system (Ahlberg et al. 1992; Shneiderman, 1992) presents users with a both a visualization of data and a set of query devices (Ahlberg & Truvé, 1994) for the database. The query devices (for example sliders, buttons or zooming) allow user do rapid, incremental and reversible changes to the criteria describing the selected subset. The visualization is integrated in the system and functions as a selection mechanism in itself, allowing users to indicate interesting points or areas directly.
Because of the direct feedback and incremental search in dynamic queries query types such as pattern and similarities detection are easy as long as the visualization can display the data points in a user-understandable way. The problem of empty query results are minimized because of the incremental and reversible properties of dynamic queries.




*Dynamiq query interface showing the Dynamic Homefinder (Williamson & Shneiderman, 1992), which lets users select different parameters of houses they are interested in, via sliders and movement of a cursor in the visualization.*

## Query result presentation

Most query language's basic output form is a list of the matched items, in text format. The output text list can be automatically integrated in a form fill-in or QBE interface to a DBMS, providing users with result browsing functions. In an incremental modification system one of the result values is

picked as representative and displayed.
When displaying the results as text only, the data from the database is displayed without modification. The other possibility is to encode the result, producing a visualization of the results. Encodings can be performed by assigning geometric positions or other displayable attributes, for example color, to each value in the result.

If a query needs to be further modified or evaluated, an important property of the result presentation is reusability, allowing users to perform new or modified queries, based on the previous query results. In SQL, the results are, in themselves, complete relations. Since the results are relations they can be used in a further SQL query. SQL thus have the reusability property.

```
SELECT Name
FROM (
  SELECT (Name, Age)
  FROM
      R
  WHERE                      Inner
      Salary > 2000 AND      query
      Age < 30
  )
WHERE
  Name = ″Wistrand″
```

*A nested SQL query, where the result from the inner query is used in the outer.*

A visualization can have the reusability property if users are able to select values directly from the visualization, and use these for modifying the query.

*Effectiveness and Expressiveness*
The presententation of query results affects how users interprets the data. Mackinlay (1987) classified the properties of a result presentation into two groups, the *effectiveness* and the *expressiveness* properties.

Expressiveness of a presentation is it's capabilities to present data at all. For example, some presentations may not be able to display certain kind of relationships in the data such as a monochrome display not being able to display color information.

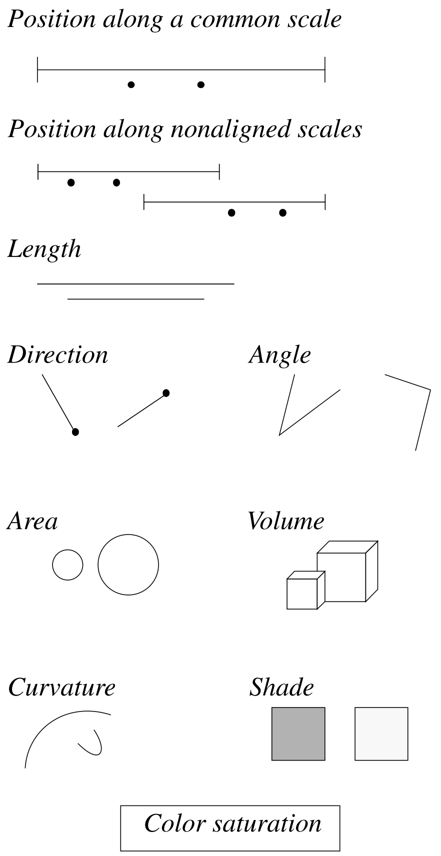
Effectiveness of a presentation is how good it is at conveying information to users. Factors with which determine the effectiveness can be the how fast users percepts information from the presentation, and the accuracy of percepted information.

## Visualizations

A visualization of data needs to have methods of displaying both *quantitative* and *categorical* values. A quantitative value is a ordered value which ranges over a continuous set of data, each value in the range possible to describe with a numeric value. Examples of quantitative values in a database can be age, distance or a popularity scale.

A categorical value has no inherent order and consists of separate items, identified by a name which is clear in the context of the categorical value. Examples of categorical values are sex (male, female) or movie categories (Action, Drama, Comedy etc.).

*The elementary perceptual tasks for quantitative values*
Cleveland and McGill (1984) made a classification of the perceptual tasks of deciding quantitative value into nine basic groups:




*Cleveland's and McGill's nine elementary perceptual tasks.*

These tasks can be ordered from the most to the least effective:

1. Position along a common scale
2. Positions along nonaligned scales
3. Length, direction and angle
4. Area
5. Volume, curvature
6. Shading, color saturation

This ranking list thus indicates a preferred set of methods to display quantitative data.

*Perceptual tasks for categorical values*
Since the perception of categorical values involves different types of tasks than quantitative perception, Mackinlay (1987) made a separate ranking for the effectiveness of tasks used for detecting ordinal and nominal values.

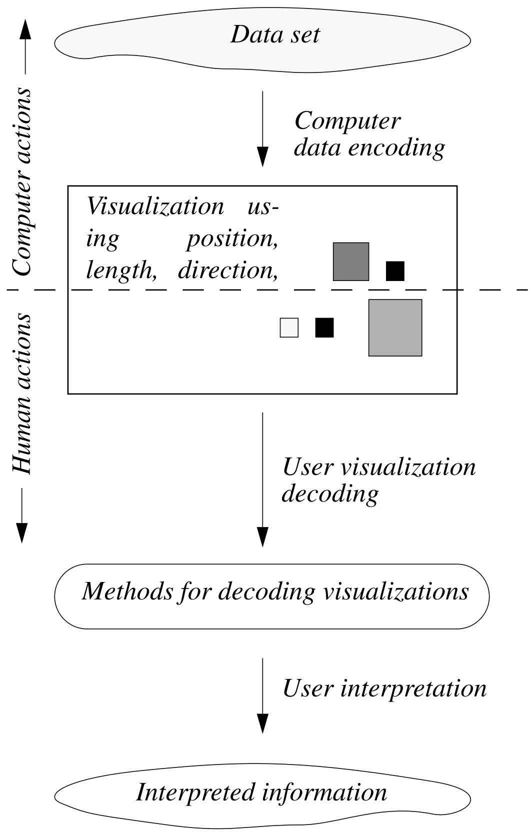
Perceptual tasks in detecting ordinal and nominal values, from the most to the least effective:

| **Ordinal** | **Nominal** |
|---|---|
| 1. Position | 1. Position |
| 2. Shade | 2. Color hue |
| 3. Color saturation | 3. Texture |
| 4. Color hue | 4. Connection |
| 5. Texture | 5. Containment |
| 6. Connection | 6. Shade |
| 7. Containment | 7. Color saturation |
| 8. Length | 8. Shape |
| 9. Angle | 9. Length |
| 10. Slope | 10. Angle |
| 11. Area | 11. Shape |
| 12. Volume | 12. Area |
| 13. Shape | 13. Volume |

The extensions to the Cleveland and McGill list are necessary because that list does not describe some of the most effective categorical perception tasks such as Color saturation or Connection.

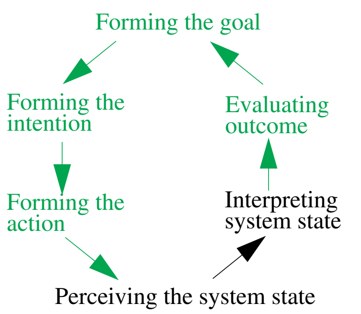
## Decoding visualizations

Any visualization of a dataset represents one of many possible encodings using combinations of the elementary perceptual tasks. Each time users need to extract information from the visualization, the data must be decoded, using one or more methods. The methods can be described by the individual tasks users performs to interpret data. In an visualization implementation the encodings must be well-defined, allowing the visualization designer to predict how users will interpret the data.




*Users decodes an encoded set of data, presented in a visualization. The interpreted information depends on the encoding.*

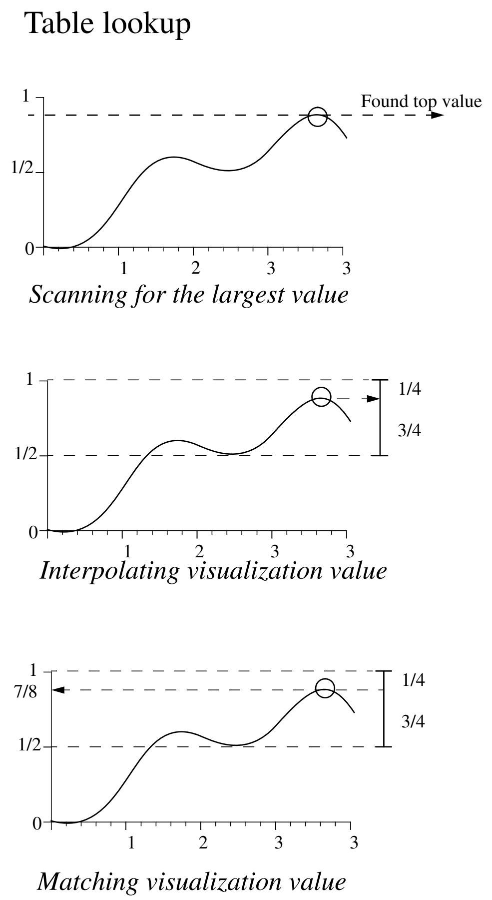
The decoding of a visualization is the part of Norman's seven stages where only the percieving and decoding of the system state is considered.




*Decoding a visualization in context of Norman's seven stages of action (Norman 1988)*

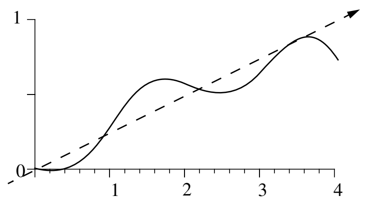
Cleveland and McGill (1994) describes a model for both quantitative and categorical visual perception tasks, based on the concepts of *table lookup* and *pattern perception*. Table lookup is the decoding of individual values, i. e. retrieving the *scale* information from the data. Pattern perception works by creating conceptual objects from several individual values, i.e. retrieving the *physical* information from tha data.

*Table lookup*
Table lookup includes the operations *scanning*, *interpolation* and *matching*. *Scanning* is the process of finding an element given a reference, for example the axis of a coordinate system. *Interpolation* is the process of retrieving the visualization value (symbol type, relative distance, approximate color) of the element, and finally *matching* which translates the visualization value to real-world value.

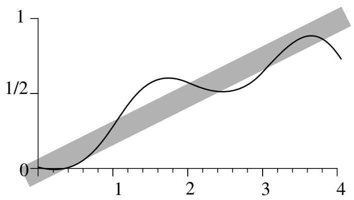


*Pattern perception*
The task of perceiving a pattern is composed of *detection*, *assembly* and *estimation*. First the user recognizes a geometric object in the visualization. This is *detection*. Then the process of *assembly* starts which groups similar geometric objects to a larger object. The last part in pattern perception is *estimation*, which relate different subsets of the larger pattern to each other by comparing the values or differences between subsets.

