ABSTRACT

KOCHERLAKOTA, SARAT MOHAN. Perception Driven Search Strategies For Effective Multi-Dimensional Visualization. (Under the direction of Christopher G. Healey)

Tracking and analysing large amounts of information in many different application areas is a critical problem. One approach to address this problem is the use of multi-dimensional visualizations to represent large datasets. Visualizations can be constructed effectively by the use of visual features and properties like color and texture. Our objective is to construct multi-dimensional visualizations using perceptually salient visual features which support rapid visual analysis and exploration of large datasets. We use a visualization system called ViA use to construct effective visualizations.

We present a search technique incorporated in ViA, that finds effective attribute-feature mappings to represent multi-dimensional datasets in a perceptually salient fashion. ViA evaluates the salience of attribute-feature mappings using evaluation engines. These evaluation engines also suggest hints that recommend how the mapping can be improved perceptually. The search technique we developed uses dataset properties and the hints generated by the evaluation engines to quickly and efficiently produce perceptually salient mappings.

Perceptual guidlines were established from studies and experiments on human perception. ViA works as a semi-automated visualization system that uses an effective search technique to find salient mappings. Applying ViA to practical datasets indeed proves the effectiveness of ViA. We think ViA can also produce salient visualizations in a variety domain areas since the guidelines for generation of effective visualizations are based on human perception.

Perception Driven Search Strategies For Effective Multi-Dimensional Visualization

by

SARAT M. KOCHERLAKOTA

A thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Department of
COMPUTER SCIENCE

Raleigh, North Carolina

2002

APPROVED BY:

_____ _____

_____
Chair of Advisory Committee

BIOGRAPHY

Sarat Mohan Kocherlakota was born to Mohan Kocherlakota and Girija Kocherlakota, in Bombay (now Mumbai), India. He received a Bachelor of Engineering degree in Computer Science and Engineering from the Mahatma Gandhi Mission's College of Engineering and Technology, Kamothe, India after which he worked for a few months with Larsen and Toubro, Bombay, India, and later also worked for a few months with an animation and special effects studio, The Fix, Bombay, India. Sarat is currently enrolled in the computer science masters program at North Carolina State University. He plans to continue his studies at North Carolina State University in the computer science doctoral program.

ACKNOWLEDGEMENTS

In life, there are certain moments that one hopes to be prepared for. Yet, when the moment does actually arrive, it completely overwhelms you. One such moment is right now, when I attempt to acknowledge the contributions and influences that have brought me thus far.

This thesis would not have been completed without the guidance and motivation I received at every stage from Dr. Christopher Healey. Throughout the years that I have been in the Masters program he has always demonstrated great concern and committment towards my progress, and indeed to the progress of all his students. I probably will never be able to thank him enough for all that he has done for me.

Dr. Thomas Honeycutt has been a strong influence in my life in the last couple of years. He opened doors for me with his wonderful intuitive ability to initiate the process of self-discovery in others. His timely insight and opinions have helped me tide many moments of doubt. I cannot thank him enough.

Dr. Robert St. Amant provided me with invaluable support especially in my early days of my Masters program when I was still new at NCState. I owe him much for all that he has done for me.

I could not have also been wherever I am pursuing whatever it is that I seek without the blessings and support of my parents, Mohan and Girija, my sister Mamata, her husband Prabhakar and their son (my nephew) Varun. I owe them more than I can possibly imagine.

I would have been lost without my friends, especially those who have been with me since my early school days. I have shared some of the best moments in my life with friends like

Pranav, Shaibal and Brijesh, to name just a few. Without their contributions and support, I would not have gotten very far.

My relatives here in the United States, innumerable as they are, have helped me tide many moments of loneliness with their love and support. In particular, I would like to thank my relatives here in North Carolina including my uncle and aunt, Prasad and Kumari and their children, Pradeep and Deepti for their love and affection, and for providing me a home away from home.

Brent, Laura, Jason, Jiae, Mark, Amit, Vivek, Mike Romeo, Alex, Micah, Nathaniel, Juan and several others, have always encouraged and supported me through many tough moments in these past few years and have given me many moments of happiness. Subhayu, Ravi, Rahul, Rohit Badlaney, Raoul, Rohit Sharma and so many others, have been so very generous to me, and their friendship and support has made my time at NCState memorable.

I cannot thank Shaileja Chopra enough for having supported and encouraged me all these years. She has lent purpose to my life, and made it more meaningful. I am truly in her debt.

I am also deeply indebted to the city of Bombay for giving me an identity, and to North Carolina State University for being an institution that I feel proud to be a part of.

My deepest gratitude goes out to all of those that I have been able to mention, and all those that I have unforgivably forgotten to mention. It is to all of them that I dedicate this thesis to.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Visualization is the area of science dealing with generation and presentation of information in effective and easy-to-understand representations. Various definitions exist for visualization. The Oxford English Dictionary describes it as “making visible, especially to one’s mind, things not visible to the eye.” The Webster’s Dictionary describes visualization as “forming a mental image of something not present to the sight, an abstraction, etc.”

With respect to computer graphics, one of the more specific definitions describes it as “the use of computer imaging technology as a tool for comprehending data obtained by simulation or physical measurement.” [HM90] Another definition describes it as “techniques that allow scientists and engineers to extract knowledge from the results of simulations and computations.”

The approach proposed in this thesis to address the need for effective visualization is a semi-automated perception driven visualization tool called ViA. ViA can be used for exploration and

analysis of multi-dimensional or multi-variate datasets. The perception driven nature of ViA allows multiple layers of data to be presented simultaneously within a single image, without undermining the user's ability to easily comprehend the information in the representation.

1.1 Need For Visualization

Although small, self-contained collections of data in its raw form of numbers and text is perhaps readable to most people, the vast amount of information being generated in various areas, makes it imperative that we devise some ways to track and comprehend it. Experts in these domains may be knowledgeable about the information they need to analyze and explore. Yet, absorbing and comprehending vast amounts of information, especially in situations where time is a critical constraint, is an enormous and difficult task. For instance, in medical scenarios critical life-saving decisions by doctors and surgeons need to be made from information generated by medical scanning devices and systems. Another example is meteorological data that contains critical information relating to tracking potentially life-threatening storms and cyclones. In such scenarios, presenting information effectively aids in making an informed decision in a short period of time.

Visualizing information is very useful for analysis and exploration of vast sets of information, and for extracting knowledge from these datasets efficiently and quickly. Presenting information using line and bar graphs, pie-charts or even pictures is widespread for analysis and exploration, and for understanding relationships between various sets of data. It is safe to say that a single picture can express a thousand words. We can further say that the purpose of

visualization is to construct such representations that convey large amounts of information to humans more easily than plain words and numbers.

1.2 Scientific Visualization

As with our definition of visualization, scientific visualization is the representation of scientific collections of numbers, strings, and datasets using one or more visual features (e.g. color, texture or motion). Images constructed using visual features allow visual exploration and analysis to be conducted.

Imagine a dataset $D = \{e_1, .., e_n\}$ containing n sample points, or data elements, e_i. A multidimensional dataset represents two or more data attributes, $A = \{A_1, ..., A_m\}$, $m > 1$. The data elements encode values for each attribute: $e_i = \{a_{i,1}, ..., a_{i,m}\}$, $a_{i,j} \in A_j$. A data-feature mapping converts the raw data into images that can be presented to a viewer. Such a mapping is denoted by $M = (V, \Phi)$, where $V = \{V_1, ..., V_m\}$ is a set of m visual features V_j selected to represent each attribute A_j, and $\Phi_j : A_j \rightarrow V_j$ denotes the mapping of the domain of A_j to the range of displayable values in V_j. Scientific visualization is thus the selection of M and the analysis of a viewer's ability to comprehend of the images generated by M. An effective M must produce images that support rapid, accurate and effortless visual exploration and analysis.

1.3 Need For Perceptual Visualization

Since scientific visualization deals with generating a mapping between datasets and visual features thus presenting information in a form understandable to users, one would think it necessary to find answers to some important questions:

- How does the human vision system perceive visual information?
- Are there visual representations or techniques that are easier to interpret than others?
- How much visual information can be presented in a single image, before it becomes difficult to absorb any additional data?

Finding answers to these questions allows us not just to better understand human perception, but also to develop strategies to incorporate these findings in techniques used for visualization. Perceptual visualization is thus a broad technique that harnesses aspects of human vision in order to generate effective representations of data.

Figure 1.1 demonstrates the visualization of a multi-dimensional dataset. Here, a weather dataset is represented using small glyphs (or perceptual texture elements) that vary their color and texture properties. These properties are used to visualize the different data attributes. Color represents *temperature;* bright pink and red strokes for hot temperatures to dark green and blue strokes for cold temperatures. Coverage represents *wind speed;* tightly packed strokes with little or no background showing through for strong winds to sparsely packed areas for weak winds. Density represents *pressure;* more strokes displayed in a fixed area of screen space for

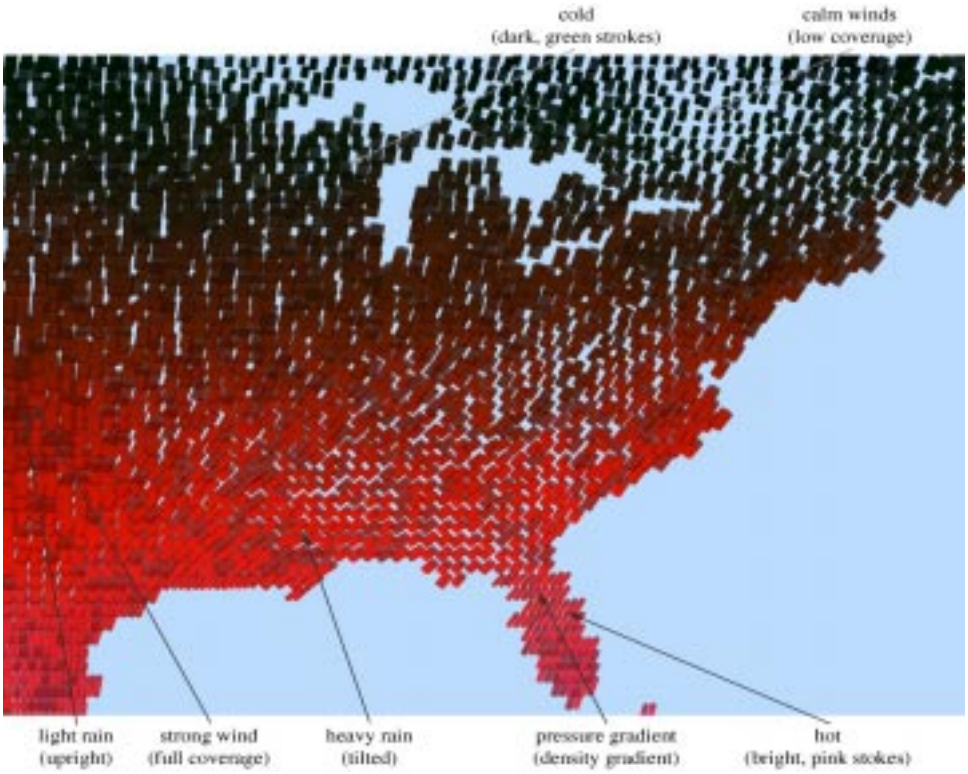


Figure 1.1: A visualization of a weather dataset using perceptual texture elements with *temperature* $\rightarrow$ color, *wind speed* $\rightarrow$ coverage, *pressure* $\rightarrow$ *density*, and *precipitation* $\rightarrow$ orientation

increasing pressure. Finally, orientation represents *precipitation;* vertical strokes for little or no rainfall to horizontal strokes for high rainfall.

The visualization shown in Figure 1.1 was constructed to maximize perceptual salience. The techniques used to visualize particular information determine the effectiveness of the resulting images. Certain techniques are more effective than others. By effective, we mean that the images are more easily absorbed, and extracting the underlying information or knowledge within the image does not require as much effort. Perceptual visualization techniques are also dependent on the type of data they are used for. For example, high spatial frequency data (i.e. data with sharp spatial changes in its value) is best displayed with luminance. Data with

discrete values, on the other hand, is best displayed with color. Understanding rules of perception allows us to choose appropriate mappings based on the properties of the datasets, and the analysis tasks the user wants to perform.

1.4 Multi-dimensional Dataset Visualization

Multi-dimensional visualization is the simultaneous representation of multiple attributes using appropriate visual features, within a single image. Such a representation attempts to present large sets of information within a single image to the user, thereby reducing the time and space needed to display the information when compared to representing each attribute in separate images. Representing multiple attributes simultaneously can also aid in understanding the relationships that might exist between different attributes.

