DVT

UNIT -4 Notes

Interaction concepts and techniques: Text and Document Visualization: Introduction- Levels of text representation- The Vector Space Model-Single document Visualizations- Document collection Visualizations- extended text visualizations
Interaction Concepts : Interaction Operators -Interaction Operands and Spaces - A Unified Framework.
Interaction Techniques: Screen Space -Object-Space -Data Space -Attribute Space- Data Structure Space - Visualization Structure – Animating Transformations -Interaction Control.

4 . Text and Document Visualization

We now have huge resources of information; from libraries, to e-mail archives, to all facets of applications running on the World Wide Web. Visualization is a great aid in analyzing such data. We can visualize in many different ways things such as a blog, a wiki, a twitter feed, billions of words, a collection of papers, or a digital library. Since visualizations are task dependent, we can look at what tasks are necessary for dealing with text, documents, or web-based objects. For text and documents, the most obvious tasks are searching for a word, phrase, or topic. For partially structured data, we may search for relationships between words, phrases, topics, or documents. For structured text or document collections, the key task is most often searching for patterns and outliers within the text or documents.

4.1 Introduction:

We define a collection of documents as a corpus (plural corpora). We deal with objects within corpora. These objects can be words, sentences, para graphs, documents, or even collections of documents. We may even consider images and videos. Often these objects are considered atomic with respect to the task, analysis, and visualization. Text and documents are often min imally structured and may be rich with attributes and metadata, especially when focused in a specific application domain. For example, documents have a format and often include metadata about the document (i.e., author, date of creation, date of modification, comments, size). Information retrieval systems are used to query corpora, which requires computing the relevance of a document with respect to a query. This requires document preprocessing and interpretation of the semantics of text. We can compute statistics about documents.

We can compute statistics about documents. For example, the number of words or paragraphs, or the word distribution or frequency, all can be used for author authenticity. Are there any paragraphs that repeat the same words or sentences? We can also identify relationships between paragraphs or documents within a corpus. For example, one could ask, “What documents relate to the spread of flu?” This is not a simple query; it isn’t simply searching for the word flu. Why? We can further look for natural connections or relationships between various documents. What clusters are there? Do they represent themes within the corpus? Similarity could be defined in terms of citations, common authorships, topics, and so on.

4.2 Levels of Text Representations

We define three levels of text representation: lexical, syntactic, and semantic. Each requires us to convert the unstructured text to some form of structured data.

1.Lexical level. The lexical level is concerned with transforming a string of characters into a sequence of atomic entities, called tokens. Lexical analyzers process the sequence of characters with a given set of rules into a new sequence of tokens that can be used for further analysis. Tokens can include characters, character n-grams, words, word stems, lexemes, phrases, or word n-grams, all with associated attributes. Many types of rules can be used to extract tokens, the most common of which are finite state machines defined by regular expressions.

2.Syntactic level. The syntactical level deals with identifying and tagging (an notating) each token's function. We assign various tags, such as sentence position or whether a word is a noun, expletive, adjective, dangling modifier, or conjunction. Tokens can also have attributes such as whether they are singular or plural, or their proximity to other tokens. Richer tags include date, money, place, person, organization, and time (Figure 10.3). The process of extracting these annotations is called named entity recognition (NER). The richness and wide variety of language models and grammars (generative, categorical, dependency, probabilistic, and functionalist) yield a wide variety of approaches.

3.Semantic level. The semantic level encompasses the extraction of meaning and relationships between pieces of knowledge derived from the structures identified in the syntactical level. The goal of this level is to define an analytic interpretation of the full text within a specific context, or even independent of context.

4.3 The Vector Space Model

Computing term vectors is an essential step for many document and corpus visualization and analysis techniques. In the vector space model [356], a term vector for an object of interest (paragraph, document, or document collection) is a vector in which each dimension represents the weight of a given word in that document. Typically, to clean up noise, stop words (such as "the" or "a") are removed (filtering), and words that share a word stem are aggregated together (.

The pseudocode below counts occurrences of unique tokens, excluding stop words. The input is assumed to be a stream of tokens generated by a lexical analyzer for a single document. The terms variable contains a hashtable that maps unique terms to their counts in the document.

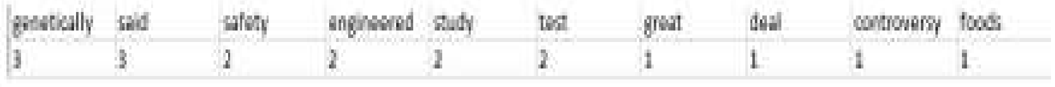
```
Count-Terms(tokenStream)
1 terms←∅ initialize terms to an empty hashtable.
 2 for each token t in tokenStream
3           do if t is not a stop word
 4                do increment (or initialize to 1) terms[t]
5 return terms
```

We can apply the pseudocode to the following text.

There is a great deal of controversy about the safety of genetically engineered foods. Advocates of biotechnology often say that the risks are overblown. ''There have been 25,000 trials of genetically modified crops in the world, now, and not a single incident, or anything dangerous in these releases,'' said a spokesman for Adventa Holdings, a UK biotech firm. During the 2000 presidential campaign, then-candidate George W. Bush said that ''study after study has shown no evidence of danger.'' And

Clinton Administration Agriculture Secretary Dan Glickman said that ‘‘test after rigorous scientific test’’ had proven the safety of genetically engineered products.

The paragraph contains 98 string tokens, 74 terms, and 48 terms when stop words are removed. Here is a sample of the term vector that would be generated by the pseudocode:



genetically	said	safety	engineered	study	test	great	deal	controversy	foods
3	3	2	2	2	2	1	1	1	1

1.Computing Weights:
This vector space model requires a weighting scheme for assigning weights to terms in a document. There exist many such methods, the most well known of which is the term frequency inverse document frequency (tf-idf).
Let Tf (w) be the term frequency or number of times that word w occurred in the document, and let Df(w) be the document frequency (number of documents that contain the word).
Let N be the number of documents.
We define Tf Idf(w)asputing Weights:

$$TfIdf(w) = Tf(w) * \log\left(\frac{N}{Df(w)}\right).$$

This is the relative importance of the word in the document, which matches our intuitive view of the importance of words. A word is more important the fewer documents it appears in (lower Df), as well as if it ap pears several times in a single target document (larger Tf). Said another way, we are more interested in words that appear often in a document, but not often in the collection. Such words are intuitively more important, as they are differentiating, separating or classifying words.