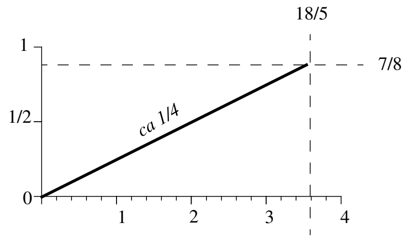
Pattern perception



*Detecting a linear pattern*



*Assembling relevant points*



*Estimating slope*

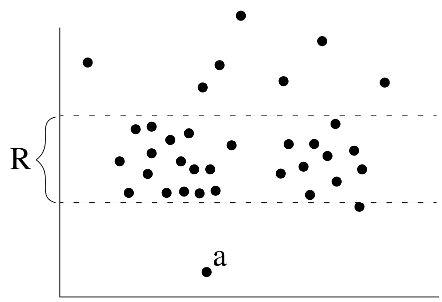
## Database query types and perception tasks

When combining the query tasks initially proposed with the perception tasks the distinction between patterns and elements queries become clear. Finding trends or similarities requires the user to group a set of values via pattern detection, while finding specific elements or anomalies requires table lookup.

*Detecting elements - table lookup:*
1. Finding specific elements
2. Detecting anomalies

*Detecting patterns - pattern detection:*
3. Detecting trends
4. Finding simiralities

For example, if the visualization shows a set of points with a pair of dense clusters and one outlier, users can find the similarity between the clusters and note the anomaly lying outside.




A possible property, describing the similarity between the clusters, constructed from this visualization can be ”the points lying within the range R”. The anomaly ’a’ can then be represented with the property ”the point lying below R”.

## Visualization dimensions

A visualization can be classified by the number of geometric dimensions it uses.
Often used dimensions is 2D and 3D, the 3D visualizations either represented in 2D on paper or a computer screen, or as physical models and in 3D representation such as head-mounted screens giving users a 3D experience of displayed data.

Hybrid dimension visualization are combinations which uses a few dimensions for some attributes and other dimensions for the remaining attributes in the dataset. Examples are the perspective wall (Mackinlay et al., 1991), using distance and perspective as representation for importance, or worlds within worlds (Feiner & Beshers, 1990), which allows arbitrarily nesting of coordinate system inside each other. A nested coordinate system is the dependent on its position in the outer system.

## Visualization types

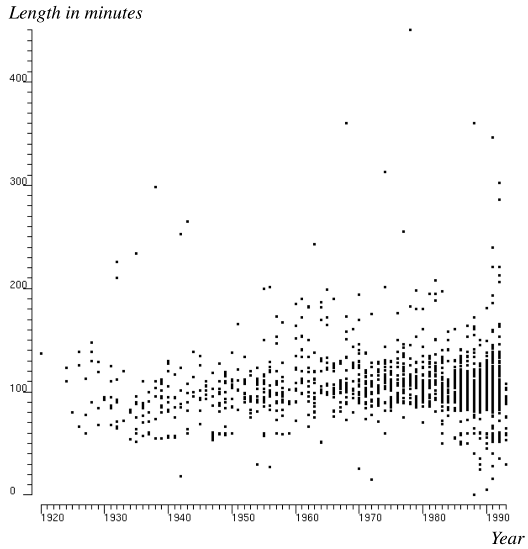
When designing a visualization, the properties of the data sets must be taken into consideration.
A dataset consists of a set of *m* points in a *n*-dimensional space,

$$M = \{ \quad \begin{aligned} & p_1 = (x_{11}, ..., x_{1n}), \\ & p_2 = (x_{11}, ..., x_{1n}), \\ & ..., \\ & p_m = (x_{m1}, ..., x_{mn}) \; \} \end{aligned}$$

Each point $p_i$ can have different types in each dimension. Types are typically numeric, strings, categorical types or even sets in a hierarchical data set.

When such a set is displayed in *k* dimensions, where *k*<*m,* the resulting loss of information necessitates the use of other means than physical dimension to represent the data dimensions.

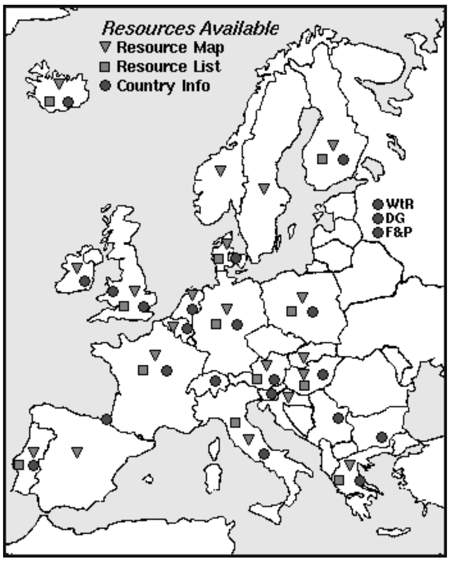
*Scatter plot*
Direct mapping of *k* of the *n* dimensions to the display dimensions results in a cartesian point visualization, or a *scatter plot.*




*Scatter plot of 1800 movies, produced from 1920 to 1993. The middle length of 100 minutes is easily detected as well as the clear trend of more produced movies later in the century. A subtle trend of longer movies later in the century can also be detected.*

The elementary perception task in a scatter plot is alignment along a common scale, indicating good accuracy in a table lookup action. The pattern detection action is natural as long as the pattern relates to the axes in a direct way.

*Map visualization*
The scatter plot may be enhanced by providing context to the individual points, with a background picture to produce a *map visualization*. The picture can be included in the data set as a separate element if the database allows complex elements. With this approach there is no formal difference between the picture and the elements but the user interpretation of the map.




*Map of WWW resources (Brandon, 1994) (http://wings.buffalo.edu/world/europe.html)*

The background map is a often-used way of enhancing the understanding of a scatter plot. Most people have been in contact with maps and readily interprets the map visualization as a direct representation of geographical data. The map lines gives additional cues to the interpretation of the individual points, largely making any axes unnecessary.

.

*Cluster visualization*

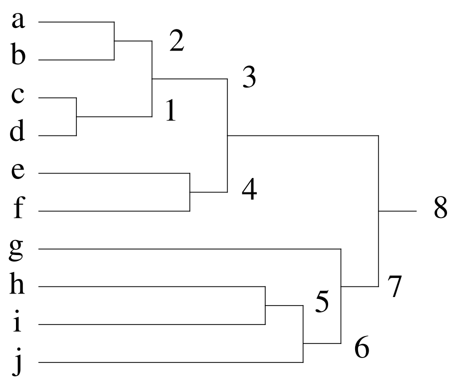
Instead of mapping a few dimensions directly to the visualization, it is possible to cluster similar data points near each other in the visualization, giving a *cluster visualization*. Clustering has, for example, been used in document searching tools for easy access to related subjects (Sassen & Caid; Lin et al., 1991). Clustering can be done hierarchical and non-hierarchical, and also in a network-based model (Bobick, 1987). Only the hierarchical model will be described here.

Hierarchical clustering , allowing clusters to contain other clusters, is either done as an *agglomeration* or a *division* of the data points.

In the *agglomerative* method, each point is initially considered a separate cluster. The nearest clusters (measured in a suitable way) are then merged to form a new cluster. This process is repeated until only a single cluster remains.

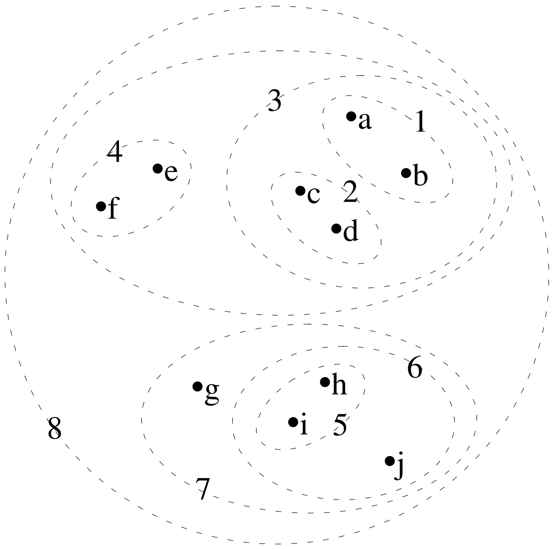
The *divisive* method, the data set is recursively divided into subsets using a threshold function which decides the subset a point belongs to.

The result of both agglomerative and divisive clustering are a dendrogram describing the hierarchy of clusters.

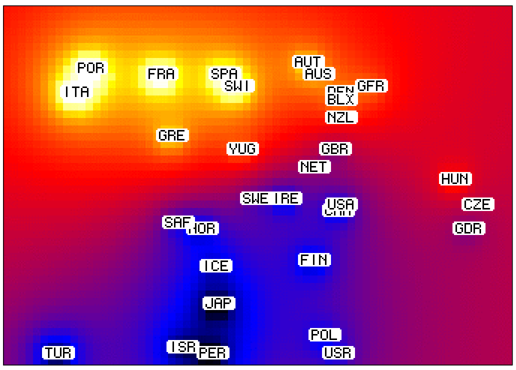



*Example dendrogram describing a clustering process of the points {a, ..., j}. Each horizontal line represents a cluster and each vertical line the combination of two clusters. The digits identifies the hierarchical clusters.*

The dendrogram can be displayed directly (dendrograms have a long tradition in the classification systems of botany and evolution), or it can be further processed to a cluster visualization, where the each point is positioned as near as possible to similar points.




*Cluster visualization of the points {a, ..., j}. The ellipsis 1-8 mark the hierarchical clusters*

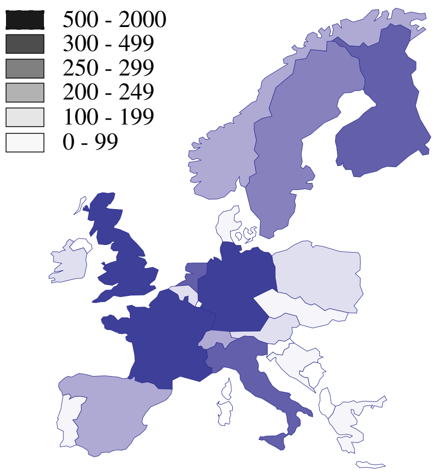



*Cluster visualization of the consumption of wine, beer, and liquor for 32 countries (Hagman, 1994). The attribute Wine consumption is color coded onto the display. Some of the European wine producers as Italy, Portugal, France and Spain are clustered together, indication similar consumption habits.*

The cluster visualization supports table lookup actions indirectly by grouping similar points near each other. If one valid point is found, other valid points are positioned nearby so further lookups of valid points are simplified. However, if the clusters don't have any form of labelling identifying each cluster, finding the first valid point can be difficult. Pattern detection and detecting similarities is directly supported by the grouping of similar objects. A cluster visualization can be initially difficult to interpret if users interpret it in terms of directions. Direction have no meaning in a cluster visualization, only the distance between points matters.