However, representing information in this multi-dimensional manner raises the question: how much information within a single image, is too much information? It is essential that the representation of multiple attributes in a single image not exceed the user's capacity to comprehend the image and its underlying information.

1.5 Problems In Visualization

Perceptual visualization raises questions that must be addressed before effective, perceptually salient representations of datasets can be generated. These questions pertain to the nature of the datasets that need to be mapped to visual features. Some of these questions are:

- What type of data set is it?
- Are the data attributes discrete or continuous?
- Do the attributes have high or low spatial frequency?
- What kind of visual feature would best suit each attribute for representation?

In addition to the above questions, the presence of a user during multi-dimensional visualization raises its own issues [BCE+92]:

- What relative importance does the viewer attach to each attribute to be displayed?
- Do the visual features being represented simultaneously interfere visually with each other?
- How many attributes can be simultaneously displayed within the same image?

Answers to all of the above questions are essential for proper visual representation of multi-dimensional data. While some of this information could be provided by domain and visualization experts, what is needed is a comprehensive and systematic set of rules and guidelines that govern basic perceptual visualization, human vision and multi-dimensional dataset visualization in a way that allows the semi-automated construction of visualizations for a wide range of problem environments.

1.6 Flexibility

Many studies and research in scientific visualization seem to focus largely on the use of visualization in domain specific areas. This means that one tries to find the best form of visualization suited to a specific visualization problem. Moreover, these efforts are sometimes ad-hoc in their methods for building optimal visualizations. There is no guarantee that the techniques can be applied to other problem domains, even if these domains are similar in nature to the original dataset.

In order to be a domain independent tool (at least to some extent), a visualization system would need some kind of processing capability to evaluate a mapping's effectiveness for a given dataset and associated analysis tasks. Formalizing perceptual characteristics into rules and guidelines to direct the visualization process might allow us to provide such an evaluation. The need to search for a suitable mapping from a set of all possible mappings using such guidelines and rules becomes a critical task in this type of system.

1.7 Research Goals

The Visualization Assistant (ViA) developed in our research lab, is a perception driven tool, that incorporates studies on human cognition and human visual perception in order to generate perceptually salient mappings between datasets and visual features. Given the ability to evaluate a mapping for perceptual salience, this thesis studies ways to search intelligently through the search space of all possible mappings to locate those mappings which best meet the per-

ceptual guidelines, analysis needs, and dataset characteristic directing the search process. The research goals of this thesis:

1. Identify the appropriate information about a dataset in the user's exploration and analysis needs to constrain a search which satisfy these initial visualization goals.

2. Supplement the evaluation process with additional information that can be used to decide how to improve weak components in the data-feature mapping.

3. Use evaluation results to rapidly and effectively identify high-quality mappings based on the dataset's properties and the user's analysis requirements.

In summary, we incorporate and study appropriate search strategies within the visualization tool to return a set of optimal data-feature mappings from a set of all possible mappings for a generalized dataset and analysis tasks, in an efficient and effective manner.

In the following chapters, we will discuss not just perceptual visualization, and related efforts, but also the working of ViA, in particular the search process for finding suitable perceptually salient mappings between datasets and visual features, for multi-dimensional dataset visualization.

Chapter 2

Perceptual Visualization

The use of perceptual guidelines for designing displaying multi-dimensional datasets is aimed at producing perceptually salient images or displays to represent complex information spaces in ways that our human visual system can easily comprehend. Hence, rapid, accurate and effortless visual exploration of the representations are crucial to the effectiveness of a visualization system [HE99]. In order to be visually effective in this way, research in visualization has utilized psychophysical studies to understand and harness human visual attention.

2.1 Preattentive Vision

Visual information is processed in two ways by the human vision system. The first could be labeled as "controlled processing." The information is dealt with in detail, and capacity to assimilate information is low. Focus or attention can be easily inhibited. Since attention is conscious and is low capacity. The manner in which tasks are carried out is serial in fashion

with attention focusing on one task at a time.

By contrast “automatic processing” or “preattentive processing” is a fast, unconscious manner of perception. Capacity to assimilate information is independent of the load on vision and is thus very high, attention is less critical and many tasks are done in parallel. A simple example of this distinction is the use of controlled processing to perceive textual information, as compared to automatic processing to perceive basic color and texture patterns.

The dual abilities of human vision to carry out focused processing as well as to complete automatic tasks are characteristics that can be exploited in order to visualize information. One of the studies on preattentive vision conducted by Triesman [Tri85], focused on identifying the characteristics of preattentive vision in humans. A result of the study found that the human vision system can rapidly identify the presence or absence of a number of basic features including color, brightness, terminators of line ends, closure or blobness, tilt and curvature.

Figure 2.1 demonstrates some instances of the preattentive nature of the human visual system. A red circle in the midst of a group of blue circles is easily detected. Similarly, a red circle in the midst of red squares is also easily identified. However, the task of locating a red circle among a group of red squares and blue circles is much more difficult, suggesting that certain features can be preattentively detected, but preattentive integration of color and shape detection is not possible.

The above mentioned features were referred to as preattentive because their detection occurs rapidly and accurately presumably preceding focused attention. We now know that low-level human vision is much more complicated than originally proposed. Also, a viewer’s at-

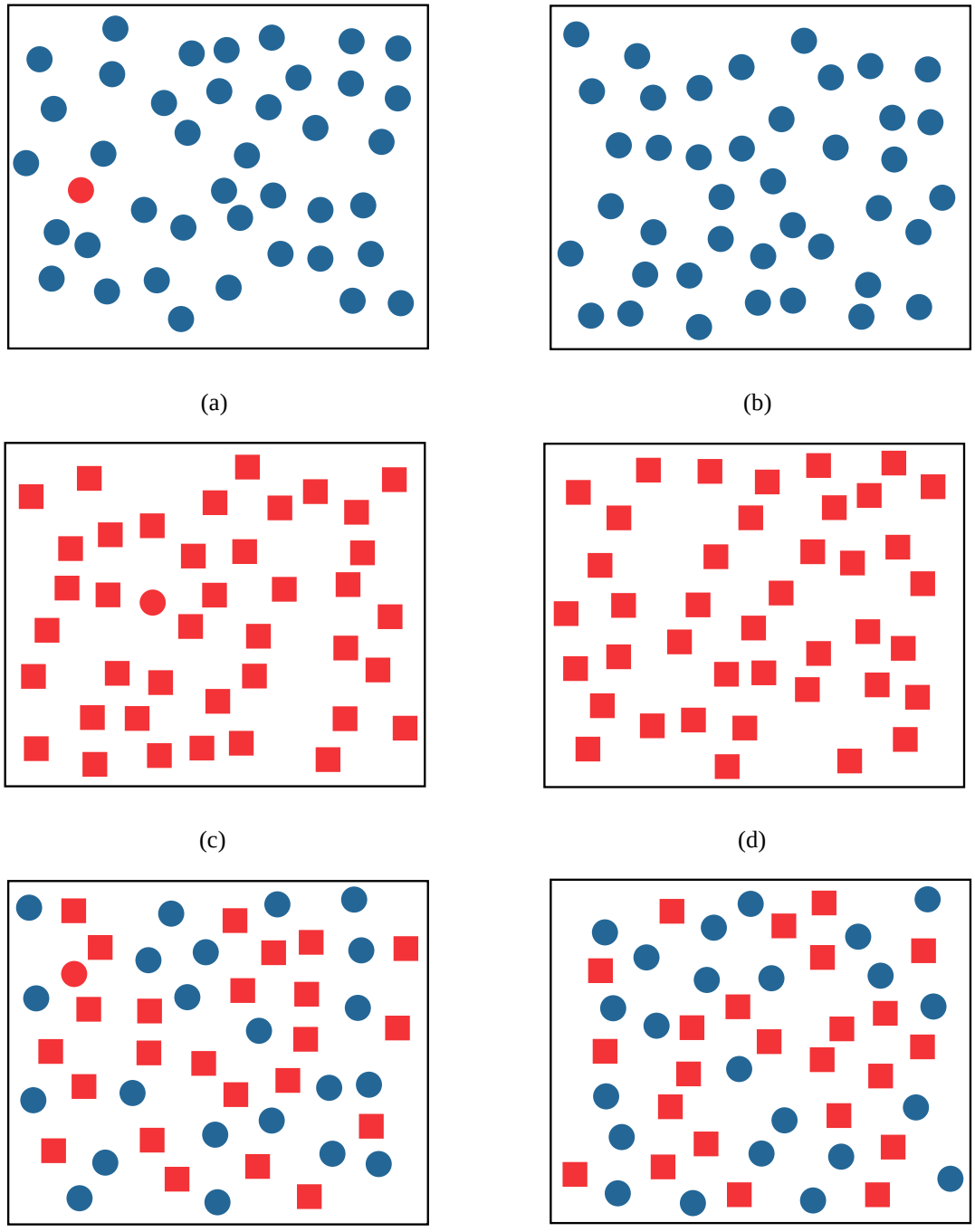


Figure 2.1: Examples of target searches: (a, b) demonstrates how the search for a red circle in a sea of blue circles is rapid and accurate, target is in (a). (c, d) demonstrates how the search for a red circle among a sea of red squares is rapid and accurate, target is in (c). (e, f) demonstrates how identifying another red circle among a sea of red squares and blue circles is significantly more difficult, target is present in (e), and absent in (f).

tention can have a significant impact on what low-level vision can detect. In spite of this, the term “preattentive” provides an intuitive indication of the type of visual processing offered by our cognitive visual system. We will therefore continue to use this terminology. The amount of time needed to detect preattentive features is largely independent of the total number of elements. The use of these features can be very helpful for performing a variety of exploratory visualization tasks. Some of these include search for target elements with specific preattentive features, identifying spatial boundaries between groups of elements, comparing elements, and tracking groups across time and space. In the following sections we discuss some of the preattentive visual features that are useful for visualization.

2.1.1 Color

Light is electromagnetic energy, and the range of the wavelengths from $400nm$ to $700nm$ denotes the visible spectrum of light. The perception of color is principally the response of the human visual system to the visible spectrum. Color perception is described as being made up of three quantities.

The first quantity known as hue is our perception of the dominant wavelength of reflected or emitted light. Hue allows us to distinguish between colors like blue, green, orange, purple and yellow. The visible spectrum spans multiple named color regions from violet, indigo, blue, green, yellow, and orange to red.

The second quantity called saturation, describes as how much white is mixed with a pure color. Saturation is our perception of the excitation purity of light. It refers to the mixture of

white light with pure light. A completely pure color is considered one hundred percent percent saturated, and is said to contain no white light. Red is a pure color and is thus highly saturated, while pink is a mixture of red with white light, and is hence considered relatively unsaturated. Mixing white light in varying quantities with a pure color reduces to the saturation of the color from one hundred percent to a possible zero percent saturation, which is seen as a shade of gray.

The third quantity is referred to as lightness. Lightness signifies our perception of the intensity of the light. Intensity of light is often defined as luminance. While lightness is used to signify the perceived intensity of light emanating from a reflected object which is not self-luminous, perceived intensity of light radiating from light sources (self-luminous objects) is referred to as brightness.

The back of the retina in the human eye is made up of two mechanisms to perceive light: rods , which detect luminance, and cones, which detect hue. With respect to the cones, there are three kinds depending on the type of photosensitive pigments within the eye.

The first kind is sensitive to the color blue , with a maximum sensitivity or peak response at a wavelength of $440nm$ of perceived light. The second kind of cones are those photopigments sensitive to color green, with a peak response at a wavelength of about $545nm$. And the third kind of cones are those sensitive to color red, with a peak response at a wavelength of about $580nm$. This theory of the response of different types of cones in the retina to light of different wavelengths (blue, green and red), is known as the tristimulus theory.

Variations in any or all the three quantities of hue, saturation and brightness produces many

different colors. The human eye itself can distinguish between millions of different colors. With luminance as a constant and varying between hues that are fully saturated, the human eye has been proven to be able to distinguish between about 128 fully saturated hues. The human eye is less sensitive to hue changes in less saturated light. [FvF+93]

Efforts to quantify colored light have led to various theories surrounding generation of colors. One theory that suggests that colors can be generated or described using mixtures of red, green and blue is useful in computer graphics, as the RGB (red-green-blue) model This model is used to display color on standard computer displays that use color cathode ray tubes (CRT). While combinations of red, green and blue can be used to produce many different colors, they cannot produce the entire spectrum of colors that are visible to the human eye.

