UNIT-3

Unit-3:

Visualization Techniques: Spatial Data: One-Dimensional Data - Two-Dimensional Data – Three-Dimensional Data - Dynamic Data - Combining Techniques.

Geospatial Data: Visualizing Spatial Data- Visualization of Point Data -Visualization of Line Data-Visualization of Area Data - Other Issues in Geospatial Data Visualization

Multivariate Data: Point-Based Techniques - Line- Based Techniques -Region-Based Techniques-Combinations of Techniques – Trees Displaying Hierarchical Structures – Graphics and Networks-Displaying Arbitrary Graphs/Networks.

References:

1.Matthew Ward, Georges Grinstein and Daniel Keim, : Interactive Data Visualization Foundations, Techniques, Applications “,2010

3.1 Visualization Techniques for spatial data:

Spatial data visualization, which corresponds to the field of scientific visualization, assumes that the data has an implicit or explicit spatial or spatio temporal attribute.

This constraint facilitates both the creation and interpretation of the visualization, as there is an intuitive, and often straightforward, mapping of the data attributes to graphical attributes of the entities conveying the information in the visualization.

As our visual perception system is constantly receiving and interpreting imagery of the physical phenomena surrounding us, it is quite natural to process the same form of imagery on a computer screen.

In creating a visualization of spatial data, we must decide what spatial attributes of the data will map to the spatial attributes (locations) on the screen.

This can involve many forms of transformation, including scaling, rotation, translation, shearing, and projection.

Once the spatial attributes are accommodated, other attributes and values associated with the data must then be mapped to other components of the visualization, whether it is an attribute such as color or texture or the size or shape of a graphical entity.

One-Dimensional Data:

One-dimensional spatial data is often the result of accumulating samples or readings of some phenomenon while moving along a path in space. For example, a drill-hole sample will contain mineral content and ore grade information based on the distance from the top of the drill-hole. This sort of sampling is often referred to as a probe when exploring structures of higher dimensions.

Given a one-dimensional sequence of univariate data (only one value per data item), we can map the spatial data to one of the screen dimensions and the data value itself to either the other screen dimension (to form a line graph, see Figure) or to the color of a mark or region along the spatial axis (to form a color bar). The data needs to be scaled to fit within the range of the display attribute (either number of pixels or number of colors). Parts of the display space or color range might be reserved for other aspects of the visualization, such as the axes, labels, and key, so the most general structure for an algorithm to generate such a visualization will use parameters for the bounds of both the data and display spaces.