*Area visualization*

By associating each point directly to an area of the display, the result is an *area visualization*. A typical example of area visualization is a color coded geographic map, where each country is represented by a bounded area, colored by a chosen attribute.




*Area visualization , showing the number of networks in some european countries 1994.*

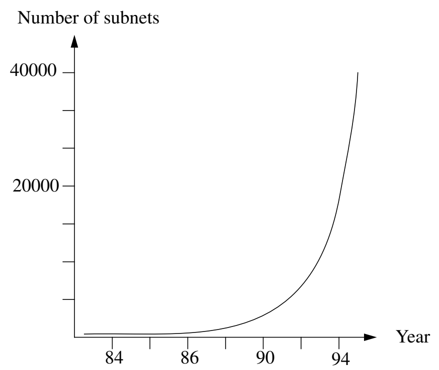
An area visualization representing a geographic map should not be confused with the map visualization, which only uses a map image to give context to the displayed points.

Another example of an area display is the circle diagram commonly used in popular statistics.

Area visualization often relies on the less accurate elementary perception tasks as area perception, color or shade, but the inaccuracy of these may be helped by a natural basic area representation as a geographical area map.

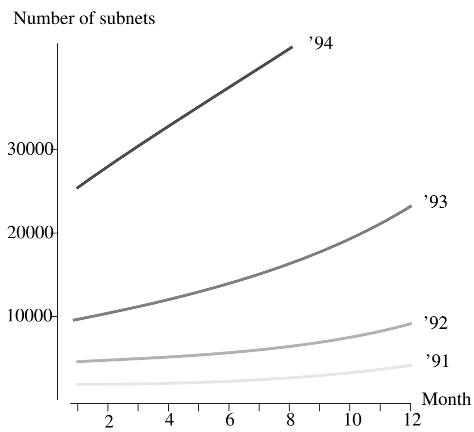
*Curve visualization*
If the data set is produced by a smooth mathematical function or any other procedure which creates continuous data, the set of points could be interpreted as points on a curve.

By extrapolating (often by simply connecting the points with lines) this curve from the data set, the result is a *curve visualization*.




*Growth of the Internet (Merit Network, 1994) The sampled points matches an exponential curve.*

Pattern detection in a curve visualization relies on the curve, angle and direction perceptual tasks while table lookup relies on the same tasks as in a scatter plot.



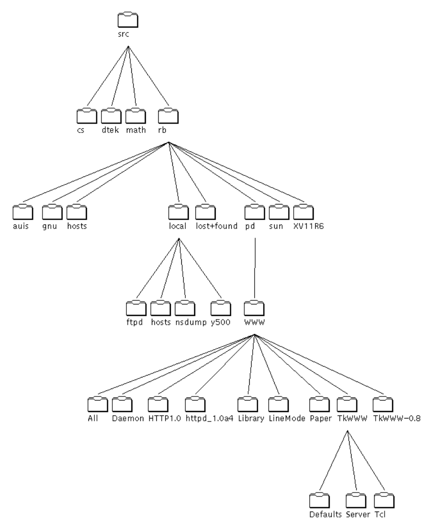

*Growth of the Internet (Merit Network, 1994) Each line represents one year.*

*Tree and graph visualizations*
Datasets can also consist of both a set of *n*-dimensional points M and a set of relations,

$$R = \{ p_i \otimes p_j \mid (p_i, p_j \in M) \}$$

Each relation $R_{ij} \in R$ can then be visualized by a connecting line between the points $p_i$ and $p_j$.

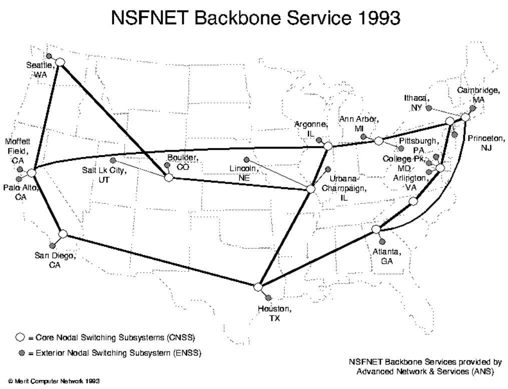
If the relation set is noncyclic, a possible visualization is a *tree visualization.*




*Tree visualization showing part of a UNIX filesystem, where only directories (tree nodes) are displayed.*

A tree visualization with many nodes uses much space, requiring a large visualization area.

The generic visualization of related data is the *graph visualization* showing the relations as connections between points.



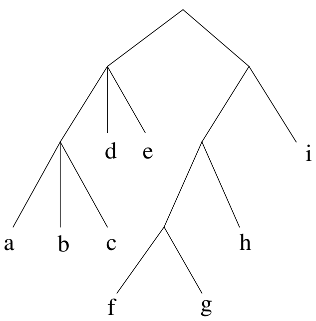

*Graph visualization showing the NFSnet backbone. The graph is superimposed on a line map representing USA to give context to the nodes and links. © Merit Computer Network 1993*

*Tree maps*

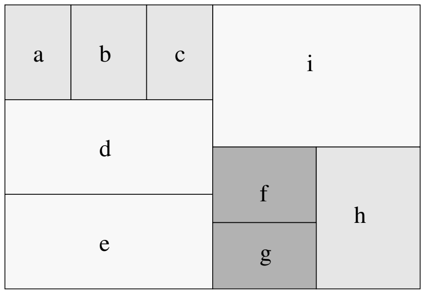
The tree map (Shneideman, 1992) is a 2D representation of a noncyclic hierarchical structure. The tree map visualization works by a simple recursive area-dividing algorithm:

1. For each of n branches from a node, divide the area in n subareas.
2. If the branch results in a leaf, draw information about the node in the subarea Else, if it is a node, repeat from 1 for each child with respective subarea.

The area subdivision may be into equal areas or into different sized areas depending on some weight function of the nodes.

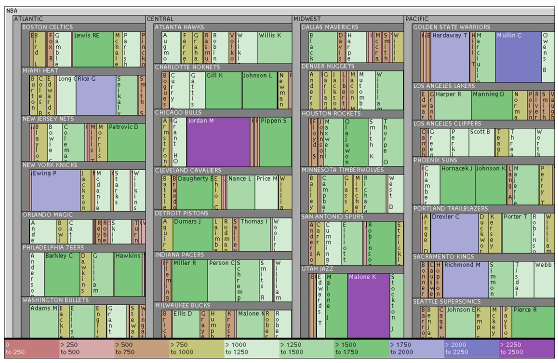



*A simple tree structure with nine leafs.*




*Corresponding tree map where shade is used to represent the depth of a node.*

The tree map visualization of a tree graph is an efficient way of displaying a large set of nodes in a small area while still allowing users to view individual relationships between nodes.



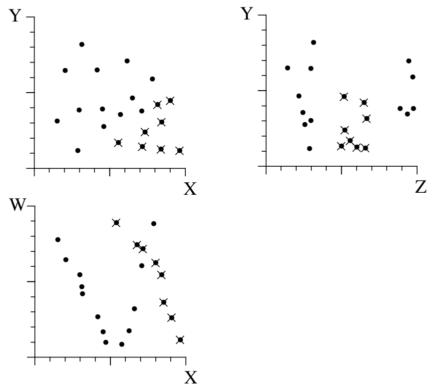
*Complex tree map showing a hierarchical structure.*

## Functionality of visualizations

Independent of the visualization type, separate manipulations can be applied on the visualization, for example zooming and navigating large visualization areas (Beard & Walker, 1990). The need for such functions arises if the visualization becomes cluttered, if different levels of details is wanted or if context between multiple visualizations is needed.

By *brushing* points (Cleveland, 1993, 1994) in a visualization users can select an interesting subset of the data. Brushing is typically indicated with highlighting or markers around points.

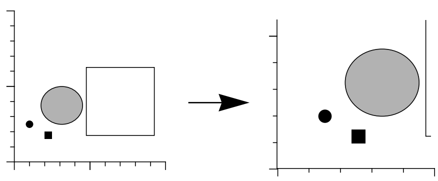
If multiple visualizations of the same data are used, *linking* (Cleveland, 1993, 1994) subsets of the data between visualizations is important for context and interpretation. Linking is the concept of connecting subsets by marking the selected points simultaneously in several visualizations and displaying visualizations aligned so that the connection becomes clear.




*Three linked scatter plot visualizations. The brushed points (marked with x) in the X-Y plot forms a clear group in the Z-Y plot and a line in the X-W plot.*

*Zooming* is a well-known function which can be applied on any of the described visualizations. The goal of zooming is to expand the geometric dimensions in the visualization, giving users a better view of selected data.

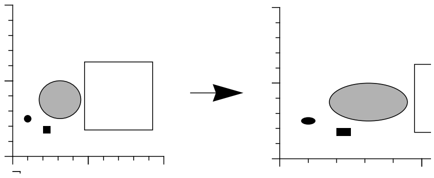
Zooming can be done with different methods (Jog & Shneiderman, 1994), for example with a *focal point*.



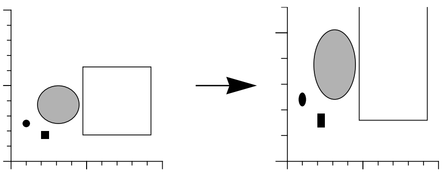
*Zooming around a focal point*

Zooming around a focal point uses this point as center for a geometric expansion in both geometric visualization dimensions, thus making objects around the focal point move away.

An alternative to focal point is the *focal line*, which uses a line as center of zooming.



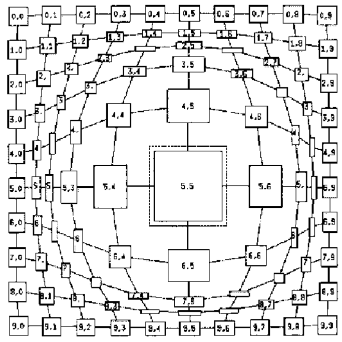
*Zooming with a vertical focal line*



*Zooming with a horizontal focal line*

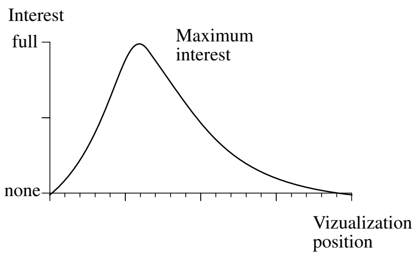
Zooming with a vertical or horizontal line only modifies one of the geometric visualization dimensions, keeping the other constant. The visualization appears to have a fixed line which doesn't move in one of the directions.

When zooming, the visualization geometry need not be uniform - some areas can be more expanded or shrunk than others. With a *fish-eye* zooming (Sarkar et al., 1994; Shaffer et al., 1994) one area of the display have a higher resolution than the others.