Other models in computer graphics also utilize the tristimulus theory in generating colors. The CMY (cyan-magenta-yellow) model uses the three colors that are the complements of red, green and blue, respectively. While the RGB model defines color as what is added to blackness, the CMY model defines color as subtracting from white light. CMY is a useful model for hardcopy devices like ink-jet printers and plotters.

The YIQ and the HSV models are also utilized in computer graphics. YIQ is used to encode color NTSC television transmissions. Y roughly corresponds to luminance, while I and Q encode chromaticity (i.e. hue and saturation). This model was designed to be backwards compatible with the original broadcast standard from black-and-white televisions.

HSV attempts to provide a more intuitive color model by using three axes representing hue, saturation and value (or lightness). This makes it easier for users to select desired colors by

providing understandable color dimensions. As with RGB, both models' gamut cover only a subset of the entire visible color spectrum.

2.1.2 Color Selection

The CIE (*Commission Internationale de l'Eclairage*) model also uses a three-dimensional approach to defining colored light. This model is an international standard established in 1931. Although, three primary colors are defined in this model, these do not correspond to "real" colors from the visible color spectrum. This is due to the fact that no combinations of three visible colors (such as the so-called "primary colors" red, green and blue) can produce all colors in the visible spectrum. However, using combinations of the imaginary primaries X, Y and Z specified in this model allows us to reproduce all possible colors.

2.1.2.1 CIE LUV

The CIE LUV color model (proposed in 1976) specifies color using three dimensions L^* (encoding luminance), u^* and v^* (together encoding chromaticity). Two useful properties allow us to control perceived color difference. Colors with the same L^* are considered "isoluminant", since they have same perceived luminance. And, Euclidean distance and perceived color difference (specified in ΔE^* units) can be roughly interchanged. This means color difference between two color stimuli x and y which are positioned in CIE LUV at (L^*_x, u^*_x, v^*_x) and (L^*_y, u^*_y, v^*_y) can be measured as:

$$\Delta E^*_{xy} = \sqrt{(\Delta L^*_{xy})^2 + (\Delta u^*_{xy})^2 + (\Delta v^*_{xy})^2} \tag{2.1}$$

Healey and Enns [HE98] used the CIE LUV model to measure color distance (or perceived color difference). They used two additional criteria, linear separation and color category to further refine their ability to determine the perceptual balance of a set of colors[Hea98]. Color distance is defined as the Euclidean distance between different colors as measured in a perceptually balanced model. Linear separation is defined as the ability to separate a target color all other, from non-target colors with a single straight line in the color model being used. Color categories are defined as uniquely named color regions that subdivide the color model. Each color can then be matched to the named region it occupies.

Experiments conducted to ascertain the effect of linear separation [D'Z91] investigated the human response to the task of finding a target color in a background of non-target colors. Results showed that when the target color could be separated by a straight line from the non-target colors in the color space, the time required to ascertain presence or absence of the target color was constant time, and independent of the total number of elements on display [Hea98]. If the target was collinear with the non-targets, the time required to identify presence or absence was no longer constant, but rather was linearly proportional to the number of elements on display. Further experiments in this area [BJC96] confirmed that the linear separation effect applies to colors from the entire visible color domain. Finally, experiments in color category [KUU95] concluded that searching for a target in a background consisting of a non-target color that occupied in the same region as the target is significantly more difficult that searching for

a color target in a background of a non-target colors which belong to a different named color region.

These experiments relate closely to understanding the perceptual properties of colors and how to effectively control color during visualization design. Some of the important questions that need to be addressed in order to use color in perceptual visualization systems include:

1. Based on human perception of color, what kind of colors (or color categories) can be used for visualization purposes?

2. What is the maximum number of colors that can be used for purposes of visualization without hindering the analysis of the data being represented?

Experiments conducted to address these questions [Hea96] showed that up to seven different isoluminant colors can be displayed simultaneously while still allowing the rapid and accurate identification of any of the seven colors. The study found that the time required to identify any one of these colors (shown simultaneously in the presence of the remaining six) was largely independent of display size, which suggested that target detection in this case occurs preattentively. However, the study also concluded that the selection criteria mentioned earlier, i.e. color distance, linear separation and color category all need to be considered in choosing colors that could be rapidly detected. Violating of any of the criteria produced a significant reduction in user speed and accuracy.

The CIE LUV model provides rough uniformity in choosing colors that can be perceptually differentiated. Within the model, the color distance of two colors corresponds approximately to the perceived color difference between the colors. Efforts on controlling color by Ware [WK95]

produced a system that allows users to minimize color surround effects and reduce perceptual errors to a constant value along a continuous color scale. This method was used to build scales that guarantee that color detection remains effective even as neighboring colors are changed within the visualization. Finally, Rheinigans and Tebbs [RT90], worked on constructing color scales interactively by allowing the users to trace a path through a perceptually balanced three dimensional color model. Such a tool allowed users to vary the visualization of an attribute by mapping its range of values to colors along the color path.

2.1.3 Texture

Aggregates of many small visual elements can be used to develop visual textures [JB83]. Such textures can be represented by either single elements like colored dots or circles, or complex patterns. We are interested in studies dealing with the use of textures for representing multi-dimensional datasets.

The use of textures in multi-dimensional visualization must address a various issues:

1. Identifying the fundamental visual properties of textures that can be used to encode information.

2. Identifying ways in which these features can be combined together with hue and luminance to visualize multi-dimensional data.

According to a DOE/NSF joint panel on visualization, harnessing aspects of perception is critical to increasing our ability to represent and absorb information. Investigations and research in computer vision and psychophysics have studied texture patterns, and have identified many

properties of textures that can be used as “perceptual dimensions” in visualization environments. These properties can be used to represent multiple data attributes together in a common texture pattern. For example, size (or height, length and width), orientation and density of individual texture elements can be varied to represent a corresponding data element’s attribute values.

Triesman’s [Tri85] studies on preattentive vision and their investigations revealed the importance of textures in preattentive vision. The results identified features like size, closure or blobness and tilt and curvature preattentively detected. Julész [JB83] also determined elongated blobs (rectangles, ellipses or line segments with specific size, angular orientation and color), ends of line segments (terminators), and crossings of line segments were visual features that were detected by the low level visual system. This study led to the development of the texton (texture element) theory.

Although texture patterns developed using textons proved that preattentive vision can identify differences in texture patterns, certain conditions can inhibit the use of preattentive vision. One condition that interfered with preattentive detection was the size of the textons. When the textons were relatively small and the spacing was sparse, preattentive detection was hampered. Other cases where preattentive detection failed was when texture patterns were created using combinations of textons. Combinations in texture shapes and form, they concluded, could only be detected with focussed attention of the display.

Rao and Lohse [RL93a, RL93b] also conducted studies on texture and texture dimensions. They experimented in identifying texture dimensions by asking viewers to classify pictures

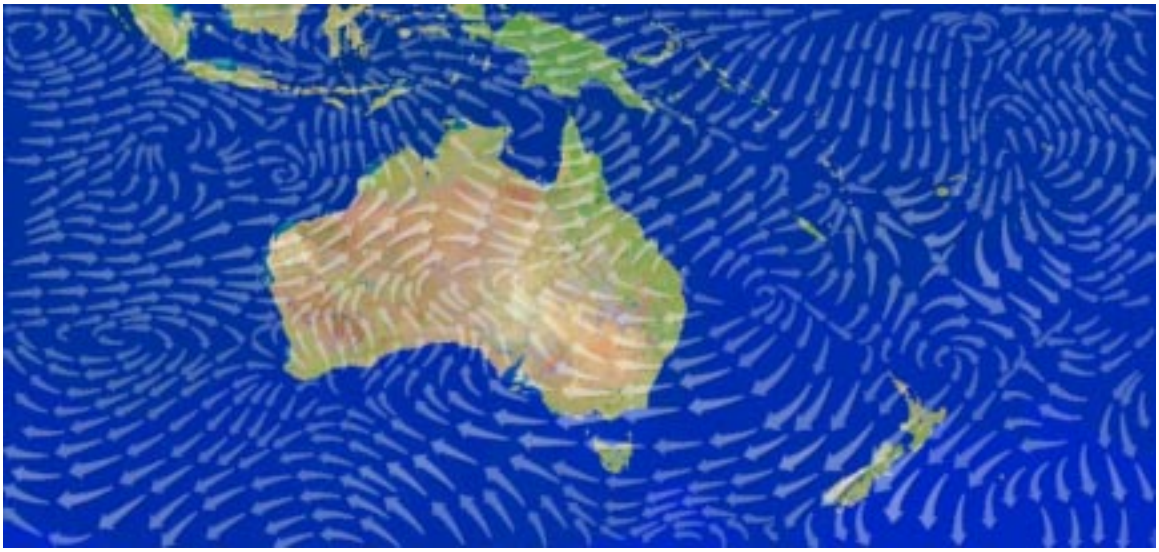


Figure 2.2: A visualization using streamlines (arrows that can vary their direction, length, and size) of a two-dimensional vector field of wind patterns over Australia and Indonesia

containing different kinds of textures. They discovered three main texture dimensions that were used by the viewers to differentiate the images. These were: repetitive patterns as opposed to non-repetitive patterns (regularity), linearly oriented patterns as opposed to circular patterns (directionality), and simple patterns as opposed to complex patterns (complexity).

Ware and Knight [WK95] applied results from vision research in synthesizing visual textures for purposes of visualizing information. They constructed texture patterns using Gabor filters. The size, orientation and contrast of the Gabor elements were controlled by using the attribute values in an underlying datasets.

Turk and Banks [TB96] proposed streamlines (directed arrows) to visualize two dimensional vector fields. They introduced a technique which used an energy function to guide the placement of the streamlines. The positions and lengths of the streamlines control the visual density of the image. The images in turn represented visualization of flow fields. Attributes

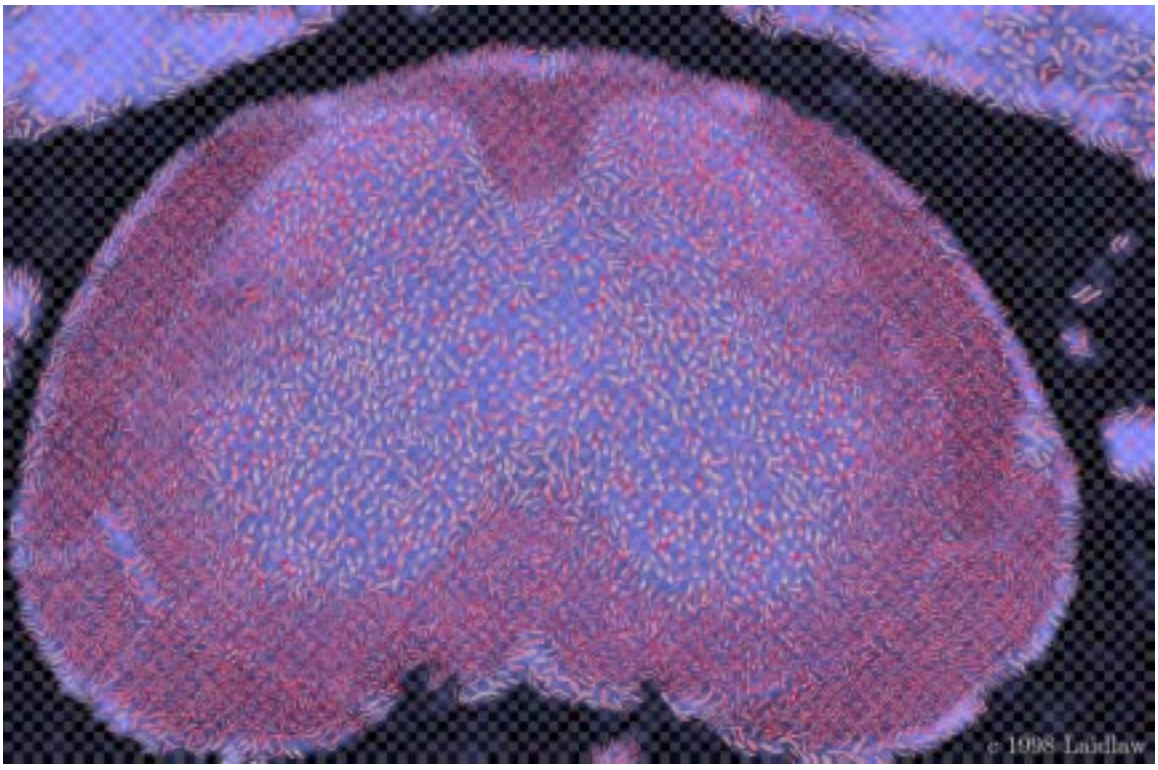


Figure 2.3: A visualization of a mouse spinal cord using painterly ellipses to represent the data; the image allows variation of the underpainting lightness, the spacing of an underlying checkerboard pattern, and the major-minor axis ratio, the major axis direction, the red saturation, and the texture frequency of each ellipse

of the flow data elements like direction and magnitude are represented by using arrowheads for the streamlines. Figure 2.2 shows a visualization using streamlines proposed by Turk and Banks to visualize wind patterns.