2.Zipf’s Law:

The normal and uniform distributions are the ones we are most familiar with. The power law distribution is common today with the large data sizes we encounter, which reflect scalable phenomena. The economist Vilfredo Pareto stated that a company’s revenue is inversely proportional to its rank—a classic power law, resulting in the famous 80-20 rule, in which 20% of the population holds 80% of the wealth.

Harvard linguist George Kingsley Zipf stated the distribution of words in natural language corpora using a discrete power law distribution called a Zipfian distribution. Zipf’s Law states that in a typical natural language document, the frequency of any word is inversely proportional to its rank in the frequency table. Plotting the Zipf curve on a log-log scale yields a straight line with a slope of −1.

One immediate implication of Zipf’s Law is that a small number of words describe most of the key concepts in small documents. There are numerous examples of text summarization that permit a full description with just a few words.

3. **Tasks Using the Vector Space Model :**

The vector space model, when accompanied by some distance metric, allows one to perform many useful tasks. We can use tf-idf and the vector space model to identify documents of particular interest.

For example, the vector space model, with the use of some distance metric, will allow us to answer questions such as which documents are similar to a specific one, which documents are relevant to a given collection of documents, or which documents are most relevant to a given search query—all by finding the documents whose term vectors are most similar to the given document, the average vector over a document collection, or the vector of a search query.

Another indirect task is how to help the user make sense of an entire corpus. The user may be looking for patterns or for structures, such as a document's main themes, clusters, and the distribution of themes through a document collection. This often involves visualizing the corpus in a two dimensional layout, or presenting the user with a graph of connections be tween documents or entities to navigate through. The visualization pipeline maps well to document visualization: we get the data (corpus), transform it into vectors, then run algorithms based on the tasks of interest (i.e., simi larity, search, clustering), and generate the visualizations.

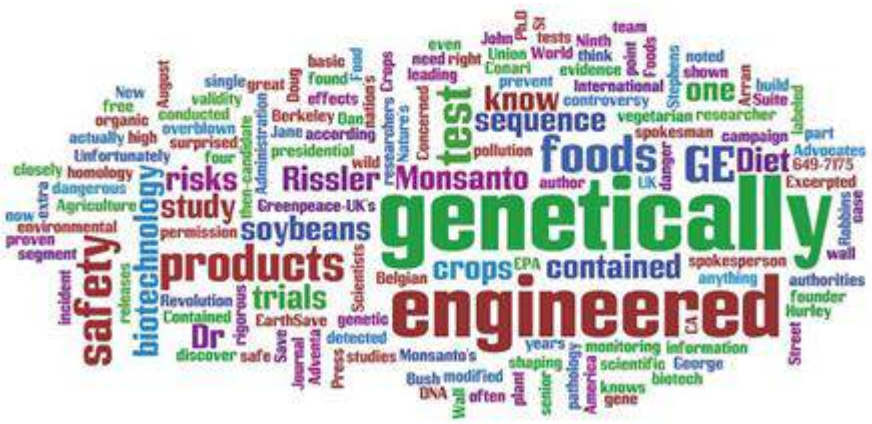
4.4 Single Document Visualizations:

Here we present several visualizations of a single text document, taken from the VAST Contest 2007 data set.

author biotechnology
contained crops danger
diet dr
engineered
foods
ge genetically
monsanto
products
researcher risks rissler safety
sequence soybeans study test trials wall world

A tag cloud visualization generated by the free service tagCrowd.com . The font size and darkness are proportional to the frequency of the word in the document

1.Word Clouds:



A Wordle visualization generated by the free service wordle.net . The size of the text corresponds to the frequency of the word in the document.

2.Word Tree:

The WordTree visualization is a visual representation of both term frequencies, as well as their context . Size is used to represent the term or phrase frequency. The root of the tree is a user-specified word or phrase of interest, and the branches represent the various contexts in which the word or phrase is used in the document

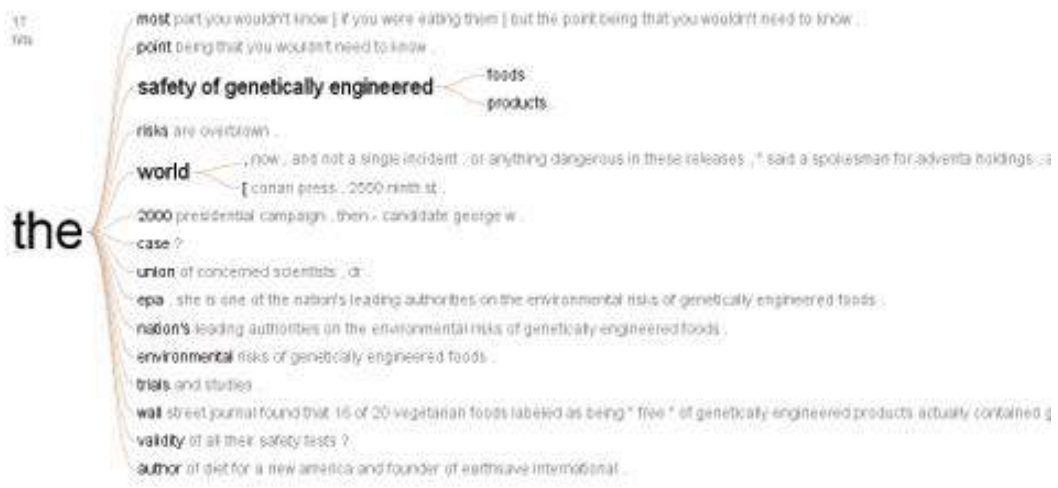

the
most part you wouldn't know | if you were eating them | but the point being that you wouldn't need to know .
point being that you wouldn't need to know .
safety of genetically engineered
foods
products
risks are overblown .
world
, now , and not a single incident , or anything dangerous in these releases , " said a spokesman for adventa holdings
[conan press , 2500 ninth st .
2000 presidential campaign , then - candidate george w .
case ?
union of concerned scientists , dr
epa , she is one of the nation's leading authorities on the environmental risks of genetically engineered foods .
nation's leading authorities on the environmental risks of genetically engineered foods .
environmental risks of genetically engineered foods .
trials and studies .
wall street journal found that 16 of 20 vegetarian foods labeled as being " free " of genetically engineered products actually contained
validity of all their safety tests ?
author of diet for a new america and founder of earthsave international .

Fig: A WordTree visualization generated by the free service ManyEyes. The branches of the tree represent the various contexts following a root word or phrase in the document.