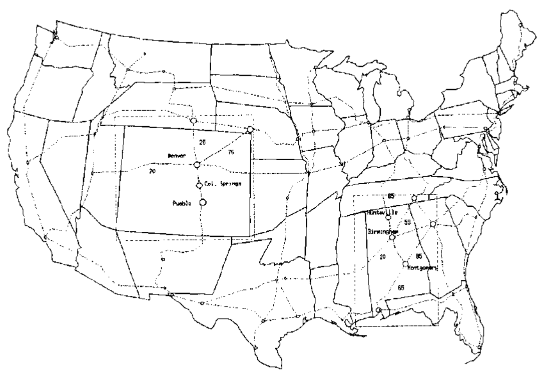


*A graph visualization with fish-eye zooming (Sarkar et al., 1993)*

The fish-eye zooming can be described by a degree of interest function, which shows how much resolution should be reserved for different parts of the display.

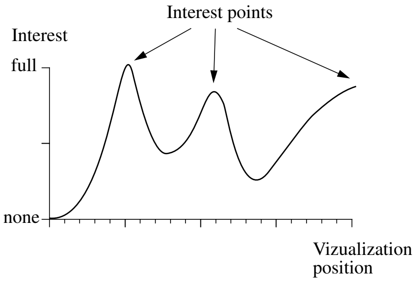



Fish-eye zooming can be extended to include *multiple focal areas,* each focal area functioning as a fish-eye.



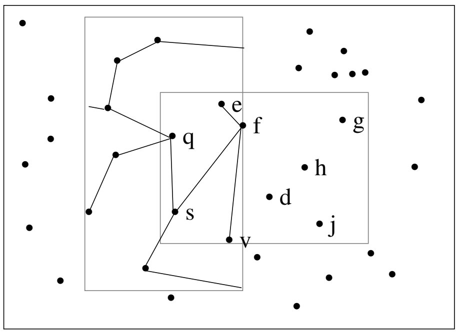
*A graph visualization of highways in the usa zoomed with a two focal areas near Denver and Birmingham (Sarkar et al., 1993).*

The degree of interest function then have multiple tops representing the focal points.



*Hierarchical* zooming (Bederson & Hollan, 1994) is a combination method based on a geometric zooming such as focal point zooming. When the geometric zooming progresses, each displayed object changes appearance to better use the available visualization area. The details of the objects are progressively refined, showing more and more information. This method helps keeping the visualization area uncluttered, and only showing details when necessary.

*Filtering* is a method for excluding or including information in the visualization on a user demand basis. Filtering works as logical and local zooming, allowing users to select an area of interest and displaying extra information in only that area. It can be implemented as rectangular filters which can be moved over the visualization area ( Bier et al., 1994). Filters can then be allowed to overlap, creating subareas with combinations of the filters.




*Two filters superimposed on a graph, one filter hiding names of points, the other hiding links between points. The filters have partly overlapping holes, letting information show through.*

## Further information encoding

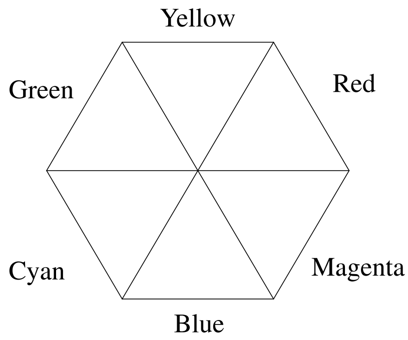
In addition to the information given by the data points from their placement in the visualization space, more information may be coded to each data point by using *color*, *shade*, *shape*, *backgrounds* or even *sound*. This extra encoding of data is needed since the visualization dimensions often are a subset of the data space. Traditional visualizations are usually only 2- or 3-dimensional representations of the data space and thus need additional information encoding separate from the visualization dimensions.

*Color* can be used as information agent as the human eye is capable of distinguishing a large number of hues. There are basically two different ways of using color, *continuous* and *discrete* color scales.

The continuous color scale assigns different color levels as a smooth function from the data dimension to be displayed. A continuous color scale can be used to display a quantitative attribute in the dataset.

A discrete color scale could be used to encode a set of categorical values, mapping each value to a distinct color. To provide good distinction between categories, the colors should be chosen to maximize the visual differences. According to (Cleveland 1994), a good set of colors for a discrete color scale are cyan, magenta, yellow, green, orange and light blue.

Color can specified by using a *color space* (Foley et al, 1990). A color space describes each color as a point in a coordinate system. Examples of color spaces are HSV, which describes color by hue, saturation and value. Hue is an angle representing the color in the color circle, saturation is the pureness of the color - full saturation meaning a pure color, zero saturation meaning no color at all. Value in HSV space is the lightness - zero meaning black, one meaning full color.




*The HLS color space hues*

The HSV space is used because of it's good relation with human's experience of color. Problems with HSV is, for example, the nonlinearity of experienced color brightness.

The RGB color space is often used in the basic display hardware, since any color can be produced by additative mixing of the components red, green, and blue. The connection between mixed colors and experienced color can be complex, making the RGB space somewhat difficult to use.

Natural language can also be used to define colors and shades of color. For example, the user may specify "greenish-blue" or "dark-red" instead of giving a numeric specifiaction of the color.

Other color spaces (Foley et al., 1990) attempts to combine experienced color and color space coordinates in a way to make them easy to use.

*Shading* can be used to display quantitative attribute by using different levels of a color to points. A shade can be built up by monochrome rasters, creating the experience of different levels of shade. Shading is most often used with gray color scales, suitable for monochrome display devices as printers or black and white monitors.

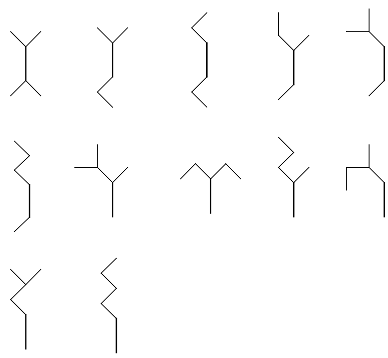
*A set of gray shades forming a scale*

Different *shapes* or *symbols* can be used to separate points with different categorical values. Cleveland (Cleveland, 1994) suggested these symbols for use in plotting values.

*Basic symbols*

The symbol are ranked from the most preferred to the left, to the least preferred to the right. The ranking was done based on experiments showing which pair of symbols which were easiest distinguishable from each other. When superimposing symbols it should still be clear which symbols are displayed.

The symbols can be extended to allow encoding of values in one type of symbols. A stick-figure icon family (Picket & Grinstein, 1988) can be designed, having different members encoding different values.



*Stick-figure icon family*

In this icon family values are encoded into the placement and direction of limbs extruding from the main line.

If the placement of points in a visualization is not clear, can a *background* picture be used to give context to the points. The background can be raster lines in an abstract visualization or a map if the visualization has a natural representation. A background doesn't encode any extra information in the points, but helps users orientate in the visualization.

*Sound* can used to identify selection of points or as a marker for cursor position in the visualization area, giving users additional feedback. Feedback via sound can also be given as queries dynamically change the selected elements (Bly, 1981).

## Visualization methods in the context of database query types

When designing a database query system using a data visualization method, such as one of the methods desrcibed above, it is important to understand the effects a given visualization method has on the users perception and understanding of the results from different types of queries. Different visualization methods can be more or less effective for different types of queries.

In this paper a distinction is made of the query types:
- Finding specific elements.
- Finding trends.
- Finding anomalies.
- Finding similarities.

In the following evaluation, three of the mentioned visulization methods are implemented in a dynamic queries system and the users perceptions of the query results, depending on vizualization method, are evaluated.

*Visualization designs in the context of a dynamic queries system*

When using a visualization in a dynamic queries system, some important design criteria should be considered.

1. *Compact data display.*
   A database may have thousands, or even millions of separate points. For completeness, the visualization must be able to display all of them. Also, all points should be directly available for manipulation.
2. *Smooth and fast redrawing.*
   Smooth and fast redrawing is essential for interactive visualizations since it gives users better context. When rapidly searching large data sets (as when looking for trends) it is important that the manipulation never ”drags behind” the users intentions.
3. *Separate data points easily distinguished.*
   If users should be able to select or identify a distinct point/points (as when looking for specific elements), the points must be visible (Picket & Grinstein, 1988).
4. *Marked subsets easily distinguished*
   In linking multiple visualizations, users should be able to visually connect the subsets between visualizations
5. *The context of any subset should be well defined.*
   Good context aid users exploration of data (Mackinlay et al., 1991; Bederson & Hollan, 1994), and also help identifying anomalies/outliers.
6. *Trends and correlations between attributes should be made visible.*
   Direct visual connection between attributes (as in a coordinate system) helps users to find trends and orientate in the visualization.
7. *Subsets connected in the data space should be connected in the visualization space too.*
   Wen looking for similarities, the visualization should aid users to identify similar subsets.

### Evaluated visualizations

In this study, three visualizations are implemented and evaluated in an empirical study:

- Scatter plot
- Map visualization
- Cluster visualization

The scatter plot was chosen because of the general way it displays multidimensional data. Depending of which pair of attributes are mapped to the display, many different kinds of relations can be viewed. The use of scatter plot is also common in many statistical utilities.

The map visualization is a special case of the scatter plot, where points is placed in a well-known geographical context. The added context from the background map makes this visualization familiar to most users.

The cluster visualization represents a completely different way of positioning multidimensional data in a visualization area.

All three visualization permits zooming and encoding points with color or shape.

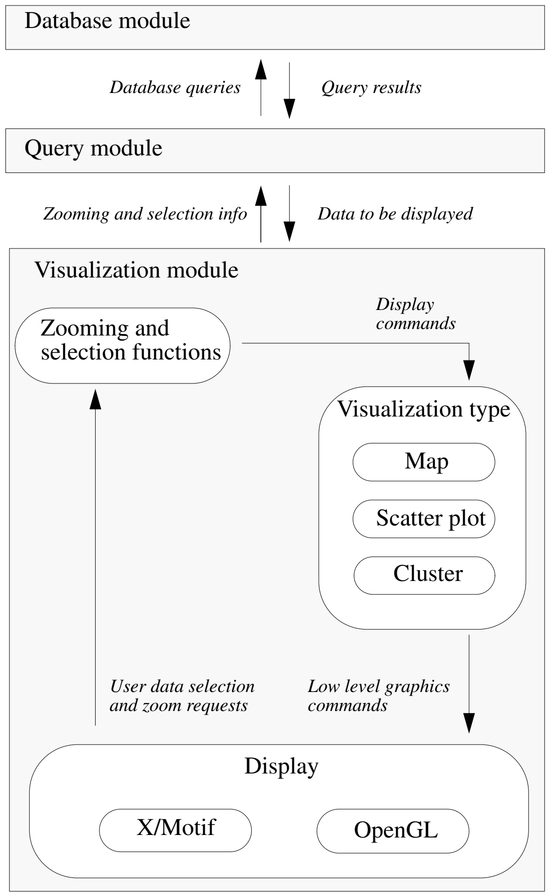
### Visualization implementation

#### Host system

The visualization program, *Information Visualization & Exploration Environment* (IVEE) (Ahlberg, 1994)*, is written in C++, running under SunOS 5.3 (Solaris 2.3) and IRIX 5.2, for the X Windows System. The interface is based on an extended Motif widget set. The extension are mainly custom widgets for dynamic queries and visualization widgets supporting different methods for data display.