Laidlaw et al [LAK+98] used ellipse-shaped glyphs to build painterly visualizations of second-order diffusion tensor fields. Figure 2.3 shows an example of this technique, the visualization of a 2D slice through a mouse spinal cord. Each sample point in the dataset is represented with an ellipse and background image properties to display up to six separate data values. The visual features being applied include lightness of the underpainting, the spacing of

an underlying checkerboard pattern, and the major-minor axis ratio, major axis direction, red saturation, and texture frequency of each ellipse.

Kirby, Laidlaw and Marmanis [KML99] used a similar technique of graphical elements representing artists' brush strokes to construct discrete and continuous visual elements arranged in multiple layers. Such representations were used to depict two-dimensional flows in fluid mechanics. These representations were multi-dimensional as they depicted velocity, vorticity and other mathematically derived quantities.

Grinsten [GPW89] furthered the idea of textons in a system called EXVIS. This technique could also be referred to as a "glyph-based" visualization method. "Glyphs" are graphical icons which have individual features like size, shape, orientation and color that can be controlled and varied by attributes of the data elements they are chosen to represent. These techniques are hence extremely useful in the display of multi-dimensional datasets. EXVIS uses stick-man icons as glyphs to construct texture patterns for the display of multi-dimensional data elements. Individual properties of the data are represented by varying the orientation and the spatial positioning of the icons. The texture patterns were used to display spatial coherence in multi-dimensional data.

Healey and Enns [HE98] discussed the use of perceptual texture elements or "pexels" to represent the multi-dimensionality of individual data elements. Pexels vary both color and texture dimensions and arrangements to visualize the underlying data. Figure 2.4 shows a visualization that uses pexels. Different attributes are bound to different pexel properties. Attribute values within each data element are then used to modify the corresponding pexel's appearance.

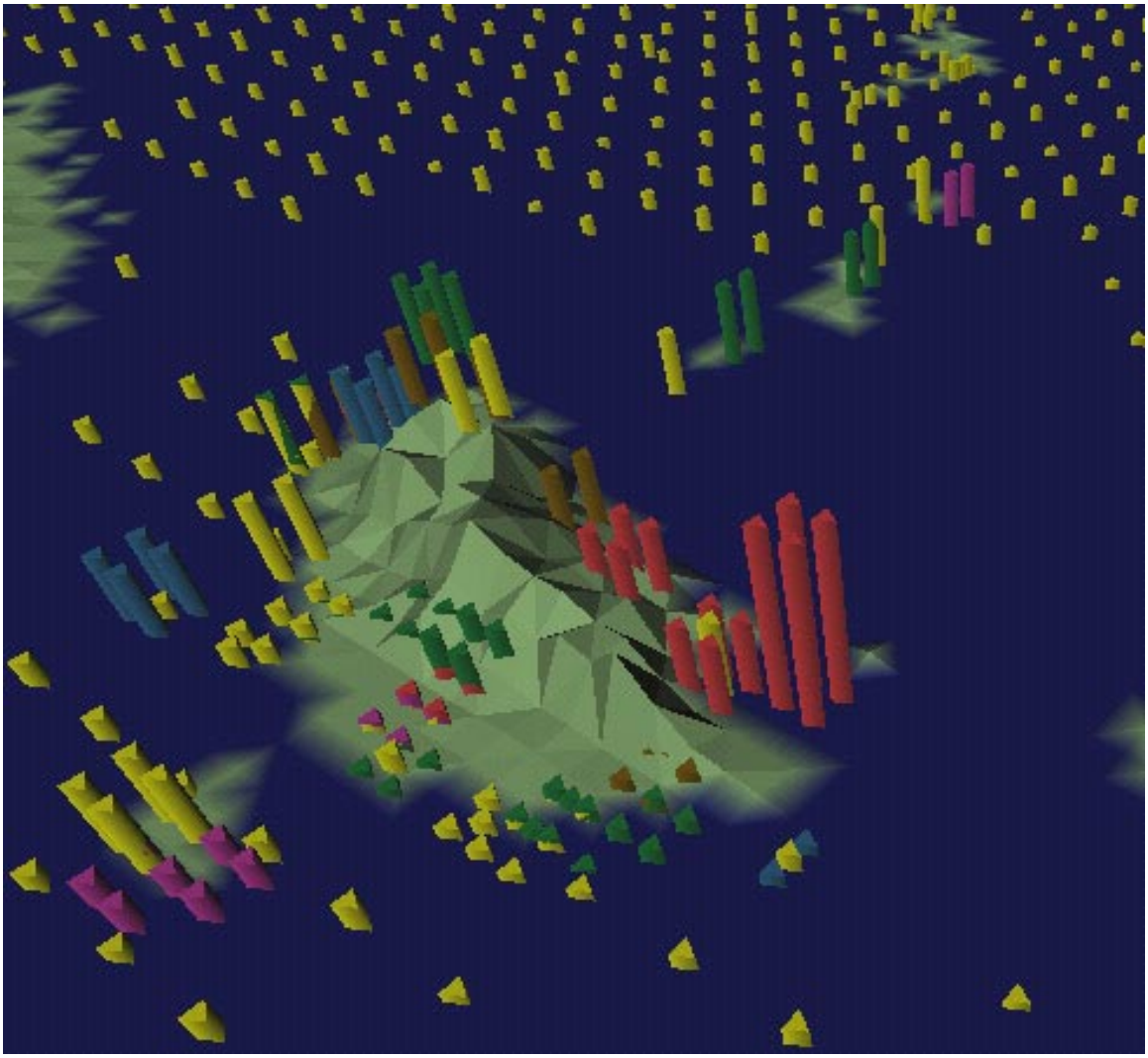


Figure 2.4: A visualization using pexels to track typhoon Amber (August 28, 1997) making landfall on the island of Taiwan. *wind speed* is visualized using height, *pressure* using density, and *precipitation* using color. Neighboring ocean regions without pexels (*i.e.* with missing data) represent regions obscured by clouds

Available visual features include:

- *color:* the color of a pexel,
- *height:* the height of a pexel in three dimensions (or its size or area in two dimensions),
- *density:* the spatial packing of the pexels (in two or three dimensions),
- *regularity:* the underlying spatial placement of the pexels, either regular in a repeating grid-like fashion or randomly jittered across the display, and
- *orientation:* the tilt of a pexel in three dimensions (i.e. its polar orientation), or rotation of the pexel in two dimensions.

EXVIS and pexel-based methods combine the use of texture patterns with color properties in developing effective multi-dimensional data visualizations. In the next chapter, we address the issue of choosing appropriate visual features to represent the attribute values of data elements to generate perceptually salient visualizations.

Chapter 3

Related Works

In the previous chapter we examined the concept of perceptual visualization and discussed how it effectively utilizes human vision and human perceptual abilities. In this chapter, we shall focus mainly on the development of multi-dimensional data visualization systems, and some of the research that has been carried out in this area. In particular, we shall look at studies that allow us to visualize multi-dimensional information simultaneously in a single image, and research on developing rule-based systems to aid in generating effective and perceptually salient mappings between data attributes and visual features. Furthermore, since our area of research deals with perceptual visualization systems, we shall focus on research that integrates visualization techniques with studies on human vision and cognition to achieve perceptual salient designs. Finally, we shall look at search techniques that assist in choosing optimal mappings for visualization from a search space of all possible mappings.

3.1 A Presentation Tool (APT)

Jock Mackinlay [Mac86] incorporated visualization tools in his Application-Independent Presentation Tool (APT). APT was developed for designing graphical presentations (such as bar charts, scatter plots and connected graphs) of datasets, in a two-dimensional form. Though limiting presentations to a two dimensions, the early research focused on generating rules for visualization that could be used in a formalized, rule-based fashion in order to generate appropriate mappings between datasets and visual features.

3.1.1 Expressiveness And Effectiveness

The underlying principles of this rule-based system were two very important criteria: expressiveness and effectiveness.

The expressiveness criterion dealt mainly with the issue of expressing information in a corresponding graphical language. The graphical language generated by APT was a collection of tuples. These tuples were composed from two separate sets: a set of graphical objects like points, lines and areas (encoded on the basis of their graphical properties: positional, temporal and retinal), and a set of locations where these graphical objects were represented.

The rule set that governed the generation of a graphical language to express information was combined with the use of an effectiveness criterion. The effectiveness criterion for the use of graphical objects was based on human perception. Graphical objects were ranked according to their perceptual effectiveness within a presentation. Objects that were more effective had a higher ranking enabling the system to display them more prominently in the presentation or

display.

The two criteria thus mentioned (expressiveness and effectiveness) are used to control the generation of a perceptually salient visualization. This process of visualization adopted by APT is encompassed in three steps:

1. *Partitioning*: where data is partitioned or classified and graphical objects are ranked or re-ordered based on their effect on the whole visualization.

2. *Selection*: where data attributes are mapped to visual features in a one-to-one fashion.

3. *Composition*: where the data attributes are composed in the same spatial area to construct the final visualization.

At every stage of this three step strategy, the expressiveness and effectiveness criteria are applied to guide the mapping process in order to make the final visualization perceptually salient and effective.

Mackinlay's work on measures of effectiveness of a visualization and the attempt to combine the various guidelines into applicable formal rules, has inspired many subsequent efforts in scientific visualization research. APT was developed with the strategy of carefully generating guidelines for the visualization process. Features incorporated within APT included the use of a graphical language to formalize the mapping of data attributes to visual features, and the incorporation of a set of perceptual guidelines and rules to be applied to the mapping process in order to produce perceptually salient mappings. All of these have served as cornerstones for many other efforts in the area of scientific visualization.

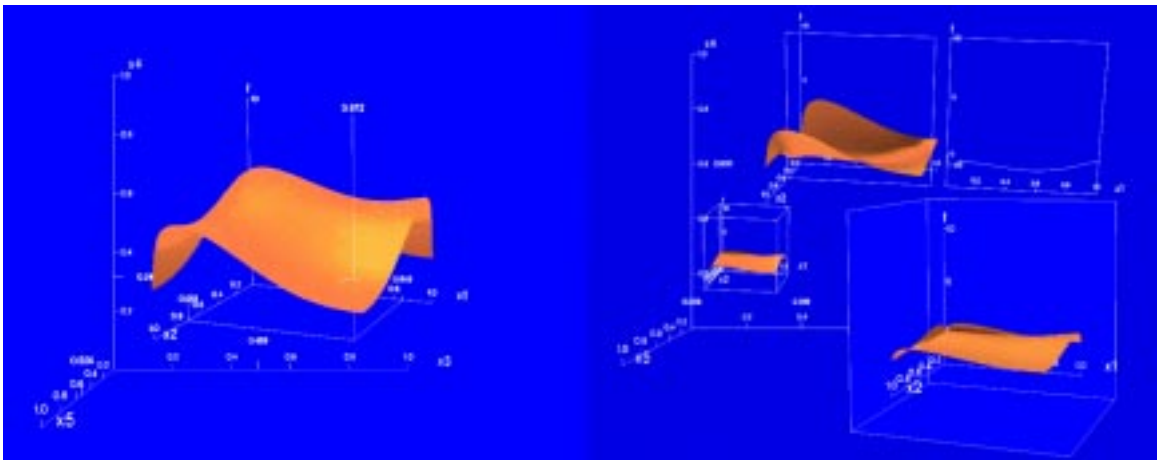


Figure 3.1: A visualization using the principle of worlds within worlds. Each data element in the graph encloses an inner world for attributes that were not represented in the outer world.

3.2 AutoVisual

One significant extension of APT is *AutoVisual*, developed by Beshers and Feiner [BF93] . AutoVisual differs from APT in that it can map datasets in a three-dimensional space. The rule system for AutoVisual is based on a worlds within worlds principle [BF93] for multi-variate dataset visualization. Each world is represented by a graph. This graph encodes a subset of the relations contained in the parent world. The subset that is represented is determined based on the position of the world's origin with respect to its parent world. Outer worlds comprise an encoding of data attributes to visual features. Each of the attributes represented in the outer worlds forms an anchor to an inner world with an encoding of additional attributes to visual features.