3.TextArc: We can extend the representation of word distribution by displaying connectivity. There are several ways in which connections can be computed. TextArc is a visual representation of how terms relate to the lines of text in which they appear . Every word of the text is drawn in order around an ellipse as small lines with a slight offset at its start. As in a text cloud, more frequently occurring words are drawn larger and brighter. Words with higher frequencies are drawn within the ellipse, pulled by its oc currences on the circle (similar to RadViz). The user is able to highlight the underlying text with probing and animate “reading” the text by visualizing the flow of the text through relevant connected terms.

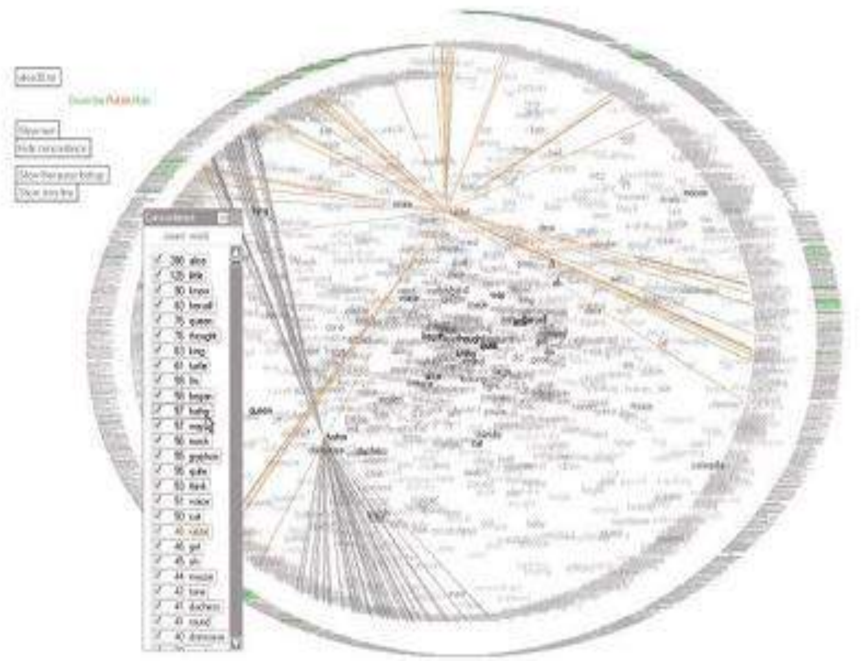
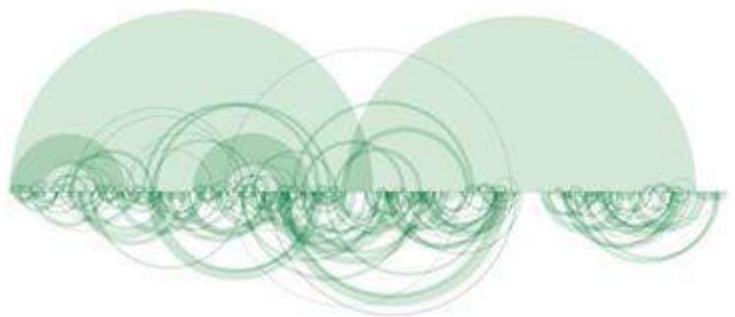


Fig: A TextArc visualization that uses the full text of Alice in Wonderland. W ords that occur evenly throughout the document are positioned in the center of the display, while words that appear only in specific sections are located closer to the circumference.

4.Arc Diagrams:
Arc diagrams are a visualization focused on displaying repetition in text or any sequence. Repeated subsequences are identified and connected by semicircular arcs. The thickness of the arcs represents the length of the subsequence, and the height of the arcs represents the distance between the subsequences.



5.Literature Fingerprinting:

Literature fingerprinting is a method of visualizing features used to char acterize text. Instead of calculating just one feature value or vector for the whole text (this is what is usually done), we calculate a sequence of feature values per text and present them to the user as a characteristic fingerprint of the document. This allows the user to “look inside” the document and analyze the development of the values across the text. Moreover, the structural information of the document is used to visualize the document on different levels of resolution. Literature fingerprinting was applied to an authorship attribution problem to show the discrimination power of the standard measures that are assumed to capture the writing style of an author .

4.5 Document Collection Visualization

In most cases of document collection visualizations, the goal is to place similar documents close to each other and dissimilar ones far apart. This is a minimax problem and typically O(n2). We compute the similarity between all pairs of documents and determine a layout. The common approaches are graph spring layouts, multidimensional scaling, clustering (k-means, hierarchical, expectation maximization (EM), support vector), and self-organizing maps. We present several document collection visualizations, such as self organizing maps, cluster maps, and themescapes.

1.Self-Organizing Maps:

A self-organizing map (SOM) is an unsupervised learning algorithm using a collection of typically 2D nodes, where documents will be located. Each node has an associated vector of the same dimensionality as the input vectors (the document vectors) used to train the map. We initialize the SOM nodes, typically with random weights. We choose a random vector from the input vectors and calculate its distance from each node.
We adjust the weights of the closest nodes (within a particular radius), making each closer to the input vector, with the higher weights corresponding to the closest selected node. As we iterate through the input vectors, the radius gets smaller. An example of using SOMs for text data is shown in Figure , which shows a million documents collected from 83 newsgroups.

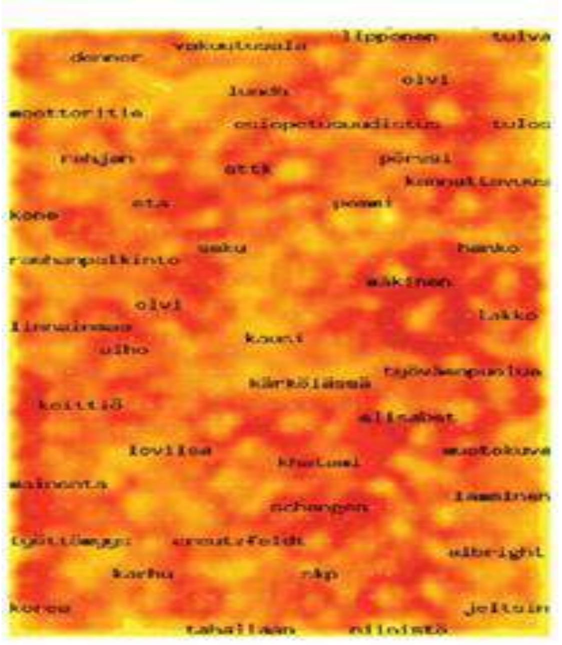


Fig: A self-organizing map (SOM) layout of Finnish news bulletins. The labels show the topical areas, and color represents the number of documents, with light areas containing more