#### Supporting database system

All visualizations evaluated in this paper are used as modules in the framework of IVEE, a dynamic queries system which allows users to browse generic database relations. IVEE is composed of three components, a database component, a query component and a visualization component. The database component stores all data and retrieves mathing elements given a database query. The query component is the user interface which translates user queries via widgets to database queries. The visualization component is continuously sent the set of selected elements at each user interaction and display the these elements, allowing users to manipulate the visualization via zooming and color settings.



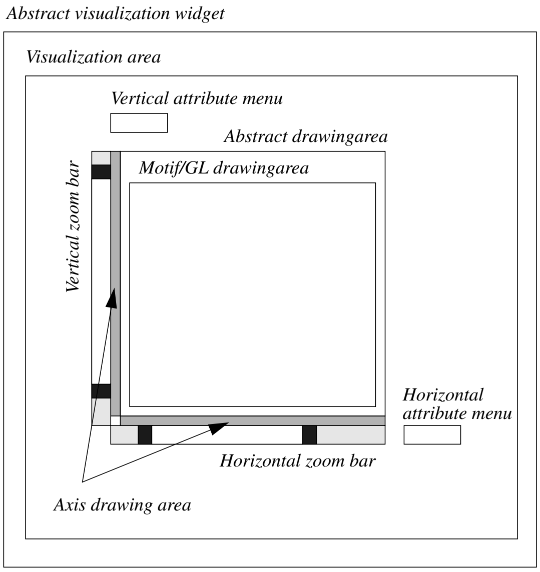

*Information flow in the IVE system*

#### The visualization widget

The visualization widget is composed of a hierarchy of widgets handling different aspects of the data visualization. These aspects are for example scaling to fit the points in the physical display area, color selection for numeric ranges, point size,

* *IVEE is availiable via anonymous ftp ftp.cs.chalmer.se, /pub/users/ahlberg/IVE/*

possible axis display and interface to the actual X/GL drawing routines. Selection of attributes for the visualization axes is also done within the visualization module.




*Visualization widget hierarchy*

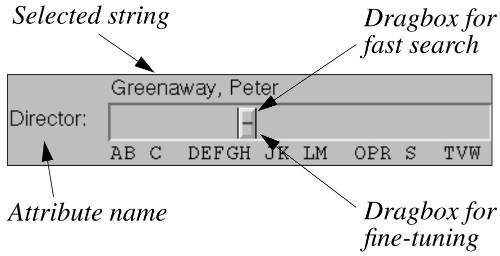
The visualization widget has a set of basic commands (or methods) for manipulating the display:

- Add a set of points
- Remove a set of points
- Redraw all points
- Set zoom level

## The Query module

The query module consist of a set of query devices, each device corresponding to one of the attributes in the database. The query devices acts as filters, removing unwanted points ( Ahlberg et al., 1992, Shneiderman & Ahlberg, 1995). The results from all query devices are AND'ed together, producing the set of points to be displayed.

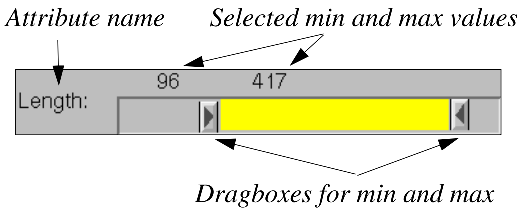
The *alphaslider* (Shneiderman & Ahlberg, 1994) allows users to select from a large set of strings. The string are ordered alphabetically and users can quickly sweep of the complete range.




*Alphaslider for chosing director names in the movie database*

The alphaslider supports two search modes, one fast mode for quick movement to a alphabetical range and a fine-tuning mode for selecting individual strings.

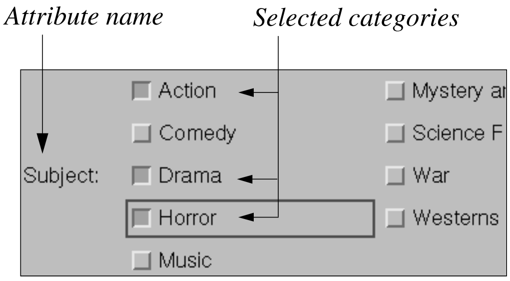
The *rangeslider* allows users to select a maximum and an minimum value in a numeric range.




*Rangeslider for selecting minimum and maximum length of movies in the movie database*

Rangesliders continuously display the selected range, both as graphical bar and as text output.

The *category selector* allows users to select one or many categories of a categorical attribute. Both the alphaslider and the category selector selects categorical values, but the alphaslider defines an ordering of the values (alphabetical order) while the category selecter has no order. The category selector allows users to select multiple values, making a query composed of the union of selected values.




*Category selector for selecting movie categories in the movie database. Action, Drama and Horror are selected.*

A special case when selecting values with the category selector is when no category is selected. This state is interpreted as all categories are selected. The design was done do make it faster to select single values. It is also a design used in mixer-tables for sound mixing.

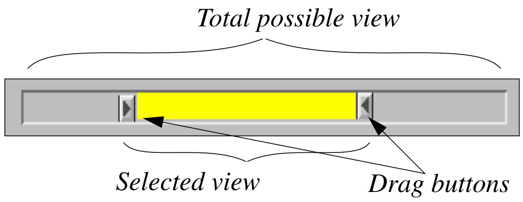
## Display attributes

All of the visualization types (scatter plot, map, cluster) share a common set of attributes:

- *Color type*
  The points in the visualization can be colored either in a discrete or a continuous scale depending on the type of color attribute.
- *Color attribute*
  The point color can be selected from any attribute in the database, both from categorical and continuous attributes.
- *Label attribute*
  A text label from any database attribute can be shown near the displayed point if there is enough space.
- *Color range*
  Continuous color ranges can be selected by users, as interpolations between two selected colors, or from a preset set of colors.
- *Point size*
  Point can be displayed in three different sizes: small, middle and large.

## Zooming

Each visualization is handled by a generic display object which handles zooming, panning and selection mechanisms. Zooming and panning are handled through the use of *zoom bars* (Jog & Shneiderman, 1994),




*Horizontal zoom bar*

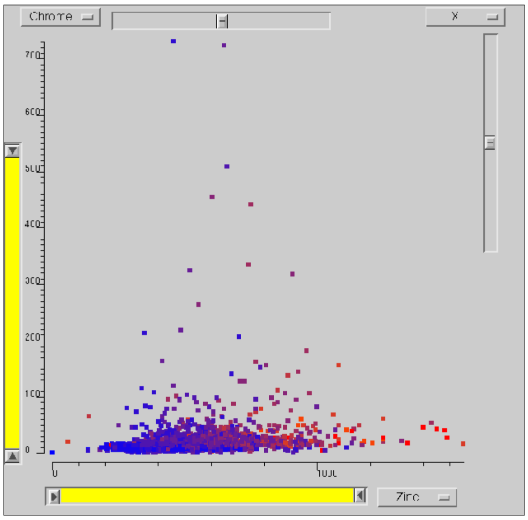
placed below and to the left of the visualization area. Zooming is also possible through pressing the middle mouse button for zooming in around the pointer and pressing the right button for zooming out. Zooming with the zoom bars uses the concept of *focal lines* while mouse button zoom uses a *focal point*, both concepts described earlier.

To achieve smooth zooming while redrawing more than 1,000 points, a doublebuffering functionality of the drawingarea widget was used.

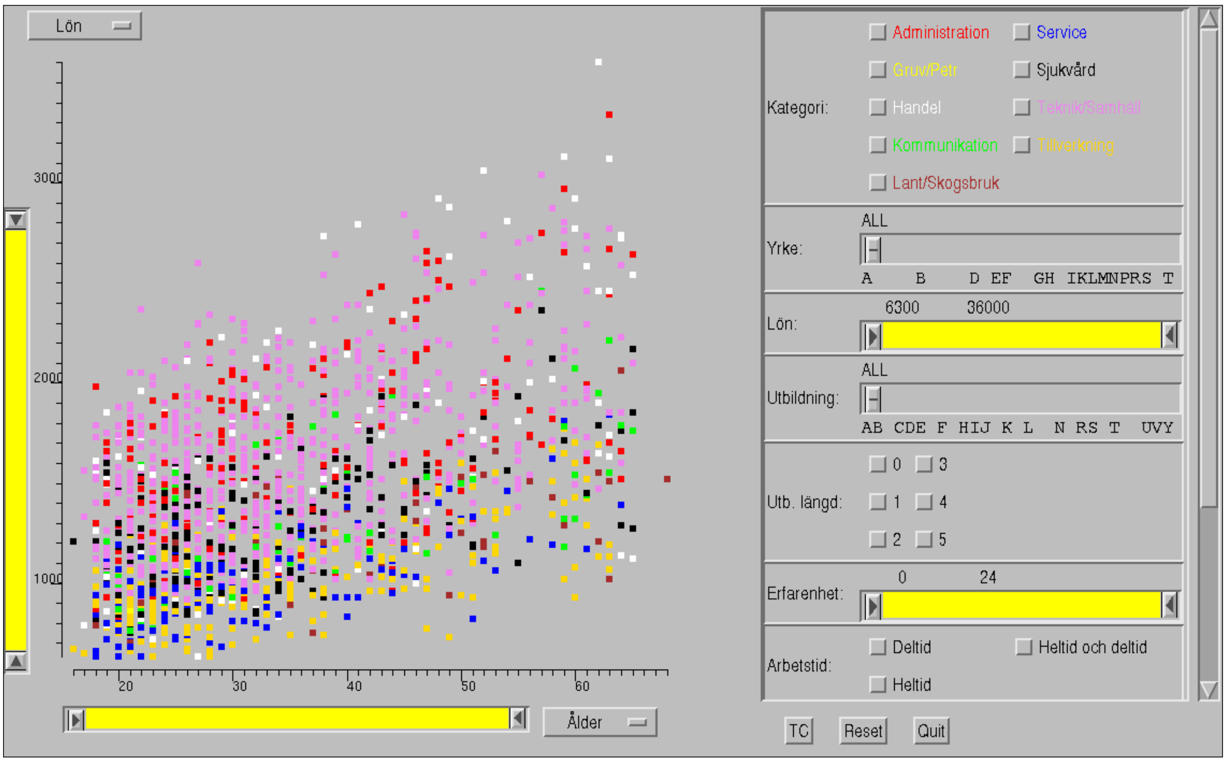
## Scatter plots

The basic visualization module is the scatter plot module. By mapping two of the database attributes to the visualization axes every point in the database can be translated into a physical display point. When updates of which points are to be displayed (add/remove points) are sent to the scatter plot some time-saving actions such as no redrawing of unmodified points are done. When all points are redrawn double-buffering is used to minimize flickering.

A labeling system is implemented which can display a text label near each point when there is enough space. The label positioning algorithm tries to minimize overlapping between labels and points by using a set of occupied areas, continuously updated.