3.2.1 Worlds Within Worlds

Figure 3.1 illustrates the worlds within worlds principle. Worlds can be represented in one, two or three dimensions. AutoVisual allows user interaction at all levels. Rather than attempting to model every data attribute to a pattern of color or texture, AutoVisual relies on using lines, and surface plots, and height fields together with their worlds within worlds principle to generate effective interactive visualizations.

One-dimensional plots are represented by an interactor known as a dipstick. In the two-dimensional form, line graphs are the mode of representation. Three-dimensional representation is done using height fields and three-dimensional colored point clouds.

Inner worlds in AutoVisual encode what could not be represented in the outer worlds. This is due to the restriction of at most three-dimensions in each world. Every position in an outer world represents a data element with up to three of its attribute values A_i being visualized. Once selected, that element can be expanded to show addition attribute values A_j by generating a new inner world anchored at the given position in the outer world. As the anchor position is moved in the outer world to select different data elements the inner world's display automatically updates to reflect the new element's A_j value. In this way, *AutoVisual* allows representation of high dimensional data. Based on this ability, the visualization is known as *n-vision*.

An *n-vision* virtual world is built as a hierarchy of interactors. An interactor consists of four basic components:

1. *An encoding space set*: An encoding space is the region in which the graph or plot is rendered. Each encoding space allows up to three dimensions for representation or

visualization.

2. *A set of encoding objects*: Encoding objects constitute the various primitives that make up the graphical representation of the information. These may include, dipsticks in one-dimensional, lines in two-dimensional, and three-dimensional height fields and three-dimensional color-scale point fields.

3. *A selection set*: A selection determines which geometric subset of the parent interactor's encoding space is represented.

4. *A user interface*: a user-interface component consists of bindings between the user's actions and the interactor's properties. Interactors also have bindings for creating, copying and deleting child interactors. Interactions supported include translation, scaling and orientation of the visualization for user analysis tasks.

The use of AutoVisual to encode data attributes using a rule-based system with the graphical features follows up on Jock Mackinalay's effectiveness and expressiveness criteria. However in AutoVisual, the user specifies the visualization task rather than the visualization technique. AutoVisual also makes a difference between task operators and task selections. Task operators allow for user tasks to analyze the visualization. These tasks include exploration of the encoded space, directed search within the space, and comparison of the world with another subset of the encoded space. Task selections represents a subset of interest within a particular task.

The visualization is constructed as a directed acyclic graph of interactors, with each state representing an interactive visualization world. An unsuccessful path of interactors can be

undone using backtracking support provided by the system. Termination conditions applied to the visualization are represented by potential expressiveness and potential effectiveness criteria (which are basically similar to the expressiveness and effectiveness criteria as postulated by Mackinlay, modified to fit interactive visualization). Termination criteria are used to establish that the system has successfully achieved a satisfactory visualization.

3.3 VISTA

Senay and Ignatius [SI94] also expanded on Mackinlay's criteria to allow for three-dimensional visualizations and user interactions. They developed a visualization tool, VISTA, as an automated visualization system that also employed a rule-based composition technique for generating effective visualizations. Three components used to build the visualizations are:

1. *Data Unit*: which mainly comprised of user selections of data attributes and operators to be represented within the visualization.

2. *Design Unit*: which is decomposed into three steps. In the first step, data decomposition takes place. The partitioning divides data into two forms: quantitative or qualitative. Quantitative data is numerical in nature, with discrete or continuous values. Qualitative data are either nominal (like the names of people, or organizations), or ordinal (names from an ordered set, like the months in a year, or the days in a week). In the second step, a primitive is selected to visualize each partition using effectiveness and expressiveness criteria (similar to those described by Mackinlay, modified to work for three-dimensional

environments). This also forms a visualization hierarchy rooted at the most important data partition. The final design step, applies a composition module to combine pairs of component visualizations to form a final display of the data.

3. *Rendering Unit*: which creates an image based on descriptions provided by the design unit, using rendering algorithms. Composition rules are applied to unify the primitives. The composition rules are similar to the composition techniques by Mackinlay [Mac86], and are listed as follows: mark composition, composition by super-imposition, composition by union, composition by transparency and composition by intersection.

Figure 3.2 illustrates examples of the composition rules employed by VISTA. User interaction was also provided by the rendering unit through image manipulation operators. Users could select any component of the visualization, and modify its attributes. Viewing operations like rotation, translation, and zooming, among others, were also supported. In addition to all of these, users could change the composition rules that were applied to a data partition.

Unfortunately, the range of uses of the composition rules is extremely restrictive. Moreover, the geometric images developed by the marks and in VISTA is too broad in nature to produce a single appropriate image. Finally, VISTA does not appear to have a suitable evaluation process for adjudicating effectiveness criteria, although it claimed to support effectiveness. The application of this criteria appears to be too ad-hoc in nature to be applicable in a formal or generalizable fashion.

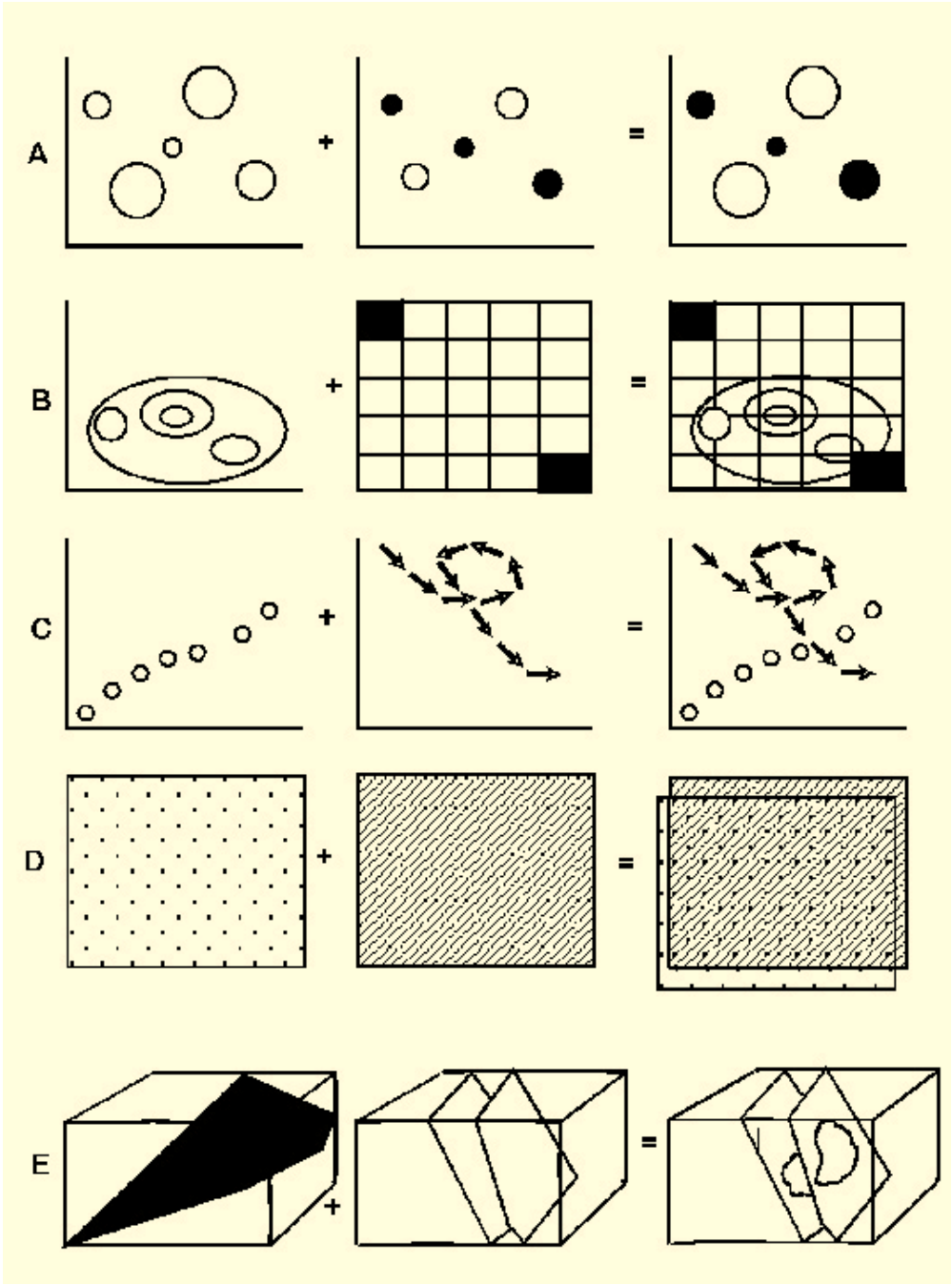


Figure 3.2: Illustration of VISTA's composition techniques. The graphs show the results of using (A) mark composition, (B) composition by super-imposition, (C) composition by union, (D) composition by transparency, and (E) composition by intersection.

3.4 Automation

Mackinlay's criteria for developing an effective visualization encoding is an important step towards designing visualization tools that are not only automated in nature, but are also domain-independent. Wehrend and Lewis [WL90] used these ideas to develop of a two-dimensional matrix, with the dimensions classified as:

- data types and objects, and
- visualization or perceptual tasks.

that can be performed on the data objects. The two together form a catalog of visualization techniques.

3.4.1 Cataloging Objects And Tasks

Data objects are data properties that represent the nature of the data being visualized. Tasks or operations represent the various visualization and perceptual tasks that the user wants to perform on the objects. Table 3.1 lists the two columns with their available values.

Wehrend and Lewis suggested that a visualization task could be broken down into smaller sub-problems, until the size of the sub-problems was at such a level that they could be categorized by entries in the object/ task matrix. The final design of the visualization is a composition of the candidate representations for the individual sub-problems.

Although this technique offers a useful formalization of objects and tasks, the catalog manner of the classifications does not allow for proper categorization of more complex structures

Table 3.1: A table of object classes and operation classes examples by Wehrend and Lewis

Object Classes	Operation Classes
scalar	identify
scalar field	locate
nominal	distinguish
direction	categorize
shape	distribution
position	rank
spatially extended region or object	compare
structure	within and between relations
	associate
	correlate

(e.g. three-dimensional fields). Wehrend and Lewis concluded that an automated visualization tool would require very specific definitions of the kinds of structures that could be used for visualization, and of the specific tasks that could be applied to each object or structure. The classification of objects and operations, nevertheless, is an effective manner of handling sub-problems within a larger visualization task.

3.4.2 Classifying Visual Knowledge

Lohse et al, [LRBW90, RL93a, RL93b] attempted to classify visual knowledge representations. Their interest in performing classification was based on the belief that data classification step is the most important step in designing a multi-dimensional visualization. Lohse et al attempted to classify data and objects, based on results from experiments on visual perception. These experiments were designed to investigate the basic properties of various graphical elements, and their ability to support visual analysis tasks.

Viewers, chosen heterogeneously, grouped graphical elements in a bottom up manner into categories of similar elements. The categorization was carried out in a trailblazing manner. Since the classification was not conceptual in nature, one potential problem was that different viewers might classify elements into completely different categories. The experimenters countered this with analyses designed to test both the classifications produced and the possibility that the subjects influenced those classifications.

The grouping procedure was repeated hierarchically, to collect the graphical elements into larger and larger clusters. This continued until all elements were in one cluster. Elements chosen for data classification were selected from a broad range of fields and included items such as icons, graphs, maps, and tables. On completion of the data sorting and classification process, the classification, the resulting categories were analyzed in two ways:

1. The first analysis task was called “subject analysis” and was done by asking the subjects which pair of already sorted items could be considered to be in the same cluster. This allowed the researchers to conclude that the sort was consistent and homogeneous.

2. The second form of analysis was called “item analysis”, in which the hierarchical nature of the clustering was collapsed to form non-hierarchical item clusters. The final classifications were expressed in a two-dimensional coordinate area, with the extremes placed on the horizontal and vertical axis. Additional groups were derived from this coordinate area. This grouping helped identify perceptually salient multi-dimensional visual representations.

The findings of this experiment provided an understanding of the way people organize and pro-

cess visual knowledge. The classification also helped researchers understand how knowledge can be conveyed visually.

The approaches of Lohse, Mackinlay and Senay and Ignatius [LRBW90], consider data classification the first stage of a visualization process. Their visualization procedures are data-driven and the approach is usually bottom up: first the data is classified, and then the classification is represented in a graphical form chosen based on its category.