2,Themescapes:
Themescapes are summaries of corpora using abstract 3D landscapes in which height and color are used to represent density of similar documents. The example shown in Figure from Pacific Northwest National Labs [407] represents news articles visualized as a themescape. The taller moun tains represent frequent themes in the document corpus (height is propor tional to number of documents relating to the theme)

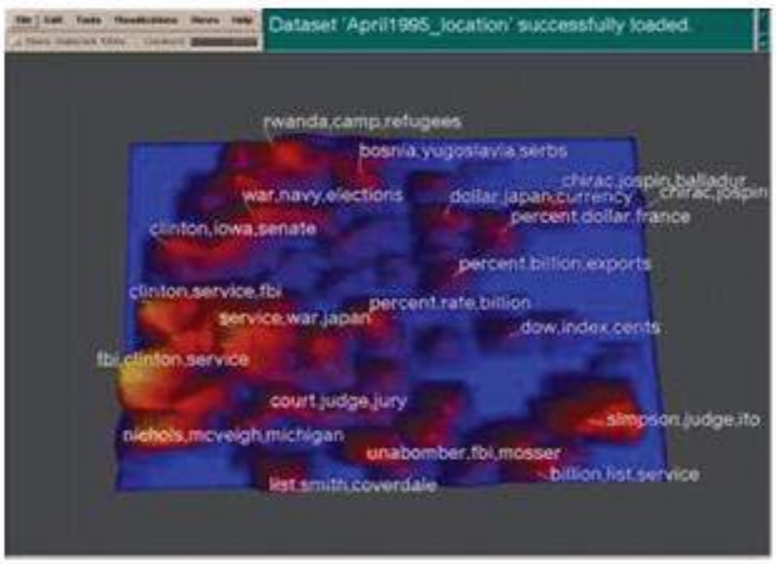


Fig: A themescape from PNNL that uses height to represent the frequency of themes in news articles.

3.Document Cards:
Document cards are a compact visualization that represents the document's key semantics as a mixture of images and important key terms, similar to cards in a top trumps game . The key terms are extracted using an advanced text-mining approach based on an automatic extraction of document structure. The images and their captions are ex tracted using a graphical heuristic, and the captions are used for a semi semantic image weighting. Furthermore, the image color histogram is used to classify images into classes (class 1: photography/rendered image, class 2: diagram/sketch/graph, class 3: table) and show at least one representative from each non-empty class.

4.6 Extended Text Visualizations

1.Software Visualization
Eick et al. developed a visualization tool called SeeSoft that visualizes statistics for each line of code (i.e., age and number of modifications, programmer, dates). In Figure , each column represents a source code file with the height representing the size of the file. If the file is longer than the screen, it continues into the next column. In the classic SeeSoft representation, each row represents one line of code. Since the number of lines is too large for one row, each line of code is represented by a pixel in the row. This increases the number of lines that can be displayed. Color is used to represent the call count. The more red a line is, the more often the line is called, and thus is a key hot-spot. A blue line is an infrequently called one. Color can be used to represent other parameters, such as time of last modification or number of modifications. With a 1K × 1K screen, SeeSoft is able to display up to 50,000 lines of code. This figure contains 52 files with 15,255 lines of code. The selected file is file1.c, a block of code with a zoomed-in view of line 408.

2.Search Result Visualization

Marti Hearst developed a simple query result visualization foundationally similar to Keim's pixel displays called TileBars , which displays a number of term-related statistics, including frequency and distribution of terms, length of document, term-based ranking, and strength of ranking. Each document of the result set is represented by a rectangle, where width indicates relative length of the document and stacked squares correspond to text segments . Each row of the stack represents a set of query terms, and the darkness of the square indicates the frequency of terms among the corresponding terms. Titles and the first words from the document appear next to its TileBar. Each large rectangle indicates a document, and each square within the document represents a text segment. The darker the tile, the more frequent is the query term set. This produces a representation that is compact and provides feedback on document structure reflecting relative document length, query term frequency, and query term distribution.

3.Temporal Document Collection Visualizations

ThemeRiver , also called a stream graph, is a visualization of thematic changes in a document collection over time . This visualization assumes that the input data progresses over time. Themes are visually represented as colored horizontal bands whose vertical thickness at a given horizontal location represents their frequency at a particular point in time.

Jigsaw is a tool for visualizing and exploring text corpora . Jigsaw's calendar view positions document objects on a calendar based on date entities identified within the text. When the user highlights a document, the entities that occur within that document are displayed.

Wanner et al. developed a visual analytics tool for conducting semiautomatic sentiment analysis of large news feeds . While the tool automatically retrieves and analyzes RSS feeds with respect to positive and   negative opinion words, the more demanding news analysis of finding trends, spotting peculiarities, and putting events into context is left to the human expert. each single news item is represented by one visual object and plotted on a horizontal time axis according to its publication time. The shape and color of an item reveal information about the category it belongs to, and its vertical shift indicates whether it has a positive connotation (upward shift) or a negative one (downward shift).

4. Representing Relationships
Jigsaw also includes an entity graph view , in which the user can navigate a graph of related entities and documents. In Jigsaw, entities are connected to the documents in which they appear. The Jigsaw graph view does not show the entire document collection, but it allows the user to incrementally expand the graph by selecting documents and entities of interest.

4.7 Interaction Concepts

Interaction within the data and information visualization context is a mechanism for modifying what the users see and how they see it. Many classes of interaction techniques exist , including:

• navigation—user controls for altering the position of the camera and for scaling the view (what gets mapped to the screen) such as panning, rotating, and zooming.

• selection—user controls for identifying an object, a collection of objects, or regions of interest to be the subject of some operation, such as highlighting, deleting, and modifying.

• filtering—user controls for reducing the size of the data being mapped to the screen, either by eliminating records, dimensions, or both.

• reconfiguring—user controls for changing the way data is mapped to graphical entities or attributes, such as reordering the data or layouts, thereby providing a different way of viewing a data subset.

• encoding—user controls for changing the graphical attributes, such as point size or line color, to potentially reveal different features.

• connecting—user controls for linking different views or objects to show related items. • abstracting/elaborating—user controls for modifying the level of detail.