*Scatter plot of zinc vs chrome values from an environmental database The database contains metal pollution data in Sweden 1975 - 1985.*



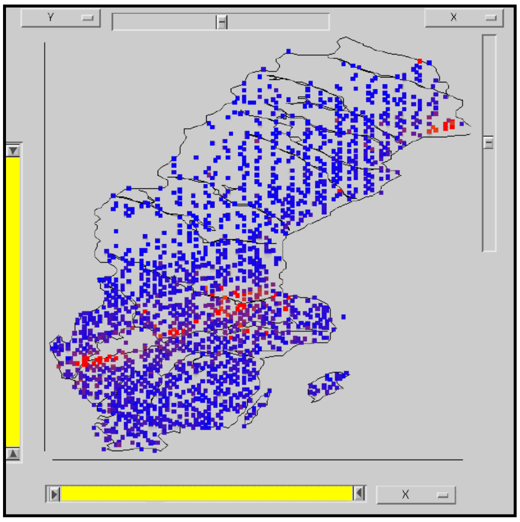

*Scatter plot visualization showing work applicants in Göteborg,. The X axis represents the age and the Y axis represents the requested salary. The points is colored by profession category.*

## Map

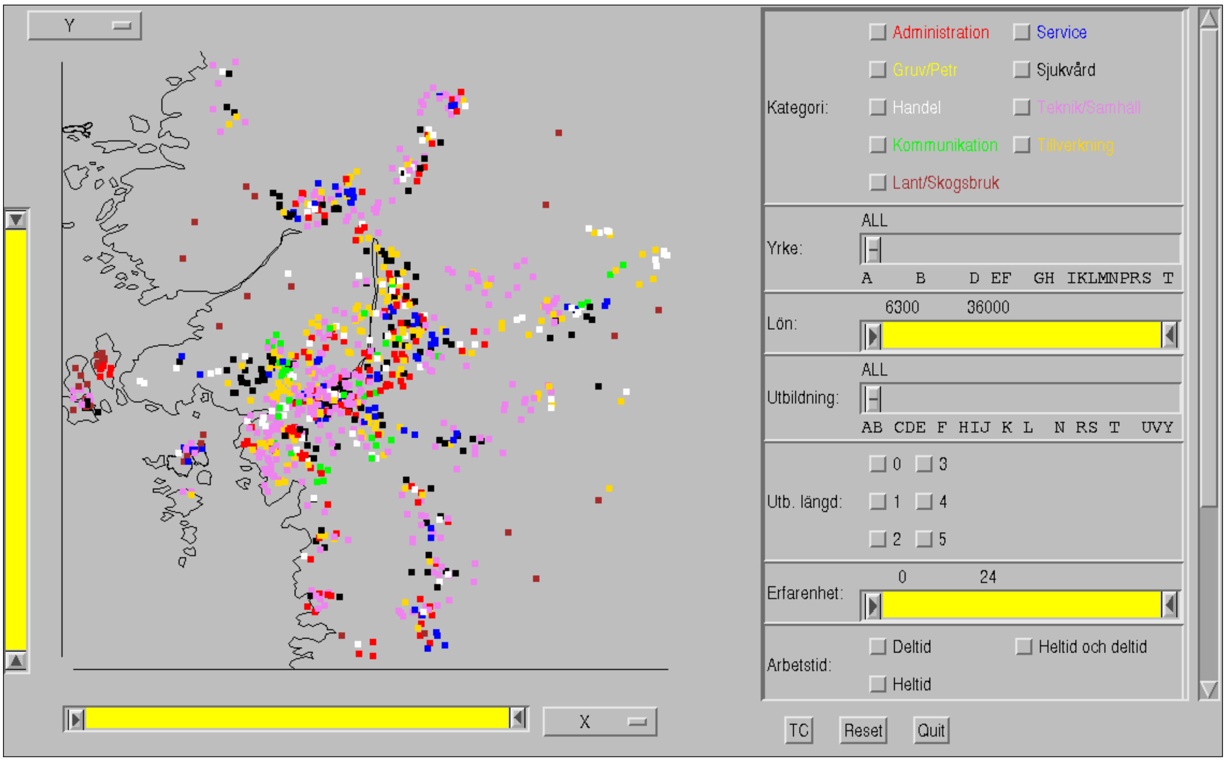
The map visualization is basically a scatter plot visualization with a background vector image. The vector image is automatically read when users select a pair of attributes which have a corresponding map file. The image is described by a text file with a set of lines in the same coordinate system as the selected attributes.

The vector map is draw as a background before any points in an ”redraw all” request. When adding or removing single elements refresh of the map is delayed until the user ends the current query.

The vector map could be used both as a geographical map or as any other image which can be defined by line segments, e.g. parts of a linked graph or separators between subsets.




*Map visualization, created with IVEE, from the environmental database. The map displays Sweden with major lakes and rivers. The light points have high chrome values.*

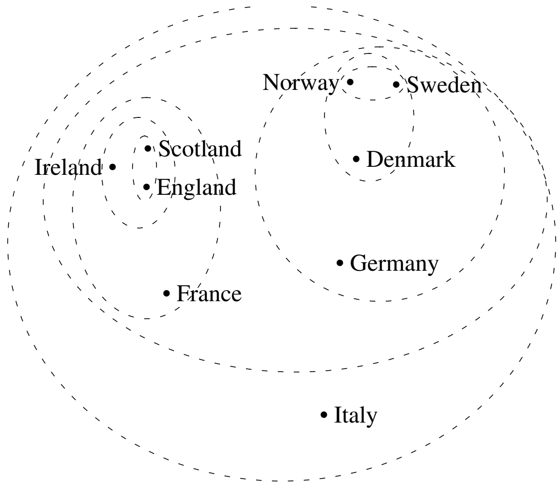



*Map visualization showing work applicants in Gothenburg. Each points is positioned at the applicant’s home address and the map shows the surroundings of Gothenburg (the coastline and parts of the archipelago is shown).*

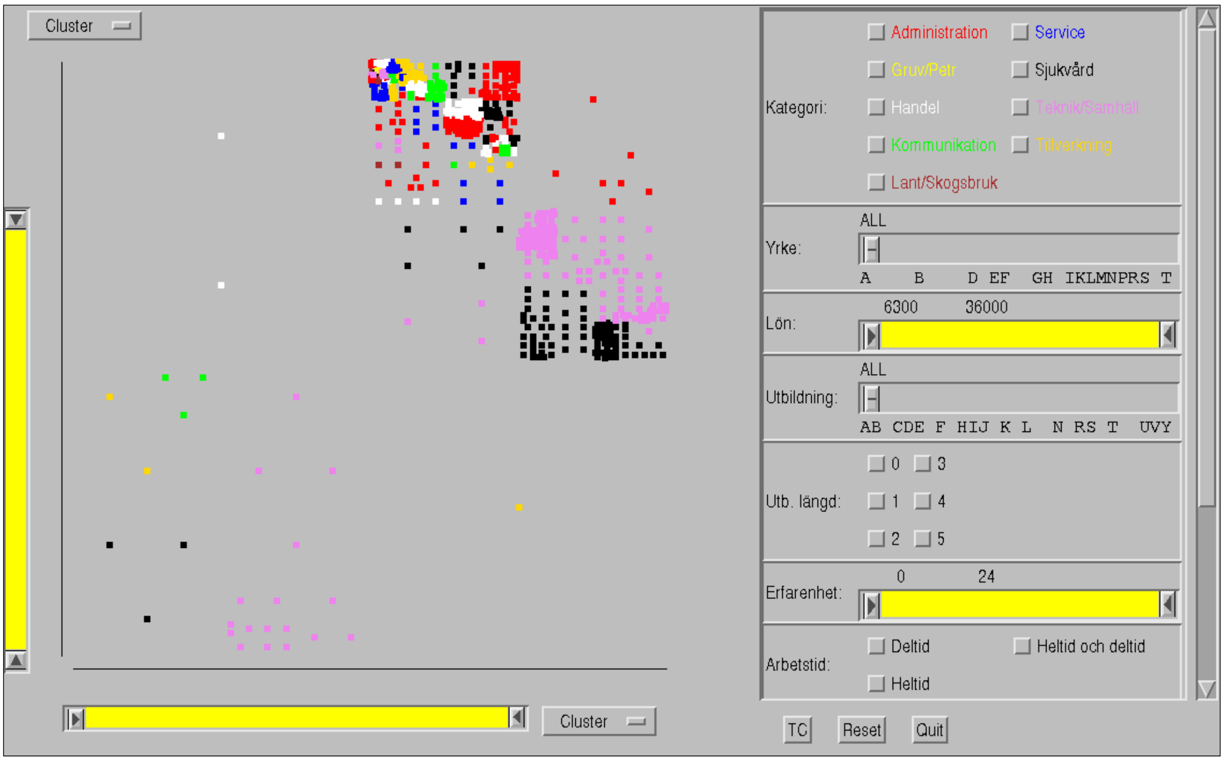
## Cluster visualization

The cluster visualization is implemented as a preprocessing of the data for the scatter plot. The cluster module first processes a dataset into a dendrogram (see next page) using a agglomerative clustering algorithm (Bobick, 1987; Murtagh, 1983). When clustering on string attributes, only equality is tested for, i.e. the distance between two string is either zero or a marker for maximum distance.

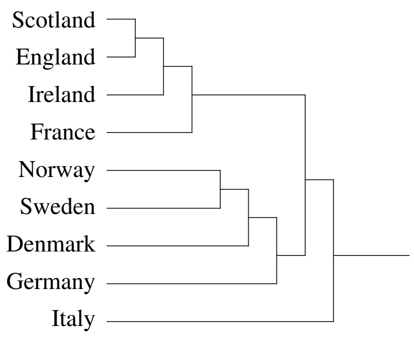
The algorithm used for converting the dendrogram to 2D coordinates uses the tree map concept (Shneiderman, 1992). In principle, a tree map of the dendrogram is created, but no areas are created for the leafs (single point clusters), instead the leafs are positioned in the middle of the parent node area (see next page).



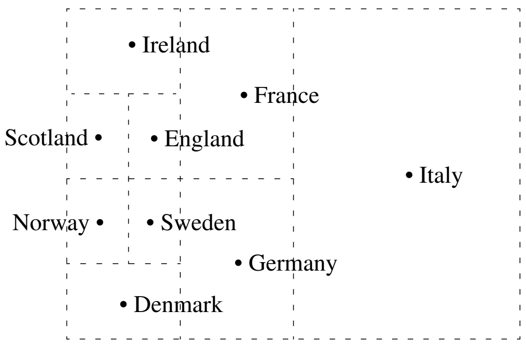

*Some european countries represented as points positioned geographically, and a hierarchical clustering on distance between points.*




*Cluster visualization, showing work applicants in Gothenburg. The points are clustered on all attributes - points with similar values are positioned near each other.*




*Dendrogram of the above european countries.*




*Cluster layout with tree maps. Note the similarity between this layout and the original map. The positions have basically only been rotated 90$^o$.*

### Comparisons to other systems

The evaluated visualizations all are 2D representations of a dataset. Other tools such as XGobi (XGobi, 1994) allows users to select similar visualizations. XGobi also allows 3D scatter plot visualizations.