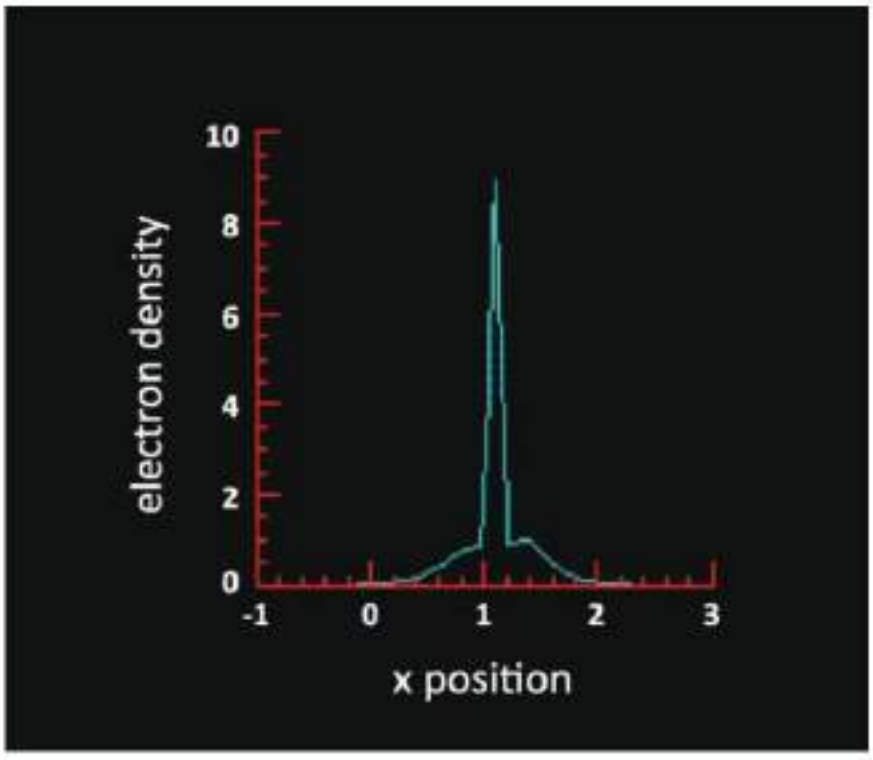

Fig : A line graph of a 1D sequence of data values

Assume that (datamin, datamax) are computed as the minimum and maximum values for the data, and datacount indicates the number of data points to be displayed (this could be all of the data, or just a selected subset). Also assume the section of the display that will hold the visualization is defined by the rectangle (xmin, ymin, xmax, ymax). To generate a line graph from an array called “data,” we could use the following code (assuming the drawing command draw-Line exists).

```
Draw-Line-Graph(data, dataCount, xMin, xMax, yMin, yMax)
1 dataMin ← computeMin(data, dataCount)
 2 dataMax ← computeMax(data, dataCount)
 3 xFrom ← xMin
 4 yFrom ← worldToScreenY(data[0], dataMin, dataMax, yMin,
5 yMax)
 6 for i ← 1 to dataCount
7 do xTo ← worldToScreenX(i, dataCount, xMin, xMax)
8 yTo ← worldToScreenY(data[i], dataMin, dataMax,
9 yMin, yMax)
10 drawLine(xF rom, yFrom, xTo, yTo)
11 xFrom ← xTo
12 yFrom ← yTo
 worldToScreenX(index, dataCount, xMin, xMax)
return (xMin + index ∗ (xMax − xMin)/dataCount)
worldToScreenY(value, dataMin, dataMax, yMin, yMax)
return (yMin + (value − dataMin) ∗ (yMax − yMin)/(dataMax − dataMin))
```

Note that each transformation is a combination of offsets and scaling that we use to convert values in one coordinate system into another. This operation is used heavily in computer graphics, and we will use variants on it numerous times in this book. It is derived based on the observation that a point within a given range can be transformed to a point within a different range by computing the relative position of the point within the range, e.g., what percentage of the distance from one end of the range to the other is the selected point. Mathematically, this is given as

$$(Ai - Amin)/(Amax - Amin) = (Bi - Bmin)/(Bmax - Bmin)$$

where Ai is a location in the range [Amin → Amax] and Bi is the corresponding location in the range [Bmin → Bmax].

Two-Dimensional Data:

Data with two spatial dimensions get visualized predominantly by mapping the spatial attributes of the data to the spatial attributes of the screen. The result can be one of the following visualizations:

1. **An *image*** results if a single data value at each location is mapped to color and all intermediate pixels are colored via interpolation .
2. A ***rubber sheet*** results if the data, whether regularly or irregularly spaced, is mapped to the height of a point in three dimensions, with the points triangulated so that a surface can be formed.
3. ***A cityscape*** is formed by drawing three-dimensional objects (generally boxes) at locations on a plane, where the data can control the attributes of the graphical objects (i.e., height and color).
4. ***A scatterplot*** results if, at each location on the plot, the data value(s) control the color, shape, or size of a marker. Note that unlike for images, no interpolation is performed.
5. ***A map*** results if the data contains linear and area features, as well as point objects. A linear feature, such as a road or stream, is represented as a sequence of connected coordinates, which are plotted as a series of line segments. Area features, such as a lake or political boundary, are generally represented as a closed contour, a set of coordinates where the first and last points are the same.
6. ***A contour or isovalue map*** conveys boundary information extracted from an image depicting a continuous phenomenon, such as elevation or temperature. The term isovalue means “single value,” and thus a contour on such a map indicates the boundary between points above this value and points below the value. It can be formed by considering two-by-two arrays of adjacent data values as the corners of a rectangle or square, and generating edges across this rectangle when one or more values are on the opposite side of the isovalue from one or more of the others.

Three-Dimensional Data:

As with two-dimensional data, **three-dimensional spatial data may be either discrete samples of a continuous phenomenon or a structure best described via vertices, edges, and polygons.** In reality, many visualizations of science and engineering data contain a combination of these data representations, such as air flow around a wing or stress attributes of a mechanical part.