3.5 Natural Scene Paradigm

Robertson and De Ferrari [RD94] however, were not convinced that the data-driven approach was always appropriate. They worried that this approach placed too much responsibility on the users. The need for user interaction means that data has to be translated for every interaction. As well, the user needs to be an expert at visualization in order to properly classify data into useful categories. Finally, Robertson and De Ferrari felt the bottom-up nature, of a knowledge base technique was inefficient in terms of keeping up with the amount of raw data available. Robertson [Rob90, Rob91] advocated a top-down approach for multi-dimensional visualization, with the emphasis placed on selecting an effective, and in our case, a perceptually salient visualization model. Such an approach requires that data attributes be represented by the individual properties of the chosen model or scene.

With this approach in mind, Robertson developed the Natural Scene Paradigm (NSP) which is based on his visualization-model driven approach. NSP is a multi-variate data visualization tool that works on the principle of inversion from Mackinlay's APT, which itself was princi-

pally data-driven and bottom-up in approach. According to Robertson and De Ferrari [RD94], user involvement in visualization systems occur in two ways: by means of user directives to the system, and by user interpretation aims. Through user directives, the user is able to explicitly define requirements to the visualization system. An instance of using directives, could involve asking the system to show a specific variable A using a specific representation B. User interpretation aims involve specifying criteria surrounding the data variables in the representation. This could, for instance, involve asking the system to show the correlation between variable B and variable C, or even criteria such as asking the system to show significant changes of the temperature variable over time. They decided to implement a user directive approach for user involvement in their system, since they believed it was easier to implement and satisfy, than user interpretation.

The Natural Scene Paradigm (NSP) implemented visualization in three stages:

1. The first stage involved the selection of a clear and understandable visualization model (such as a three-dimensional natural scene or structure).

2. The second stage involved the decomposition of the scene into its recognizable properties (these involve properties that can be controlled by external data input).

3. The third stage involved the combination of user demands with the recognizable scene properties using graphical simulation techniques.

The multi-dimensionality of the data, the use of different display methods combined with different constraints, and user directives generate varying representations of the scene. Control-

ling and modifying these factors thus controls the construction of the scene.

3.6 Flow Visualization

Gallop [Gal94] consolidated and improved upon the underlying data models in a flow visualization system. Gallop's dataset was mainly flow data. He insisted on a classification of data models and abstract visualization objects, along with an underlying data representation.

The flow datasets were mainly continuous or empirical, and time-dependent in nature. Gallop considered the empirical components and the time components as critical attributes for the decomposition of the datasets into variables and functions.

Gallop offered the opinion that data elements are basically of two kinds: dependent variables, and independent variables. His visualization methodology depended on this distinction.

Dependent variables, were ranked in a hierarchical order using dependency as a factor. Data sampling was done on these variables based on their rank.

For independent variables, no ranking was used; data sampling was done in a non-hierarchical manner. Both dependent and independent variables were then mapped to a coordinate system.

The need to rearrange data models before visualizing them was suggested by Gallop. The nature of scientific data being so diverse, Gallop concluded that a formal methodology was necessary to classify data models, and to express them.

3.7 PRAVDA

Rogowitz and Treinish [RT93] believed that in order to build visualization systems that are perceptually salient, the importance of human perceptual and cognitive mechanisms needs to be better understood and incorporated within visualization systems. The use of mathematical models for mapping visualizations could achieve representations that were multi-dimensional in nature, but interpretation of such models may not be consistent across many users. This is because although many models of visualization have mathematically similar relationships, they may be very different in their perceptual effect. Hence the need to understand the human perceptual mechanism better, and the need to incorporate this in a more effective manner within visualization systems.

Lohse et al [LRBW90] suggested ordering data models based on viewer's ad-hoc classification of elements into similar categories. Such an approach is not sufficient, however. One needs to better understand how humans perceive spatial variations of luminance and color and how variations in the physical objects or glyphs used for representation influence their interpretation. This was the basis for PRAVDA.

Properties of human vision and perception were incorporated within a set of perceptual rules. These rules were then used within PRAVDA to guide the task of mapping data attributes to suitable visual features. The perceptual rules were themselves categorized into two separate classes:

- *Class I perceptual rules*: These rules are used to build a suitable data representation. Class I rules guide the conversion of multi-dimensional data into perceptual forms, and

ensure that the structure of the data is faithfully preserved in the visualization. This is described as a "perceptual isomorphic mapping" by Rogowitz an Treinish. A "non-isomorphic mapping" occurs when the structure of the data cannot be preserved by its visual feature. This can happen, for example during the conversion from a continuous form to a discrete representation. Class I rules are therefore rules that provide isomorphic mappings from data to perceptual dimensions, or rules that ensure the invariance of higher-level visual features, such as color, size and shape.

- *Class II perceptual rules*: These rules are used to highlight features within the data being visualized. Class II rules intentionally change the data's structure to attracting attention, with methods like highlighting or segmenting the data into sub-regions to better aid exploration and analysis. These rules also allow the scaling of visual features to exaggerate details within the scenes.

PRAVDA's primary research focus has been on the perceptual effectiveness of a color model called "PRAVDAColor." PRAVDA uses a rule-based algorithm to select appropriate colors for a given data attribute. PRAVDAColor uses spatial frequency to guide the use of luminance, hue and saturation. For instance, high spatial frequency data is mapped using a monotonic luminance component. Low spatial frequency data is mapped to a single hue that varies in saturation. The rule-based operations of PRAVDAColor can also be used to select ranges for data segmentation and highlighting. Though it offers a wide range of data mapping operations, PRAVDA still requires user interaction to make a final decision on which color mapping to apply.

According to the authors, PRAVDA implements a perceptually salient rule-based system for domain-independent visualization design. PRAVDAColor, uses a rule-based approach to control the variation of hue, luminance and saturation to best match properties of a target data attribute.

To implement a more flexible multi-dimensional visualization system, variations of hue, luminance and saturation do not necessarily suffice, especially if the number of attributes that need to be supported is large. Visual interference that may occur when hue, luminance and saturation are shown together in a common display, cannot be overlooked. Healey and Enns [HE98] concluded that the maximum number of hue variations that the human visual system can rapidly distinguish is only seven. They also concluded that luminance masks hue patterns when both are shown simultaneously. PRAVDA may not produce high perceptual effectiveness in a situation that requires the display of a large number of data attributes.

Color forms an important part of any multi-dimensional perceptual visualization system. However, additional methods are needed to combine with efficient color models to more fully support the purpose of multi-dimensional dataset representation.

The Visualization Assistant (ViA) utilizes the work done by Healey and Enns [Hea96, HE98] to incorporate perceptual guidelines for use of a larger range of visual features. These guidelines, not only include color models, but also rules pertaining to the use of two-dimensional and three-dimensional texture properties. Furthermore, the rules support the identification of patterns in a candidate mapping that could produce visual interference. Although we initially focussed on direct control of individual properties related to a pexels [HE98] we have adopted

ViA to use other types of glyphs and visualization metaphors.

3.8 Perceptual Texture Elements

Healey and Enns [HE98] designed a technique using perceptual textures to visualize multidimensional datasets. Information in multi-dimensional datasets is contained in data elements. Each data element encodes one or more data attributes. These data elements are arrayed across an underlying three-dimensional height field. One or more pexels (perceptual texture elements) are used to display each data element, with the attribute values of the element controlling the visual appearance of its pexels. Texture patterns formed by groups of neighboring pexels can be visually identified by a viewer during exploration and analysis. Since the pexels are designed to be perceptually salient, many tasks like searching for a target or target area, boundary detection between spatial groups with common attribute values, region tracking, and estimation can be performed preattentively.

Healey and Enns [HE98] also tried to focus their research on answering the following questions:

1. Which perceptual dimensions should be used to control the appearance of a texture pattern?

2. How can dataset attributes best be used to control the dimensions of the pexels?

3. How much visual interference occurs between the perceptual dimensions when they are varied in a common display?

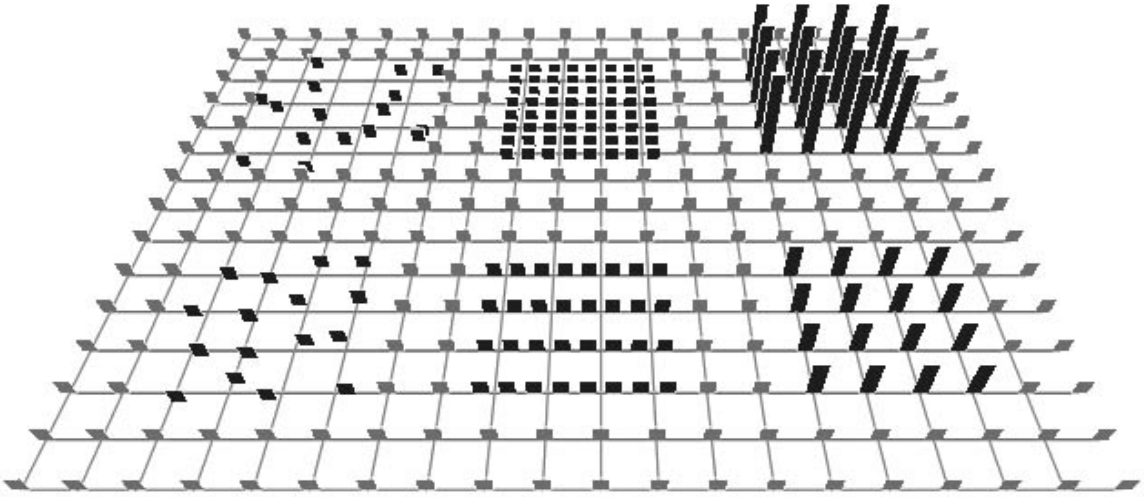


Figure 3.3: An illustration of the different kinds of pexel strips that could be used for multi-dimensional visualization. Starting from left, and moving right we have pexel strips showing two levels of irregularity, two levels of density, and two levels of pexel height.

Healey and Enns used thin paper strip pexels to support the visual features color, height, density of pexels per unit area, and regularity of placement. Figure 3.3 shows examples of using pexels in the form of thin paper-like strips. The examples illustrate the use of pexels of different irregularities, densities and heights.

In order to establish the effectiveness of pexel based visualizations, Healey and Enns conducted psychophysical experiments designed to test a user's ability to rapidly and accurately identify the presence or absence of target pexels with a unique height, density and regularity. The experiments also tested viewer performance with different target pexel types, exposure durations, target group sizes, and background texture dimensions. Results from these experiments formed a set of guidelines that measure the effectiveness of a given configuration of texture elements for multi-dimensional dataset visualization. As we shall see in the next chapter, these

guidelines act as the basis for ViA's evaluation of candidate data-feature mappings.

3.9 Search Techniques

Search techniques are useful in visualization systems for exploring search spaces containing possible solutions to the problem of selecting effective data-feature mappings.

A "situation space" [RN95] is an exploratory space of possible conditions or states. In the case of visualization systems, where finding an optimal, perceptually salient mapping is important, these states are possible mappings or solutions for our visualization problem. For simple visualization tools that represent only a single attribute, each state would contain a one-to-one mapping between the attribute and a visual feature. Given a single attribute and a set of visual features $V = V_1, \ldots, V_n$ with n visual features that could be mapped to the attribute, our situation space would contains n states. For multi-dimensional dataset visualization, where we consider a set of m data attributes and n visual features, with $n \geq m$, we have a possible situation space containing $n!/(n-m)!m!$ possible mappings, where each node in the space is a complete mapping with every data attribute A_i mapped to a unique visual feature from V_j. As m and n grow in size, the number of states in the situation space increases dramatically.

Traditional problem-solving techniques often involve examination of every node in the space prior to selecting those that fit the criterion for the given task. Although this method is comprehensive, it is inefficient for large datasets. In addition to a very large situation space to explore, there is the problem of pursuing solution paths that do not lead to high-quality results.

To construct and explore large search spaces efficiently when only a limited amount of time and effort are available, a system needs a more flexible search strategy. Obtaining a solution that may be very good, but not necessarily the best possible solution for the task, could be more desirable when selecting a visualization for a large dataset.

3.9.1 Planning

Russell and Norvig [RN95] advocate that the use of artificial intelligence (AI) planning techniques to find optimal solutions efficiently, for tasks involving a wide range of possible states or actions, as opposed to accurate but inflexible comprehensive problem-solving techniques. The planning process, when assisted by a rule-base containing guidelines for search, can explore a situation space and provide near-optimal solutions in short periods of time, even when the search space is large and the time required to search it completely would be great. While heuristics devised by planning systems may not always provide the best solutions for a given problem, they can still produce good workable results for a given task. This may be more desirable than comprehensive techniques which may arrive at the best solution, but at a prohibitive cost in time and computing effort.