The cluster visualization can be performed in any system which supports scatter plots, by precalculating the coordinates representing the clustering positions, and then using these coordinates in a traditional scatter plot.

Pad++ is a zooming display system for hierarchical data. The zooming function in Pad++ is similar to the one used in the study, but extended with a more general system for displaying objects at different zoom levels.

## Experiment

The study was performed with a database from the Gothenburg employment service, describing people looking for employment in the Gothenburg area. The questions in the study were designed after interviews with representants from the Gothenburg employment service.

### Apparatus

A Sun SparcStation10 with a 8-bit color monitor and an optical mouse was used for the study. The sessions with the participants were recorded with a tape recorder. Each session lasted for about an hour.

### Interfaces

The evaluated interfaces all used the zooming and query mechanisms described above.

Color coding was used in all interfaces to show profession category. The database described 1,611 applicants in the Gothenburg area with 13 attributes:

- Category (9 categories)
  The main category (AMS, 1983) of the applicant's desired profession
- Profession (ca 3,000 strings)
  The applicant's desired profession, as coded by the swedish employment service (AMS, 1983)
- Salary (numeric 6,300 - 30,000)
  The applicant's desired salary
- Education (ca 100 strings)
  Type of applicant's education
- Length of education (numeric 0 - 25)
- Working hours (3 categories)
- Sex (2 items)
- Name (1611 strings)
- Age (numeric 16 - 65)
- Reference (2 categories)
- X and Y
  Geographic position

Parts of the database were obtained from the Gothenburg employment service (Category, Profession, Education, Length of education, working hours, sex and name). The remaining attributes were added manually (Salary, Name, Reference, X and Y)

### Scatter plot

The scatter plot interface showed each applicant as a point in the 2D coordinate system spanned by the attributes Age and Salary. Age was placed on the horizontal axis and salary on the vertical axis. The age attribute ranged from 16 years to 65, and the salary attribute from 6,300 to 36,000 SKR per month (about $970 to $5,500 per month). The visualization axes were used to display attribute marks.

### Map

The map interface showed each applicant as a point in the 2D coordinate system spanned by the X and Y attributes (geographical position). Additionally, a vector map showing the outline of Gothenburg was superimposed. Since the X and Y coordinates used in the database didn't correlate directly to any standard metric, the visualization axes were not used to display any numeric information.

### Cluster display

The cluster interface showed each applicant as a point in non-directional 2D plane. Since no specific attributes matched any of the axes, the axes were not used to display any numeric information.

### Types of questions

Each of the visualization types were tested with the following types of questions, matching the database query types previously mentioned:

1. Finding similarities between subsets.
2. Detecting trends.
3. Detecting anomalies.
4. Finding specific subsets.

### Expected results

Since the cluster display algorithm works by placing similar points near each other, it was expected that this display type should be helpful when looking for similarities between subsets. Additionally, any anomaly was expected to be easily distinguished since it's position is far away from the main clusters. However, when using the cluster display to look for specific subsets, an inexperienced user may ”get lost” in the visualization space because of the directionlessness of cluster displays.

The scatter plot was expected to give a good sense of confidence in detecting trends or anomalies in any of the plot's axes attributes, since the direct visualization of this attribute makes it unnecessary to make any widget queries on the visualized attribute.

The map display adds only geographic position to the data points, so questions concerning relation between non-geographical data were expected to be somewhat harder to answer than with the cluster display or scatter plot. The added information from the background map and the well-known display type could help users orientate themselves in the visualization space, thus easier finding specific information.

### Variables

The independent variable in the evaluation was the visualization type:

- Scatter plot
- Map
- Cluster

The dependented variables were the participants subjective and objective query results.

### Participants

Eight users participated in the study. Four of the participants were employees of the Gothenburg employment agency and thus had very good knowledge of the data set. The other four participants (university students) had no previous experience with the data set.

All of the participants had used at least one computer system and a mouse. Four of the participants were male, four female.

## Procedure

A within-subjects design were used. The visualization methods were evaluated by ”talk aloud” sessions (Allwood & Wikström, 1986, Ericsson & Simon, 1983) with the participants, that is the participants were continuously describing their actions while performing tasks (see appendix for full set of questions). Example tasks were:

”Find all applicants with five years educations and with more than 20,000 SKR as desired salary”
or
”Find one or more common factors between the categories Health care and Administation”.

Each session were taped and later transcribed.

The participants were given written description of the study and a question form about their background. After reading and filling in the form, they were given a verbal and visual introduction to the system. Any questions about the visualization methods or the database were encouraged.

The participants were then given a few test questions, which they tried to solve with the system.

The main tasks were composed of three sections, one for each visuzalization method (map, scatter plot, cluster) Each section was composed of four questions, each one representing one of the query types (similarities, trends, anomalies, subsets). Each section was answered with the help of one of the visualization methods (map, scatter plot and cluster display). The participants were continuously describing their actions and were encouraged to ask questions or comment on difficulties. After each section the participants completed a question form about their impression of the visualization method. After the last section a form describing general impressions, including a ranking of the visualization methods, were completed by the participants.

## Results

*General user preferences*
Notable, seven of the eight participants made identical rankings of the visualization methods. From the most preferred to the least:

- Scatter plot
- Map
- Cluster display

The eight participant made the ranking: Cluster diagram, Scatter plot and Map - from the most preferred to the least.
These rankings were the general impressions of all methods, without any specific opinions about functionality in different query situations. When compared to the actual results of the participants, this ranking matches well. The results from the cluster display were notably less correct than the results from the scatter plot and map.

Seven of eight participants classified the session with the system as ”enjoyable”. No one found the system ”difficult” to understand. Four of eight found it ”easy”. Some participants had minor problems with the query interface, most notable an initial motorical problem concerning string selection with the alphasliders. Four participants had an initial conceptual problem with the category widgets, not realizing the mixer-table-functionality of the buttons, thus confusing all categories selected and one category selected. All of the participants who had problems with the category widgets said, after explanation, that they ”now understood” the category widget function. However, their later use of these widgets sometimes indicated misunderstanding of these. This may have affected the participants methods of detection similarities and anomalies in the test questions, since these types of questions often rely on comparisons between categories.

*Zooming*
The zooming function were seldom used, four of the participants did not used it at all, and they only used it in one or two of the questions. The zooming were mainly used to ”unclutter” the display if several points were positioned very near each other and it thus were hard to distinguish individual points. As a selection mechanism, zooming were only used twice, then in the cluster visualization to select a subcluster ”looking interesting”.

The cluster visualization got several comments about ”being cluttered”, but only two of the participants used the zooming functions to unclutter the cluster visualization. A possible explanation of this is the directionlessness of the cluster visualization, making the participants unsure of which areas in the visualizations were important or unimportant. Since zooming removes some areas from the direct display, something could be missed.

Zooming in the map visualization was only done by one of the participants to unclutter the display. A reason why zooming not were used more could be that the contents of the test questions never used geographical location as a search criteria, thus not making it necessary to use zooming as a selection mechanism.

In the scatter plot visualization, zooming were used both as a uncluttering and as a selection mechanism. Uncluttering via zooming could fail if the area of interers were composed of points having exacly matching coordinates, the points would still overlap at full zoom.

When zooming was used it was never used to zoom in on the points of interest only - a surrounding area with points were left as context.

*Results depending on query types*

*1. Finding similarities between subsets.*
In these test questions, participants were asked to compare subsets of the database which each other and find relations between subsets. As the profession categories were used as subsets and the color coding also were done by profession category, most participants had no problem distinguishing the separate subsets. The point positions in both the map and scatter plot visualization led participants to search for similarities in the axes directions - geographical position for the map visualization and salary/age relationships for the scatter plot. In the cluster visualization, two participants asked ”Should I take groups near each other, then?”. According to the cluster algorithm, this would be a good idea, but presented problems in the actual decision of which criteria had formed the clusters, since no such clues were visible.

When the similarity between subsets were overlapping ranges in one of the attributes (salary), the participants could see this relation directly in the scatter plot, while they had to manually query the database in the map visualization, leading to some confusion about the correctness of their answers. However, no actual difference between answers were found when later checking the protocols.

Other criteria, such as number of points in subsets were hard to decide if the visualization had many overlapping points. The cluster visualization had worst results since the algorithm’s goal is to group points together. The map visualization had similar problems since the position of points represented people in a town, having several natural dense-populated areas.

General for all visualizations were a tendency for those who had problems with the category widgets to select one subset at a time, remembering it, selecting a new subset and comparing this to the remembered picture.

*2. Detecting trends.*
This query type was considered the hardest one to answer, on several occasions leading to incorrect answers and one complete failure ”I give up”. The failures were most often present in the cluster and map visualizations, probably because the trends didn’t have any relations to the visualization axes. In the scatter plot, trends directly related to any of the axes were more easily detected. ”You see it from the slope” as one participant said when finding an age/salary trend
The participants had to do several queries to get the same information from the cluster and map visualizations. Some used the query devices interactively, sweeping over ranges in an attribute, keeping other attributes constant, and simultaneously checking the rate of point appearance/disappearance, thus finding ”how many were affected”. Other participants set up several small ranges and tried to remember a ”visual pattern” of the result. Then, they changed some of the ranges and compared to their remembered pattern. The first variant, rapid sweeping, was noted as giving more confident answers.

*3. Detecting anomalies.*
Finding anomalies related to one or two attributes were straightforward in the scatter plot when these attributes corresponded to the axes. The participants could directly see the points lying at the extremes of the display area or outside the main group. The map visualization required methods much like methods for detecting trends. Notable, immediately detected anomalies in the scatter plot were mostly points lying near the edges of the display, while anomalies in the map visualization were found in recognizable geographical areas such as islands or city regions, showing how the participants used the extra information in the map visualization in the search for anomalies.

*4. Finding specific subsets.*
Almost all participants found searching for specific elements easy using any visualization method. The problems that appeared, mostly depended on problems with the query devices and not with the visualization. The probable reason for the absence of problems is that a specific query narrows the query result to sufficiently small set and removes the need of any more information extracted from the visualization. The participants just set up the query devices, noted there were few points left in the display, and considered the query complete. This could mean that some of the remaining points were incorrectly described as matches. The scatter plot was somewhat more immune to this error since incorrect points were easily distinguished as erroneous due to their position in the display. The cluster visualization should have showed a similar property, but most participants did not realize the importance of points being far away (thus different) and just accepted all points as valid.