• hybrid—user controls combining several of the above in one technique, for example, increasing the screen space assigned to one or more focus areas to enable users to see details, while showing the other areas of data in a smaller space, in a way that preserves context.

A. Interaction Operators

In this section we describe in more detail a wide range of interaction operations commonly found in data and information visualization.

1.Navigation Operators:

Navigation (also sometimes referred to as exploration) is used to search for a subset of data to be viewed, the orientation of this view, and the level of detail (LOD). The subset in question may be one that is recognized by some visual pattern or one on which further or more detailed exploration is desired. In a typical three-dimensional space, this can be specified using a camera location, a viewing direction, the shape and size of the viewing frustum, and an LOD indicator. In multiresolution visualizations, LOD changes can correspond to drilling down or rolling up hierarchical representations of the data.

2.Selection Operators

In selection, the user isolates a subset of the display components, which will then be subjected to some other operation, such as highlighting, deleting, masking, or moving to the center of focus. Many variations on selection have been developed to date [461], and decisions need to be made on what the results should be for a sequence of selections. For example, should a new selection replace the previous selection or supplement the previous selection? The granularity of selection is also an issue. Clicking on an entity in the display might result in selection of the smallest addressable component (e.g., a vertex or edge) or might target a broader region around the specified location (e.g., a surface, region of the screen, or object.

3.Filtering Operators

Filtering, as the name implies, reduces the volume of data to be visualized by setting constraints specifying the data to be preserved or removed. A good example of such a filter is the dynamic query specification described by Shneiderman et al.. One- or two-handled sliders are used to specify a range of interest, and the visualization is immediately updated to reflect the changes made by the user. Range queries are just one form of filtering, however. One might also select items from a set or list to preserve or hide, such as the column hiding operation in Excel.

4.Reconfiguring Operators

Reconfiguring the data within a particular visualization can often be used to expose features or cope with complexity or scale. By reorganizing the data, say by filtering some dimensions and reordering those that remain, different views are provided to the user. For example, a powerful tool with tablebased visualizations is to sort the rows or columns of the data to highlight trends and correlations. Other types of reconfiguration might be to change the dimensions being used to control the x- and y-coordinates of a plotted marker, or even to derive positions based on transformations of the data. Popular instances of this include the use of principal component analysis (PCA) or multidimensional scaling (MDS) to lay out data points so that relationships among all the dimensions are best conveyed in the 2D layout.

5.Encoding Operators

Any given data set can be used to generate countless different visualizations. Recoding can provide the user a library of possible different types of visualization; features of the data that are difficult or impossible to see with one such mapping might become quite apparent in another. For example, a scatterplot with one axis representing years may have many points that overlap, whereas a parallel coordinate display would represent these uniquely. Many visualization tools today support multiple types of visualization, because no single visualization is effective for all tasks that need to be carried out by the user. Each visualization is most suitable for a subset of data types, data set characteristics, and user tasks. While some work has been done to identify or select the best visualization, it is apparent that such guidelines are at best suggestions, and the analyst is most likely to benefit from examining their data using a number of different mappings and views. This is the essence of interactive visualization.

6.Connection Operators

A frequent use for selection operations is to link the selected data in one view to the corresponding data in other views. While other forms of connection between sub windows of an application exist, such as when opening a new data file, linked selection is probably the most common form of communication between windows found in modern visualization tools. Its popularity stems in large part from the fact that each view of one’s data can reveal interesting features, and that by highlighting such a feature in one view, it is possible to build a more complete mental model of the feature by seeing how it appears in other views . This can also help reveal relationships between this feature and others in the data set. For example, when examining multivariate spatial data, it is often useful to jump between the spatially referenced view and the dependent variable view, which often does not preserve the spatial attributes.

7.Abstraction/Elaboration Operators

In dense data and information displays, it is often desirable to focus in on a subset of the data to acquire details (elaboration) while reducing the level of detail (abstraction) on other parts of the data set. One of the most popular techniques of this type is using distortion operators. While some researchers classify distortion as a visualization technique, it is actually a transformation that can be applied to any type of visualization. Like panning and zooming, distortion is useful for interactive exploration.

B. Interaction Operands and Spaces:

Parameters of the interaction operators described in the previous section are discussed in more detail later in the chapter. First, however, we present our categorization of the interaction operands, as this will help clarify the role these parameters take in the interaction process and their semantics within the different spaces.

An interaction operand is the section of space upon which an interactive operator is applied. To determine the result of an interactive operation, one needs to know within what space the interaction is to take place. In other words, when a user clicks on a location or set of locations on the screen what entities does he or she wish to indicate?

1.Screen Space (Pixels)

Screen space consists of the pixels of the display. Navigation in screen space typically consists of actions such as panning, zooming, and rotation. Note that in each case, no new data is used; the process consists of pixel-level operations such as transformation, sampling, and replication.

Operations such as transformation, sampling, and replication. Pixel-based selection means that at the end of the operation, each pixel will be classified as either selected or unselected. As previously mentioned, the selection can be performed on individual pixels, rectangles or circles of pixels, or on arbitrarily shaped regions that the user specifies. Selection areas may also be contiguous or non-contiguous. Screen space filtering consists of cropping or masking a region of pixels.

2.Data Value Space (Multivariate Data Values)

Operations on data space are applied directly to the data, rather than to the screen. Navigating in data value space involves using the data values as a mechanism for view specification. The analogous operations for panning and zooming would be to change the data values being displayed; panning would shift the start of the value range to be shown, while zooming would decrease the size of this range.

Data value space selection is similar to a database query in that the user specifies a range of data values for one or more data dimensions.

3.Data Structure Space (Components of Data Organization)

Data can be structured in a number of ways, such as lists, tables, grids, hierarchies, and graphs. For each structure, one can develop interaction mechanisms to indicate what portions of the structure will be manipulated, and how this manipulation will be manifested. Navigation in data structure space involves moving the view specification along the structure, as in showing sequential groups of records, or moving down or up a hierarchical structure (as in drill-down and roll-up operations).

For example, Figure shows the difference between a screen space zoom (involving pixel replication) and a data structure space zoom (involving retrieval of more detailed data). A technique presented by Resnick et al. [335] selects subsets of data to be visualized by specifying a focus, extents, and density in a regular grid structure, where the density can be a function of distance from the focus.

4. Attribute Space (Components of Graphical Entities)

In attribute space, operators are focused on one or more of the attributes associated with the graphical entity being used to convey information. Such attributes could include color, size, shape, or any other of the eight visual variables. Navigation in attribute space is similar to that in data value space; panning involves shifting the range of the values of interest, while zooming can be accomplished by either scaling the attributes or enlarging the range of values of interest.