A planning system could be further enhanced by employing a knowledge base of rules that can act as guidelines to direct the search along promising paths. A limited rule-base would be significantly easier to manage than problem-solvers utilizing a comprehensive knowledge base that contains all possible actions for every possible situation. Moreover, a rule base for a planning system need not be considered complete, since new rules can be added as they are

found.

3.9.2 Mixed-Initiative Search

Mixed-initiative search is an efficient search strategy that is inspired from the AI planning approach. Such a strategy is very effective for exploration and analysis of very large search spaces.

The mixed-initiative search technique resulted from advances in AI planning techniques as well as studies in human-computer interaction and usability. It approaches generation of a solution to a task or problem (in our case, that of generating a suitable mapping) as a planning task, involving both the system and the human user. The system is partly guided by user commands and queries. According to St. Amant and Cohen [AC97], a system can be considered to be an assistant as opposed to being merely a tool kit, if it adopts a mixed-initiative approach. St. Amant and Cohen proposed a mixed-initiative strategy for performing Exploratory Data Analysis (EDA) tasks for statistical analysis and related areas. Two conditions separate assistants from being tool kits:

- *Partial autonomy of assistants*: One needs to give general and brief instructions, and let the assistant decide how to carry them out, as opposed to giving detailed, step-by-step instructions.

- *Response to guidance or "Accommodation"*: Assistants can be guided while they work as opposed to waiting for a task to be completed before modifying the search parameters. Responsiveness to human guidance, termed as "accommodation" [LP88] is an important

characteristic of assistants.

A mixed-initiative search assistant could be supported by a perceptual rule-base containing guidelines to evaluate candidate data-feature mappings for perceptual salience. For visualization systems, a mixed-initiative approach may make the task of searching for effective mappings more efficient.

In this chapter we examined various related efforts in scientific visualization and perceptual visualization. We also discussed rule-based methods for automated or semi-automated visualization design. We described the pexel method for representing multi-dimensional datasets in a perceptually salient manner. We also examined mixed-initiative search strategies as one potential technique for effectively searching for perceptually salient data-feature mappings.

The following chapter discusses the design and architecture of ViA, and in particular, examines the concept of generating and utilizing "hints" against the backdrop of the search process to locate appropriate visualizations for a given dataset and user-chosen analysis requirements.

Chapter 4

ViA

In the previous chapters, we discussed the need for perceptual visualization and some of the various studies and efforts within scientific visualization. We introduced ViA (Visualization Assistant) as a semi-automated perception driven assistant that aids in designing effective visualizations of multi-dimensional datasets. In this chapter, we shall discuss in some detail the architecture and processes that guide the working of ViA.

4.1 General Architecture

A part of our discussion on related works focused on the approach of developing a visualization tool that aids in generating effective perceptual visualizations. Some of the methods adopted were clearly data driven, wherein the classification of the data attributes to be visualized was the most important aspect of the visualization system. Other methods focused on classifying visual representations based on their ability to display different types of datasets. These methods were

concerned more with the visual representations as compared to the nature of the datasets. An observation that one could perhaps draw from both these approaches to visualization, is that one would need a balanced approach towards incorporating not just the nature of the data being visualized but also the visual methods used to represent the data.

Raw data is input to a visualization system, through an input mechanism. The input mechanism could also be responsible for the classification of the attributes. For data classification purposes, information about the nature of the dataset would be very important for the visualization system.

A processing mechanism is responsible for performing a visual mapping for the data attributes, by assigning appropriate visual features to represent the attributes.

A display mechanism displays the representation generated by the processing mechanism, and aids the user in search and exploration tasks. The display mechanism supports various view transformations to be performed on the visualization.

Additionally, the visualization system would need to support an effective mechanism for human interaction. Interaction could involve many different forms, from providing suitable support for input of raw data to the system and describing the data properties, to performing view manipulation, exploratory and analysis tasks, or direct manipulation of the visual objects within the representation, and suggestions to produce new mappings or visualizations.

Our design of the visualization system ViA focuses on the mapping of data attributes of multi-dimensional datasets to visual features. The following sections will discuss in detail the mapping procedure.

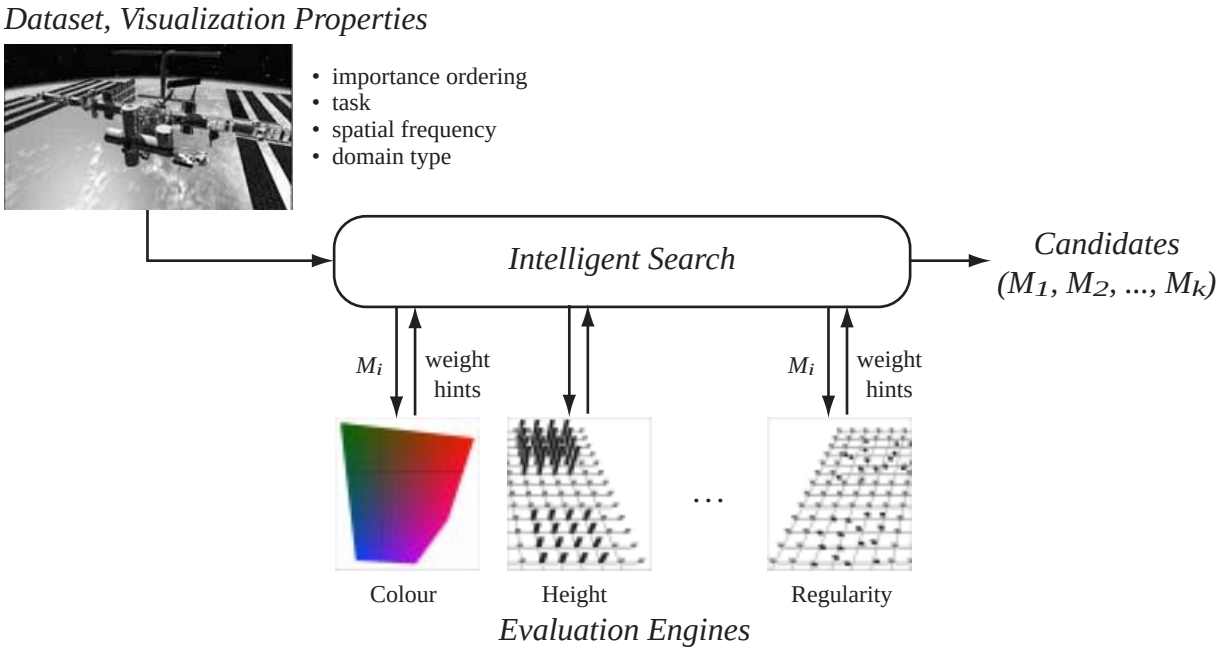


Figure 4.1: An illustration of the architecture of ViA

4.2 ViA Architecture

Figure 4.1 shows the architecture of ViA. ViA consists chiefly of two main components called the search engine and the evaluation engine. The search engine is responsible for generating mappings between data attributes and visual features. The evaluation engines are responsible for evaluating the mappings produced by the search engine based on the perceptual guidelines that determine whether the mapping generated is perceptually salient or not. The search engine also responds to various suggestions or hints on how to improve the current mapping. The hints are used by the evaluation engines to offer a collection of recommendations to help to fix weaknesses in the mapping. Because hints can conflict with one another, the search engine is responsible for deciding how to apply the hints in an appropriate and effective manner.

4.3 Evaluation Engines

An illustration of ViA's architecture is presented in the Figure 4.1. The individual evaluation engines color, luminance, height, density and regularity represent the evaluation process. Each visual feature utilized by ViA for multi-dimensional visualization is supported by a corresponding evaluation engine. The evaluation engines critique data-feature mappings produced by the search engine.

4.3.1 Data-Feature Mappings

A data-feature mapping M is maintained as a data structure with two parts:

1. The data attribute-visual feature pairs that make up the mapping,

2. The relevant properties of the dataset that are needed to evaluate M.

When the dataset is first given to ViA, several dataset properties are established, either by examining the dataset itself, or by asking the viewer. These include:

1. *Domain type*: Is the attribute's domain type discrete or continuous?

2. *Importance ordering*: What is the relative importance ordering of the data attributes to the user?

3. *Spatial frequency*: Is the spatial frequency of each attribute high or low?

4. *Tasks*: What specific analysis tasks (search, estimate, boundary detect, tracking or none) need to be performed on each attribute?

When M is evaluated, each individual engine tackles the data-feature pair in M containing its corresponding visual feature. For example, if M utilizes color and luminance, it will contain a pair with an attribute mapped to color, and another pair with an attribute mapped to luminance. The color evaluation engine will evaluate the color pair and the luminance engine will evaluate the luminance pair.

The evaluation engines thus analyze each mapped pair within M and note any violations with respect to the perceptual guidelines that M is supposed to satisfy. Based on the number and the severity of these violations, the evaluation engines generate a normalized evaluation weight. Wherever possible, along with the evaluation weight, suggestions to improve the mapping in the form of "hints" are also generated by the individual evaluation engines. Hints include a suggested action to improve M and an estimated improvement weight if the hint were applied and implemented on M.

Hints are critical to allow the search engine to intelligently guide the search process for better mappings. Design and interpretations of the hints and the manner in which they guide the search process forms one of the primary focuses of this thesis. We will examine these topics in this and later chapters.

4.3.2 Evaluation Process

Let us consider a typical mapping M that is to be evaluated. For each data attribute A_j, M contains information about the attribute in terms of the visual feature V_j associated with A_j, domain type of A_j (whether discrete or continuous), its spatial frequency (whether high or

low), its importance (as a normalized value that orders A_j relative to other attributes), and the user tasks (if any) to be performed on A_j.

The evaluation engines collectively contain the perceptual guidelines that allow analysis of the mappings produced. Each mapping M that is to be evaluated contains mapped pairs of data attributes and visual features, $A_j \rightarrow V_j$. Each pair, as mentioned before, also contains information about the attribute with respect to its properties and tasks. The attribute-feature mapped pair, $A_j \rightarrow V_j$, is evaluated by each of the evaluation engines (color, luminance, height, density, orientation and regularity) on the basis of:

1. *Attribute domain type*: Whether V_j supports the domain type of A_j. For instance, if A_j is continuous in nature, can V_j produce continuous scales? Or if A_j is discrete, can V_j produce an appropriate number of distinguishable values to represent the range of discrete values of A_j?

2. *Interference*: Whether another mapped pair within M interferes visually with $A_j \rightarrow V_j$. For the purposes of the evaluation process, the perceptual salience order of the visual features is: luminance, color (hue), height, density and regularity. Interference can occur if a less important attribute A_k is utilizing a more perceptually salient attribute V_k.

3. *Spatial frequency*: Whether the spatial frequency of A_j is appropriately represented by V_j.

4. *Task applicability*: Whether user tasks on A_j can be supported by V_j.

The complete mapping M is thus evaluated by the evaluation engines, with each evaluation

engine testing the pair $A_j \rightarrow V_j$ for its perceptual salience. Each evaluation engine returns an evaluation weight in the range 0.0 to 1.0 for M, along with zero or more hints. An evaluation weight of zero means that the mapped pair $A_j \rightarrow V_j$ is a flawed mapping, and one means that the pair is a good mapping with no weaknesses. A lower evaluation weight for a mapping results from violations of the perceptual guidelines in the evaluation engine that returned the evaluation weight. A low evaluation weight, is often accompanied with hints that provide suggestions for potential improvements. The hints also estimate the improvement weight for the mapping if the hint were applied to the current mapping.

The perception driven search procedure we designed that utilizes hints to generate perceptually salient mappings is explained in the following chapter.

Chapter 5

Search Engine

The previous chapter focused on the evaluation engines. We discussed briefly the constitution of the individual evaluation engines. In this chapter we focus on the search engine. In particular we study the design of hints to be used by the evaluation engines to guide the search for perceptually salient mappings. This forms one of the main contributions of this thesis.

The search engine, as mentioned earlier, is responsible for producing mappings that are evaluated for perceptual salience of the mappings. The function of the search process is to generate high-quality mappings in an efficient fashion. As mentioned earlier, this involves having to search through a situation space [RN95] of all possible mappings and follow one or more paths that lead to the most perceptually balanced or optimal mappings. User input involves providing ViA with details regarding the data attributes, and the input files containing the datasets themselves which need to be visualized. A sample mapping that is assumed to be fairly perceptually salient is used as the starting point for the search process.

We also discussed some of the aspects of the evaluation process. The evaluation procedure returns an evaluation weight adjudging the perceptual salience of the mapping. Good mappings return high evaluation weights and poor mappings return low evaluation weights, thus allowing us to compare between mappings using evaluation weights as a suitable metric.