*Query type vs Visualization method results*
The particpant’s results depending on the visualization method and query type can be displayed in table form. The result are ranked from -- to ++, where -- is complete failure, = is neutral, and ++ is complete success. The participants are displayed as xx/yy where xx is the participants subjective results (i.e. if they thought they had answered correctly) and yy is the objective result.

| | *Scatter plot* | *Map* | *Cluster* |
|---|---|---|---|
| Finding similarities | ++/+ | +/+ | =/- |
| Detecting trends. | ++/+ | +/= | =/- |
| Detecting anomalies. | ++/+ | +/= | =/- |
| Finding specific subsets | ++/++ | ++/+ | +/+ |

*Data exploration*
During the study several of the participants ”strayed of the path” and started looking for other things than original specified in the question. Comments like ”What if I do this?” were common in both the map and scatter plot visualization. A difference in the data exploration were noted between the map and scatter plot. In the map visualization participants tended to note subareas with some property they found notable, such as ”Only Health care on Hönö? - Funny!” (Hönö - island outside Gothenburg). In the scatter plot the questions were more like ”What if I check only women here...” - that is, believing the data displayed to be true and possible to explore further - which it wasn’t in this case.

### Discussion

The preference of scatter plots before maps can be explained by the contents of the questions. No questions asked specifically about geographic trends or similarities, which could have been found in the map visualization. Some questions asked specifically about properties related to the salary and age attributes - directly displayed in the scatter plot, thus making this visualization effective for answering those questions.

The cluster visualization was considered hard to understand. Some of the participants tried to identify clusters by randomly selecting elements in them. This identification method failed because of the complexity of the clusters (points in the same cluster can have different values in several attributes while still having a common ”mean” value). When not getting a complete understanding of the clusters, the participants resolved to use it as a non-specific point arrangment - not using either position or distance.

## Conclusion

A number of visualization methods (scatter plots, maps, area visualizations, cluster visualizations and different forms of graphs) are discussed in terms of query types and perceptual tasks. Operations on visualizations such as zooming or filtering are described.

Three visualization methods (scatter plot, maps and clustering) are implemented in modular dynamic query system, IVEE. A hierachical abstraction of the visualization widgets were used to simplify the communication with the datbase and query modlues, and proved to be successful.

IVEE were used to evaluate the three visualization methods in an empirical study.

The evaluation suggests several points to consider when selecting a visualization to a dynamic query system.

The query types can be divided into *finding specific subsets, finding similarities between subsets, detecting trends* or *detecting anomalies.*

*Finding specific subsets* can be done with any of the visualization methods as long it results in a uncluttered display. The cluster display often becomes cluttered, but can be zoomed to be more clear.

*Detecting trends* are most easily done using a scatter plot with axes related to the query.

*Finding similarities between subsets* can be done with a scatter plot or a map. Maps supports more general geometric pattern recognition than scatter plot. The scatter plot can be of good help in detecting similarities if these have simple relations to the axes.

*Anomalies* can be detected both with a scatter plot and a map, possible more easily with a scatter plot if the axes match the related attribute. Otherwise a map, giving more context information is preferred.

## Further work

- The cluster visualization needs to have easier identification of clusters to be usable. In the study clustering was done with equal weight of all attributes. These weight should be possible to be user defined, possible removing some attributes from the clustering altogether. An identification system, labelling each cluster with the ”most identifying criteria” will also be useful.
- As zooming were used so little in the study, the question arise which types of zooming are of interest in dynamic queries. A multiple fish-eye zooming system can be implemented and evaluated.
- The system for displaying maps and individual points should be extended to handle more complex figures, which details should depend on the zoom level.

## Acknowledgments

Christopher Ahlberg
Arbetsmiljöfonden
Länsarbetsnämnden Göteborg
Arbetsförmedlingen Haga
Arbetsförmedlingen Västra Frölunda
NUTEK

## References

Ahlberg, C., Truvé, S., *Exploring Terra Incognita in the Design Space of Query Devices*, SSKKII Technical Report SSKII94.07, SSKKII, Göteborg University (1994)

Ahlberg, C., Williamson, C., Shneiderman, B., *Dynamic Queries for Information Exploration: An implementation and evaluation*, Proc. ACM CHI’92 Cnf., 619-626 (1992)

Ahlberg, C., Wistrand, E., *IVEE: An Environment for Automatic Creation of Dynamic Queries Applications*, Proc. of ACM CHI’95: Human factors in Comp. Systems (1995)

Allwood, C-M., Wikström, T. *Learning complex computer programs*, Behavior and Information Technology, vol 5, No 3, 217-255 (1986)

AMS, Arbetsmarknadsstyrelsen, *Nordisk yrkesklassificering: Svensk grundstandard*, Swedish Arbetsmarknadsstyrelsen (1983)

Beard, D. V., Walker II, J. Q. *Navigational techniques to improve the display of large two-dimensional spaces*, Behaviour & Information Technology, vol. 9, No. 6, 451 - 466 (1990)

Bederson, B. B., Hollan, J.D., *Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics*, UIST (1994)

Bier E., A. et al., *Toolglass and Magic Lenses: The See-Through Interface*, Proc. of ACM CHI’94 Conf. on Human Factors in Comp. Systems,445-446 (1994)

Bly S., *Presenting Information in Sound*, ACM (1981)

Brandon P., *The Virtual Tourist - Europe*, WWW document, http://wings.buffalo.edu/world/europe.html (1994)

Bobick, A.F., *Natural Object Categegorization*, Report number AI-TR 1001, MIT Artificial Intelligence Lab. (1987)

Chamberlin et al., *SEQUEL 2: A Unified Approach to Data Definition, Manipulation and Control,* IBM Journal of research and Development, 20:6, november (1976)

Cleveland, W.S., McGill, R., *Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods*, Journal of the American Statistical Association, 387, 531-554 (1984)

Cleveland, W.S., *Visualizing Data*, Hobart Press, New Jersey (1993)

Cleveland, W.S., *The Elements of Graphing Data*, Hobart Press, New Jersey, (1994)

Ericsson, K. A., Simon, H. A., *Protocol Analysis*, A chapter in *Handbook in Discource-Analysis* by T. van Dijk (1983)

Elmasri, R., Navathe, S. B., *Fundamentals of Database Systems*, The Benjamin/ Cummings Publishing Company, Inc. (1989)

Feiner, S., Beshers, C. *Worlds within Worlds. Metaphors for Exploring n-Dimensional worlds*, ACM SIGGRAPH 76-83, (1990)

Foley, J., van Dam, A., Feiner, S., Hughes, J., *Computer Graphics: Priciples and Practice*, Addison-Wesley (1990)

Hagman, J., *The Cluster Analyzer & Visualizer: a Prototype for Data Exploration,* SSKKII Technical Report SSKKII94.05, SSKKII, Göteborg University (1994)

Lin, X., Soerge, D., Marchionini, G., *A Self-Organizing Semantic Map for Information Retrieval,* Proc. of the Fourteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 262-269 (1991)

Jog N., Shneiderman, B., *User Controlled Smooth Zooming for Information Visualization with a Starfield Display*, HCI Lab, institute for Systems Research, Univ. of Maryland, College Park MD 20742-3255 USA (1994)

MackInlay, J. (1986), *Automating the Design of Graphical Presentations of Relational Inforamtion*, ACM Transactions on Information Systems, Vol 5, No 2, 1986

Mackinlay, J.D., Robertson, G.G., Card, S.K., *The Perspective Wall: Detail and Context Smoothly Integrated,* Proc. of ACM CHI'91 Conference on Human Factors in Computing Systems, 173-179, (1991)

Merit Network Information Center Services, *NFSnet statistics*, FTP documents, ftp://nic.merit.edu/nsfnet/, Merit Network, Inc. (1994)

Murtagh, F., *A Survey of Recent Advances in Hierarchical Clustering Algorithms*, The Computer Journal, vol. 26, no. 4 (1983)

Picket, R.M., Grinstein, G.G., *Iconographic display for visualizing multidimensional data,* IEEE Int. conf. on Systems, Man and Cybernetics (1988)

Sarkar, M., Snibbe, S. S., Tversky, O. J., Reiss, P., *Stretcing the rubber sheet: A Metaphor for viewing Large Layouts on Small Screens*, ACM Proc. of the ACM symposium on user interface software and technology, 81 - 91 (1993)

Sasseen, R.V., Caid, W.R. *Docuverse: A Context Vector-Based Approach to Graphical Representation of Information Content,* HNC, Inc., 5930, Cornerstone Court West, San Diego CA 92121 USA

Shaffer, D., Zuo, Z., Greenberg, S., Bartram, L., Dill, J., Dubs, S., Roseman, M. *Navigating Hierarchically Clustered Networks Through Fisheye and Fullzoom Methods*, ACM Transactions on Information Systems, (1994)

Shneiderman, B., Ahlberg, C. *The Alphaslider: A Compact and Rapid Selector,* Proc. of ACM CHI'94 Conference on Human Factors in Computing Systems, 365-371 (1994)

Shneiderman, B., Ahlberg, C., *Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays*, to appear in ACM Proc. of CHI'95 (1995)

Shneiderman, B. *Designing the user interface: Strategies for effective Human-Computer Interaction*, Addison-Wesley (1992)

Shneiderman, B. *Tree visualizations with Tree-maps: a 2-d space-filling approach*, ACM Transactions on graphics (1992)

Williams, M., D., *What makes RABBIT run?*, Int. Journal of Man-Machine Studies, 21, Academic Press (1984)

Williamson C., Shneiderman B., *The Dynamic HomeFinder: Evaluating Dynamic Queries in a Real-Estate Information Exploration System*, Proc of the Fifteenth Annual International ACM SIGIR Conf. on Research and Development in Information Retrieval, 338-346 (1992)

XGobi, publicly availiable software from Univ. of Washington, department of statistics, URL: http://www.stat.washington.edu/tech.reports/pub/index.html (1994)

Zloof, M., *Query-by-Example*, Proc, National Computer Conference, AFIPS Press, 431-437 (175)

# APPENDIX
## Questions used for evaluation

The questions are ordered in three sections 1-3, one for each visualization method (scatter plot, maps, clustering). In each section there are four questions a-d, where each of the four questions concerns one of the four query types (a - finding similarities, b - detecting trends, c - detecting anomalies, d - finding specific sets)

**Section 1**

1a. Rank the categories
- a) Service
- b) Engineering/Public
- c) Retail/Economics

according to salary.

1b How much more/less does a 50-year old applicant request as salary relative to a 20-year old?

1c.Find the three applicants with highest salary requests. Also find the three with lowest salary.

1d Gothenburg Trams needs a technician with at least 5 years of experience. Their personell department needs to know if the vacancy should be announced as a full time or a part time job. Can you give any advice?

**Section 2**

2a Rank the categories
- a) Service
- b) Engineering/Public
- c) Retail/Economics

according to number of applicants.

2b Information is needed concerning possible relationships between salary, age and exprecience. Can you find such relationships?

2c Find the youngest applicants with juridical education.

2d The road maintenance department in Gothenburg needs truck drivers with good experience. Suggest suitable applicants.

**Section 3**

3a Find one or more similar factor in the categories Healt Care and Administration.

3b A work advisor on a high school needs to recommend education to the pupils. Can you give any advice?

3c Can you find any men in professions dominated by women? Any women in professions dominated by men?

3d Volvo needs a person with technical education for work in their assembly line. The applicant should have at least 4 years of experince. References from previous work are preferred. Find a few matching applicants.