As in data value-driven selection, attribute-space selection requires the user to indicate the subrange of a given attribute of interest. For example, given a visual depiction of a color map, a user can select one or more entries to highlight. Similarly, if data records have attributes such as quality or uncertainty, a visual representation of these attributes, accompanied by suitable interaction techniques, can allow users to filter or emphasize data according to the attributes. Remapping is often done in attribute space, either via selecting different ranges of an attribute to be used in the data to graphic mapping, or by choosing a different attribute to be controlled by the data.

5.Object Space (3D Surfaces)

In these displays, the data is mapped to a geometric object, and this object (or its projection) can undergo interactions and transformations. Navigation in object space often consists of moving around objects and observing the surfaces on which the data are mapped. The system should support global views of the object space as well as close-up views. The latter may be constrained to enable the user to find good views more quickly. Selection involves clicking anywhere on the object(s) of interest, or indicating target objects from a list.

A typical example of remapping in object space would be changing the object upon which the data is being mapped, such as switching the mapping of geographical data between a plane and a sphere.

6.Visualization Structure Space

A visualization consists of a structure that is relatively independent of the values, attributes, and structure of data. For example, the grid within which a scatterplot matrix is drawn and the axes displayed in many types of visualizations are each components of the visualization structure, and can be the focus of interactions. Visualization dashboards, consisting of a set of tightly or loosely connected visualizations juxtaposed on the screen, have become increasingly popular for monitoring and analyzing complex data environments, such as for business analytics and command-and-control centers.

Examples of navigation in visualization structure space might include moving through pages in a spreadsheet-style visualization tool or zooming in on an individual plot in a scatterplot matrix.

C. A Unified Framework

For each interaction operator to be applied to a specified space/operand, several parameters are required. Some of these may be constants for a given system. The parameters are described below.

1. **Focus**. The location within the space at the center of the area of user interest. There may be multiple simultaneous foci, though for navigation this usually requires multiple display windows.

2.Extents. The range within the space (can be multidimensional) defining the boundaries of the interaction. The metric used for specifying the range is specific to the space; in screen space this would be in pixels, while in structure-space this might be the number of rows in a table or links in a graph.

3.Transformation. The function applied to the entities within the extents, generally a function of distance or offset from the focus. The shape of this transformation might also depend on the type of information being affected. For example, text distortion is more likely to have a flat peak to the transformation function. Another component of the transformation is the degree or scale factor for the transformation, thus allowing varying amounts of the specified action.

4.Blender. How to handle parts of space touched by more than one interaction. For selection, this operation may include performing logical operations on overlapping entities. For distortion, Keahey and Robertson identify several approaches, including weighted average, maximal value, and composition]. Each has advantages in terms of smoothness and ease of interpretation.

4.8 Interaction Techniques

In this section, we discuss algorithm and implementation details pertaining to the interaction concepts .

1.Screen Space

Screen-space distortion is a common tool for providing focus+context. We present one example of such a distortion: the fisheye lens [140]. The implementation of a fisheye lens is rather straightforward. One needs to specify a center point for the transformation (cx, cy) a lens radius rl, and a deflection amount d. We then transform the coordinates of our image into polar coordinates relative to the center point. The lens effect is simply a transformation on the radius portion of the coordinates. One popular transformation is

$$r_{\text{new}} = s \log(1 + d * r_{\text{old}}),$$

where

$$s = \frac{r_l}{\log(1 + d * r_l)}.$$

The overall pseudocode for the algorithm is as follows:

```
1. Clear the output image.
2. For each pixel in the input image,
(a) Compute the corresponding polar coordinates.
 (b) If the radius is less than the lens radius,
i. Compute the new radius;
 ii. Get the color at this location in the original image;
iii. Set the color of the pixel in the result image.
(c) Else set result image pixel to that of original image.
```

Depending on the transformation used in screen space distortion, it is possible to either leave gaps or to cause pixels to overlap on the output image. While the overlaps do not cause much concern (in some implementations, these are averaged), the gaps need to be resolved via interpolation. Different functions give different shapes to the lens; for text visualization, it is common to use a piecewise linear function with a flat top to make reading easier.

2.Object Space (3D Surfaces)

One of the methods for navigating large document and data visualizations is via a perspective wall [283], which shows one panel of the view on a surface orthogonal to the viewing direction, with the rest of the panels oriented so they fade into the distance via perspective foreshortening. A simplistic version of the perspective wall can be created by noting that the front wall implements a horizontal scaling on a segment of the 2D image being mapped, while the adjacent segments are subjected to a horizontal and vertical scaling proportional to the distance to the edge of the front wall, along with a shearing transform. Thus if the left, center, and right sections of the original image to be plotted are bounded by (x0, xleft, xright, x1) and the left, center, and right panels of the result image are (X0, Xleft, Xright, X1), the transformation is as follows:

- for $x < x_{\text{left}}$:

$$x' = X_0 + (x - x_0) * \frac{(X_{\text{left}} - X_0)}{(x_{\text{left}} - x_0)},$$
$$y' = (X_{\text{left}} - x') + y\left(1 - \frac{(X_{\text{left}} - x')}{(X_{\text{left}} - X_0)}\right),$$

- for $x_{\text{left}} <= x < x_{\text{right}}$:

$$x' = X_{\text{left}} + (x - x_{\text{left}}) * \frac{(X_{\text{right}} - X_{\text{left}})}{(x_{\text{right}} - x_{\text{left}})},$$
$$y' = y,$$

- for $x >= x_{\text{right}}$:

$$x' = X_{\text{right}} + (x - x_{\text{right}}) * \frac{(X_1 - X_{\text{right}})}{(x_1 - x_{\text{right}})},$$
$$y' = (x' - X_{\text{right}}) + y\left(1 - \frac{(x' - X_{\text{right}})}{(X_1 - X_{\text{right}})}\right).$$

Given the perspective wall, the user could interact with it by sequential page movements (forward and backward), either one at a time or in a scanning or page-flipping manner. Indexes could also be used to jump to sections of interest, perhaps implemented as a tab sticking out of the top of the page at the start of each section. It might also be useful to have more than one page be in the direct focus, thus trading off some detail for a larger undistorted window on the data.