1 .Visualizing Explicit Surfaces:

An explicit surface is one that has been defined in one of two ways:

1. a list of three-dimensional vertices, a list of edges, i.e., connections between the vertices (specified as a pair of indices into the vertex list), and a list of planar polygon patches (usually specified as a fixed or variable length list of indices into the edge list);
2. a set of parametric equations for defining the x-, y-, and z-coordinates of points on the surface, along with an interconnection strategy (e.g., a triangular or rectilinear grid) for computing the edges and patches. The step size of the parameters can be used to control the smoothness of the curved surface.

2.Visualizing Volume Data:

As pixels are to two-dimensional visualization, voxels, or volume elements, are to three-dimensional visualization. Volume data is generally a sampling of a continuous phenomenon, and can be either acquired via sensors (e.g. tomographic data sets) or generated via simulations (e.g., computational fluid dynamics). In each case, we have one or more data dimensions with regular or irregular positions, and the goal is to convey to the viewer the structure, patterns, and anomalies within the data.

Most approaches to visualizing volume data fall into one of the following categories :

Slicing techniques. Using a cut plane, either aligned with an axis or arbitrarily oriented, probe the data to extract a two-dimensional slice of the data, and then use one of the two-dimensional spatial data visualization methods.

Isosurface techniques. Given a user-specified value, generate a surface description and visualize it using one of the explicit surface visualization

techniques.
Direct volume rendering. Either cast rays into the volume and compute a pixel value based on the data encountered by the ray, or project each voxel onto the projection plane using some method of accumulating effects on pixels.
Direct Volume Visualization Techniques. Direct volume rendering means that no three-dimensional polygons are created for use with traditional graphics rendering techniques. Rather, pixels in the resulting image are computed on an individual basis, either by casting rays through the pixel through the volume, or by projecting voxels onto the plane of projection.

The basic process of rendering a volumetric data set starts by transforming the positions of the voxels into the viewing coordinate system, following the same procedure used in the traditional three-dimensional graphics pipeline. This may or may not include perspective distortion. The viewer must specify a view reference point (the origin of the plane of projection), a view direction (the normal to the plane of projection), the height and width of the image to be projected on the plane of projection, and, for a perspective projection, the distance from the camera to the plane of projection. The reader is referred to textbooks on computer graphics for details of this process.

Once the voxels have been positioned, we have the option of either:

forward mapping—project each voxel onto the plane of projection and determine which pixels will be affected and in what way;

inverse mapping, also called ray casting—send a ray from each pixel in the plane of projection through the volume, sampling values along the ray, and determining the resulting value for each pixel.

Combining Techniques:
Many effective visualizations are actually combinations of two or more of the techniques . Each technique has its strengths and weaknesses in terms of the types of information it can or cannot effectively visualize, so

a combined visualization, as long as occlusion is minimized, can generate results that are the sum of the strengths. At the same time, more and more problems require the simultaneous analysis of multiple data sets to arrive at an informed result.

A. Slice Plus Isosurface:

The isosurface is capable of conveying surface structure, which is difficult to obtain from volume slicing, even with animation of the slice position. However, the isosurface only provides information on a single value within the entire volume, with no indication of the distribution of other values or the gradient (rate of change) of the selected value at different locations.

The slice provides very detailed two-dimensional information, especially with an appropriate choice of color assignments. It can convey to the user the regions of relative uniformity, as well as those exhibiting significant change. Another advantage is that the image slice can convey nested regions of a particular value range, while the isosurface, in general, will only display the outer-most surface.

B. Isosurface Plus Glyphs:

isosurfaces are useful for conveying details of three-dimensional surfaces, but, in general, do not incorporate other aspects of the data. Glyphs , such as the popular arrow glyph, can be used to display the magnitude and direction of change within a data set, either as a gradient in static data or a flow in dynamic data. The glyphs may be positioned in close proximity to the isosurface, since these are known to be positions of interest, or the positions may be controlled separately

C. Rubber Sheet plus Contour Lines and Color:

In this example, we start with a rubber sheet to convey a two-dimensional field of values as a height field. This can reveal peaks and valleys found in the data, and by creating a virtual landscape, we can build on the users' intuition about interpreting the data in a topographic manner . We can augment this visualization by mapping color to the elevation, thus making it easier to identify widely separated regions of similar height . Finally, we can superimpose contour lines at certain levels, making the gradient information much more apparent.