3.Data Space (Multivariate Data Values)

Data space transformations are actually quite common in data analysis and visualization. Some typical functions include:

• scaling and translating to fit a particular range of values;
• exponential or log scaling to spread or compress the distribution of the data;
• sinusoidal functions to help study cyclic behavior;
• absolute value transformation to focus on magnitudes of change;
• negating the values so that low values become high, and vice versa

A key requirement for these and other types of transformations is that the user be made aware that a transformation on the data has occurred. It is all too common that a data set is modified without the viewer being informed of this modification. Clear labeling of axes is essential; augmenting the visualization with a note describing any data transforms that have been applied reduces the chances of misinterpretation. Another key requirement is to translate and scale the resulting data values to fall within an acceptable range for the graphical entities and attributes being rendered. Failure to incorporate this transformation can result in objects being mapped off the display, color wrap-around, negative values for graphical attributes, and other undesirable artifacts.

4.Attribute Space (Properties of Graphical Entities)

A great many interactions on attributes have direct equivalents with interactions in data space. Examples include global or local scaling, boundary enhancement, and color equalization. However, in some situations it makes more intuitive sense to consider the interaction to be simply on the graphical representation of the data, rather than the data itself. Thus, for example, when enlarging a data glyph of interest, the intuition is that you are not changing the data, just modifying the attributes of the glyph. There is also some overlap between the actions in attribute space and those in screen space, as the previous example hints at. If the glyph is not overlapping other glyphs, the enlarging can be based on pixel operations, rather than scaling the drawing primitives. In attribute space, however, you are more likely to avoid blocky artifacts that are typical of pixel-oriented zooming operations.

Perhaps the widest range of attribute space interactions involves modification of the color and opacity attributes. Techniques such as contrast enhancement, color histogram equalization, and others have been designed to make better use of the color space and make features of the data more readily perceived. For example, via controls for contrast and brightness, we can emphasize certain regions of the data to attract the attention of the analyst and make feature detection, classification, or measurement easier.

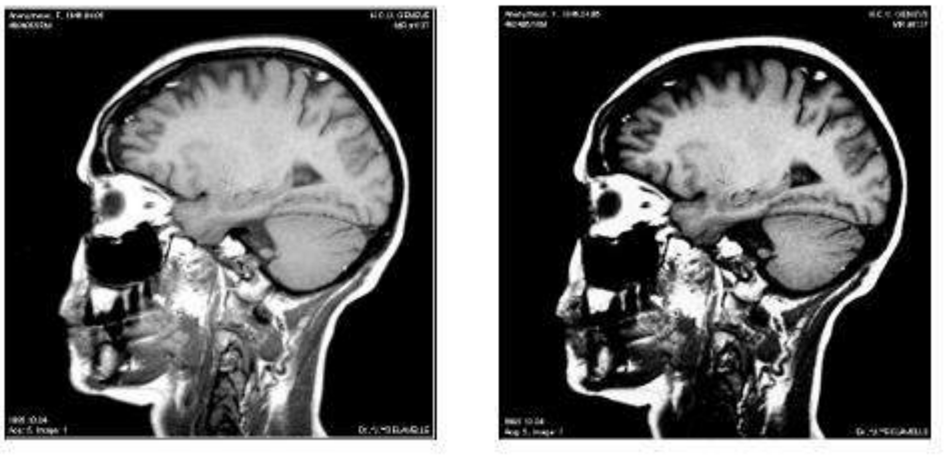


Figure 12.1. Example of modifying the contrast and brightness of an image to emphasize certain features.

5. **Data Structure Space (Components of Data Organization)**

Most of the common structures for data (i.e., ordering, grids, grouping, hierarchies, and networks) have logical operations that users will want to interactively perform. In terms of the implementation, the main decisions to be made are the degree to which the operation can or should be automated, and whether the interactions will be specified directly on the visualization or in a separate dialog box. For automatic techniques, one must evaluate the tradeoffs between thorough, time-consuming techniques versus quick, yet suboptimal methods. In this section we will examine some of the design decisions that need to be made in providing these sorts of interactive operations.

Consider dimension ordering in multivariate data visualization. Fully manual techniques might involve manipulating text entries in a list, either via shifting operations (move up, move down) or drag-and-drop methods. In some visualizations, such as parallel coordinates or scatterplot matrices, direct manipulation of the axes may be possible. In the case of the scatterplot matrix, the movement of one element of the matrix would entail changes to entire rows and columns, in order to preserve the symmetry along the diagonal plots.

Automated dimension reordering requires at least two major design de cisions: how do you measure the goodness of an ordering, and what strategy will you use to search for good orderings? Many measures are possible. One commonly used one is the sum of the correlation coefficients between each pair of dimensions. The correlation coefficient between two dimensions is defined as follows:

$$\rho_{X,Y} = \frac{\sum(x_i y_i - n\mu_X \mu_Y)}{(n-1)\sigma_X \sigma_Y},$$

where n is the number of data points, X and Y are the two dimensions, x_I and y_i are the values for the i^{th} data point, μ_X is the mean value for X, and σ_X is the standard deviation for X. Other statistical measures of similarity between dimensions include the correlation ratio and the mutual information.

Another measure of goodness could involve the ease of interpretation. Different dimension orders can result in displays with more or less visual clutter or structure . For example, one might conjecture that when using star or profile glyphs to represent data points, simpler shapes would be easier to analyze than complex shapes. Thus, if one could measure the average or cumulative shape complexity, e.g., by counting the number of concavities or peaks in the shape, this could be used to compare the visual complexity of two different dimension orders.

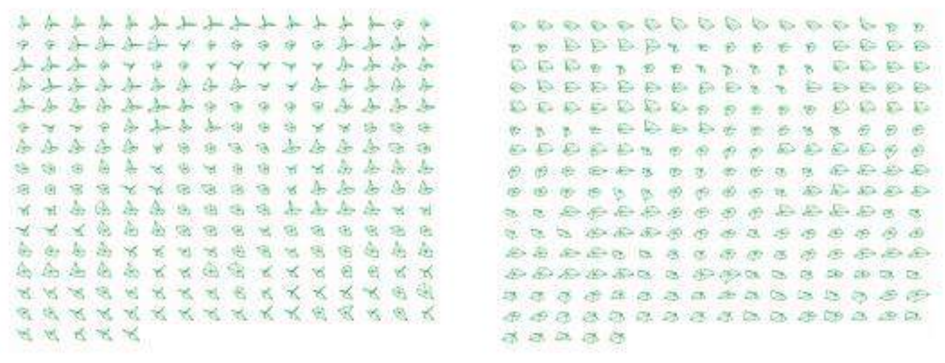


Fig: Example of shape simplification via dimension reordering. The left image shows the original order, while the right image shows the results of reordering to reduce concavities and increase the percentage of symmetric shapes.

6.Visualization Structure Space (Components of the Data Visualization)

Several of the techniques described earlier can be readily adapted to work in visualization structure space. For example, the fisheye lens technique in screen space could just as easily be used in any of the visualization structures with sequences or grids of components; we could use the same distortion function to adjust the spacing between axes in parallel coordinates or the size of the grid cells in scatterplot matrices or TableLens. As another example, we might adapt data clustering, filtering, or optimization actions (both manual and automated) to select and organize sets of visualizations on the screen, such as the results of performing volume visualization with hundreds or thousands of different parameter sets. Even drill-down, roll-up operations can be effectively utilized in multi-component visualizations that are hierarchically structured. One key is to make sure the user is aware at all times of the operations that can be done on the structure of the visualization, using consistent icons, labels, and visual representations that the user can quickly learn to use.

7.Animating Transformations

Virtually all user interactions within visualization systems will result in a change in the image being viewed. Some of these changes will be quite dramatic, such as when opening a new data set. Others may keep some aspects of the view the same, while changing other aspects. In situations where the user needs to maintain context while having his or her attention drawn toward the change, it is best to provide smooth transitions between the initial and final visualizations. Thus, for example, when rotating a three-dimensional object or data set, smoothly changing the orientation is generally better than simply jumping to the final orientation. In some cases, this might involve simply a linear interpolation between the starting and ending configurations, perhaps with the number of intermediate views dependent on the magnitude of the change. In some cases, however, linear interpolation does not result in a constant rate of change (such as moving a camera along a curved path). In addition, in most cases, a more appealing result can be obtained by performing smooth acceleration and deceleration of the change, rather than a constant velocity change. In this section we will discuss the algorithms necessary to achieve this control of changes. Students interested in a detailed exposure to these concepts will find them in textbooks on computer animation.

8.Interaction Control:

At each stage of the pipeline , the user requires mechanisms to control the type, location, and level of each interaction as he or she navigates within both the data and the visualization. The realization of these controls must be intuitive, unambiguous, and at a level of detail and accuracy appropriate for the space being operated upon. In particular, the following lists typical controls and reasonable candidates for their implementation.

1.Focus selection.
Selection is most readily accomplished via direct manip ulation tools, e.g., using a mouse or other selection device to indi cate the focus location. In screen and object space, this can be eas ily accomplished via normal selection operations. In data space, an n-dimensional location might need to be indicated. Depending on the method of display, this could involve multiple selections (e.g., selecting in a scatterplot matrix only enables simultaneous specification of two dimensions). In attribute and structure space, one first needs a graph ical depiction of the structure or the range of the attribute, such as a display of a tree or table, or a curve showing the range of colors in the color map. Finally, the focus can be specified implicitly, by assuming that the focus is the center of the extents of the interaction, which can be specified as outlined below.

2.Extent selection.

Specifying the extents for an interaction is generally depen dent on the type of interaction and the space in which the interaction is being applied, and can be done either via direct manipulation or separate interface tools. It may be specified via a single value (e.g., a radius or maximum number of items) or via a vector of values (e.g., a range for each data dimension or a set of constraints). In many systems, the extents are often hard-coded to reduce the effort in per forming the operation

3.Interaction type selection.

Given the many types of interaction possible, and the variety of spaces in which they may be applied, a reasonable interface for this task would be a pair of menus: one to select the space, and the other to specify the general class of the interaction. Icons or buttons in the user interface could also be used, as the number of choices would normally be modest.

4.Interaction level selection.

The degree of interaction is an important control parameter that can be specified by a single value (e.g., the magnitude of scaling that will occur at the focal point). A slider or dial is sufficient for this activity, along with a button to reset the operation to its minimum level. A direct manipulation equivalent would be to associate upward mouse motions with an increased interaction level, perhaps in conjunction with direct manipulation of the extents via horizontal mouse motions.

5.Blender type selection.

If more than one interaction can be simultaneously viewed and manipulated, there must be some mechanism for selecting a strategy for mixing regions of space affected by more than one inter action. As with interaction-type selection, this is best accomplished via a menu of options. Available options might be dependent on both the space in which the interaction is occurring and the type of inter action being used. As interactions in different spaces are applied at different points in the pipeline, it is necessary to consider methods for controlling the combination of interactions involving two or more spaces.

An important feature that should be present in all operations is the animation of interpolated values of the interaction parameters as they are changed. This has been shown to be extremely effective in many implementations of operators for helping users to both preserve context and to obtain a better understanding of the effects of the operation on the data [435]. Rapid changes can lead to confusion and a loss of orientation, especially when interactively exploring large data or information repositories.

6.Algorithms for Selection

The following pseudocode derives a set of selected records in a scatterplot based on a selection rectangle created by the user. This could be readily extended to other spaces as well. Hashing or other access/indexing functions could help avoid scanning the entire data set.

```
SCATTERPLOT-SELECT($xDim, yDim, xMin, xMax, yMin, yMax$)
s ← ∅ ▷ Initialize the set of records
for each record $i$ ▷ For each record,
        do $x$ ← NORMALIZE($i, xDim$) ▷ derive the location,
           $y$ ← NORMALIZE($i, yDim$)
           if $xMin < x < xMax$ and $yMin < y < yMax$
                do $s \leftarrow s \cup i$ ▷ select points within the rectangle.
return $s$
```

The next pseudocode determines whether a point is in a given polygon. This task is essential in determining, for example, which polygon in a choropleth map or glyph in a scatterplot is under a given mouse location. The code uses the even-odd rule algorithm [130], which counts the number of times a ray coming from the point to be tested intersects faces of the polygon. If the ray intersects the polygon an odd number of times, then the point is inside the polygon.

```
POINT-IN-POLYGON($xs, ys, numPoints, x, y$)
$j \leftarrow numPoints - 1$
$oddNodes$ ← false
for $i \leftarrow 0$ to $numPoints - 1$
        do if $ys[i] < y$ and $ys[j] >= y$ or $ys[j] < y$ and $ys[i] >= y$
                do if $xs[i] + (y - ys[i])/(ys[j] - ys[i]) * (xs[j] - xs[i]) < x$
                        do $oddNodes$ ← not $oddNodes$
           $j \leftarrow i$
return $oddNodes$
```