5.1 Hints

Search in ViA is perception driven in nature. Perceptual guidelines incorporated within the evaluation engines, are used to evaluate the perceptual salience of any mapping produced by the search engine. Although evaluation weights are extremely useful in determining the perceptual salience of a mapping, we need additional information to aid us in producing better mappings. The perceptual guidelines guide the evaluation procedure, and violations in the current mapping with these guidelines result in reduction of the evaluation weight for the current mapping. Reporting these violations give us clues that would aid us in potentially fixing the current mapping. Structuring these clues in a way that suggests fixing or improving the current mapping to potentially lead to higher evaluation weights, is done with hints.

When the search engine applies the suggestion within the hint to improve the mapping, it creates a new mapping that has the potential to be better than the original mapping that caused the hint. The evaluation engine evaluates the new mapping for perceptual salience and returns an evaluation weight and hints to potentially improve the new mapping.

We can thus divide the search process into three broad stages:

1. *Evaluating the mapping*: Starting with the first mapping, the evaluation engines analyze

each mapping sent by the search engine.

2. *Generating hints and hint-chains*: The evaluation engines provide feedback in form of evaluation weights and hints about how to improve the mapping. The hints generated are combined into chains of hints that can be applied simultaneously to improve the mapping. The hint chains and are stored in a priority queue ordered by their estimated improvement weight.

3. *Processing hint chains to generate new mappings*: Each hint chain suggests certain changes to the original mapping that could improve the mapping. The most promising hint chains are chosen and processed to generate new mappings, which are then sent for evaluation to repeat steps 1, 2 and 3.

The first stage involving the evaluation process is carried out by the evaluation engines. We discussed the nature of the evaluation engines in the previous section. The second stage of the search process involves the analysis of hints and creation of hint-chains from these hints to guide the search for new mappings. These operations are explained in the following sections.

5.1.1 Hint Structure

A hint comprises chiefly of:

1. The hint class.

2. The hint type or category.

3. The visual features or importance weights that the hint is recommending to modify. These could also be considered the hint operands.

4. The estimated improvement in evaluation weight if the hint were to be applied.

The improvement weight provided with every hint, is the estimated improvement in the current evaluation weight for the particular mapped pair, should the hint be applied.

Hints are produced on the basis of the four tests: attribute domain type, visual interference, spatial frequency and task applicability. These are referred to as hint classes. Based on these four tests, hints of four different types or categories are generated:

- *discretization*: suggests that a continuous attribute be reduced to appropriate discrete intervals, or the number of discrete intervals of an attribute be reduced to a lower value,

- *feature swap*: suggests that the visual features currently assigned to an attribute be swapped with a more appropriate feature; this can be caused by a mismatch between the feature and the attribute's task, spatial frequency, or domain type; it can also occur when visual features interfere with one another,

- *importance weight modification*: suggests that the importance weighting of an attribute be adjusted suitably again to avoid the visual interference between two attributes, and

- *task removal*: suggests that an unimportant task be disallowed or removed from on an attribute's task list.

Table 5.1: A table of hint classes (domain type, visual interference, spatial frequency, and task applicability) and visual features (luminance, color, height, density, regularity, and orientation). Each cell shows what the visual feature supports for the given hint class: "ok" for a good match, "poor" for a weak match, and "no" for an unallowed match. The final row shows the hint types that might be suggested to fix weaknesses in the given hint class

	Domain Type	**Visual Interference**	**Spatial Frequency**	**Task Applicability**
Luminance	*continuous:* ok *discrete:* up to 3		*hi:* ok *lo:* poor	*search:* discrete *estimate:* discrete *boundary:* ok *track:* ok
Color	*continuous:* ok *discrete:* up to 7	w/ luminance	*hi:* poor *lo:* ok	*search:* discrete *estimate:* discrete *boundary:* ok *track:* ok
Height	*continuous:* ok *discrete:* up to 5	w/ luminance w/ color	*hi:* ok *lo:* ok	*search:* discrete *estimate:* discrete *boundary:* ok *track:* ok
Density	*continuous:* no *discrete:* up to 4	w/ luminance w/ color w/ height	*hi:* poor *lo:* ok	*search:* discrete *estimate:* discrete *boundary:* ok *track:* ok
Regularity	*continuous:* ok *discrete:* up to 2	w/ luminance w/ color w/ height w/ density	*hi:* poor *lo:* ok	*search:* no *estimate:* no *boundary:* ok *track:* discrete
Orientation	*continuous:* ok *discrete:* up to 12	w/ luminance w/ color	*hi:* ok *lo:* ok	*search:* discrete *estimate:* discrete *boundary:* ok *track:* ok
Hint Types	Feature Swap	Feature Swap Weight Modify	Feature Swap	Task Remove

5.1.2 Perceptual Guidelines

The perception driven nature of the search process arises from the fact that the search is influenced by the perceptual guidelines that produce evaluation metrics and hints. Table 5.1.1 shows the different hint classes and their properties. As we mentioned before, there are four different types of hints produced by the evaluation process. Each evaluation engine can produce any of the four types. These types include:

5.1.2.1 Discretize

Each visual feature can support discrete valued attributes, but only up to a certain perceptually salient limit. These limits vary for the different visual features. For example, according to our perceptual guidelines, color as a visual feature can be used to display up to seven easily distinguishable colors or hues (red, blue, green, etc.). More than seven different colors hampers the ability of the visual system to rapidly identify different attribute values, within the visualization. In other words, the representation stops being perceptually salient, if color is used to represent more than seven discrete values of an attribute. Likewise, height used as a visual feature can support to five discrete values, density can support four discrete values, luminance can support up to three values and regularity can support up to two discrete values.

Note that data can still be displayed. However, it may not be an optimal representation. In all these cases, if the attribute bound to the visual feature has more discrete values assigned than the visual feature can optimally support, then the corresponding evaluation engine suggests that the attribute be discretized to an appropriate maximum size.

5.1.2.2 Feature Swap

This hint suggests that visual features assigned to certain attributes be swapped or interchanged in order to make the mapping more perceptually salient.

The conditions that lead to a feature-swap being suggested by the evaluation engines are numerous. For instance, if an attribute A_j is assigned to a visual feature V_j that supports fewer than the number of unique values found in A_j, the evaluation engines could suggest switching to a visual features V_k that can support A_j in its current form. If some other attribute A_k is currently using V_k, then A_j and A_k must swap features. This hint would be suggested in addition to a discretize hint (as described previously). As an example, suppose $V_j = luminance$ was used to represent an attribute A_j with seven unique values. Luminance is not optimal since it supports a maximum of three values. A feature-swap hint, generated by the luminance evaluation engine would recommend that luminance be swapped with color, since color can support up to seven unique discrete values. This kind of feature swapping will also be recommended if V_j does not support A_j's spatial frequency (e.g. if A_j was low frequency when V_j was luminance), or if V_j does not support the tasks the user wants to perform on A_j.

A feature can also be suggested to avoid visual interference. If a less important attribute A_k is assigned to a perceptually salient visual feature V_k, visual interference occurs between A_j and A_k. In such a scenario, the evaluation engine detects this condition and suggests a feature swap between A_j and A_k.

5.1.2.3 Importance Weight Modify

This hint suggests that the importance weight for an attribute be adjusted up or down by a certain amount. This happens in visual interference situations where a less important attribute A_k is assigned a more salient visual feature V_j. If the difference in importance weights between A_j and A_k is small, then weights can be modified slightly to reverse their relative ordering(i.e. to make A_k more important than A_j). This resolves the interference conflict. In this way, an importance weight modification can be used as an alternative to feature swapping.

5.1.2.4 Task Adjust / Task Remove

Analysis and exploration of the information represented by the visualization is done by the user in the form of various tasks. Tasks supported by ViA are of four different kinds:

1. Search.
2. Estimate.
3. Boundary detect.
4. Tracking.

The ability of an attribute A_j to support different tasks depends on its visual feature V_j and on A_j's domain type: discrete or continuous. For example, color, luminance, height and density when mapped to attributes of discrete types can support any tasks, i.e. all tasks are considered valid within the representation. The same four visual features when mapped to continuous

attributes can only support boundary detection and tracking. Regularity can support boundary detect when mapped to an attribute of discrete type, and tracking for both discrete and continuous attributes.

Whenever an inappropriate task is assigned to an attribute, the evaluation engines issue a warning that the task is not supported for the given attribute type and visual feature. If the importance weight of the attribute is below a lower threshold, the evaluation engine suggests that the offending task be removed for the given attribute, in the form of a task-adjust or task-remove hint.

All the hints generated by the tests within each evaluation engine are stored in a hint array. The hint array is refreshed with new hints after each complete complete evaluation cycle.

5.2 Hint Chains

Hints are produced independently by the evaluation engines. While acting on a particular pair in the mapping, each hint test will produce hints independent of those generated by the other engines. Although hints can be applied individually to build new mappings, working with one hint at a time may restrict the efficiency of the search process. This can confine our search for better mappings to a local maxima within the search space, which does not allow us the option of pursuing other paths along the search space that would lead to potentially better mappings. We need a technique that addresses this problem of restricted search by using hints to probe other areas of the search space.

Our proposed technique is to combine hints together in chains that allow us to apply multi-

ple hints simultaneously to produce new mappings. Such a technique may reduce or eliminate the problem of restricting search to a local maxima within our search space since a hint chain repositions the search multiple steps from its current location in the search space. This allows us to investigate new mappings or paths that may be more appropriate for the user, and that may eventually lead to more salient mappings. It also allows us to quickly move to promising areas within the search space.

5.2.1 Hint Chain Generation

Hint chains are formed as a combination of multiple hints returned by the evaluation engines. Hint chains are produced only after a complete evaluation of the current mapping, involving all evaluation engines, has taken place. Combinations of hints are created by a recursive procedure that generates all possible sequences of hints. This procedure builds subsets of the hints into chains whose lengths vary from a minimum of one (representing an individual hint) to a maximum equal to the total number of hints returned by all the evaluation engines.

Hint chains are generated to make search more efficient and flexible. However, since evaluation engines produce hints in an independent manner, a complete evaluation of a mapping often produces hints that involve the same visual feature or the same data attribute. Such hints may be produced by different evaluation engines, or even by different tests within the same engine. Generating all sequences of these hints will produce hint chains that modify a common attribute or visual features in different ways. This leads to potential conflicts when they are applied, since we have no way of knowing which hints should take precedence.

5.2.2 Conflicting Hints

One instance where hints may conflict with each other concerns those hints acting on the same visual feature. For example, consider a situation where the color engine produces a hint for feature swap suggesting that color be swapped with luminance (this could be occur if the mapping contained luminance is assigned to an attribute with a lower importance weighting than color). The height evaluation engine also generates a hint for feature swap suggesting that height be swapped with luminance (for the same reason that luminance has been assigned to an attribute with lower importance weighting than height). Such a situation presents a conflict in the sense that both hints cannot be applied simultaneously, because they both deal with luminance being swapped with different visual features.

In another instance, conflicts could result when a chain contains more than one hint dealing with the same attribute. For example, consider a situation where more than one hint generated by the evaluation engines requires that the importance weight for a particular attribute be modified to two different values.

Both these situations are prevented during the chaining process by using pruning conditions that check for recurring visual features or data attributes as hints are added to a current chain. If we prune a hint that conflicts with other hints that are already part of the chain, it is not added to the chain.

The use of these conditions also helps to keep the total number of hint chains to a manageable size. Because all possible orderings of hints are generated, conflicting hints will both have a chance to be applied but only as a part of two different hint chains.

Duplicate hint chains with exactly the same hints, but in a different order will be managed by the search algorithm itself. The search engine keeps track of each mapping it evaluates. Any request to process a mapping that was previously seen is ignored. Therefore, two hint chains with the same hints generated from a common mapping M will produce an identical result M'. the first occurrence of M' will be evaluated by the search engine. The second occurrence will be ignored.

The sequence generator that combines hints to form chains, also produces hint chains comprised of a single hint. This allows ViA to consider both individual hints and hint chains to try to produce improved mappings.

Hints of various sizes compete with each other. Based on the evaluation weights that result from the application of the hints and the evaluation of the new mappings, the success of hint chains of varying lengths can be determined.

5.2.3 Processing Hint Chains

As hint chains are generated they are added to a priority queue structure. Hint-chains are stored in priority queue ordered by their estimated improvement weights. Removing a hint chain from the front of the priority queue returns the hint chain with the highest estimated improvement weight.

Each hint within the hint-chain is applied to the mapping that produced it. Each hint chain contains a description of how to recreate this mapping M. Since hint chains are structured such that the hints do not conflict with each other, all the hints in the chain can be applied

simultaneously. Processing a hint chain hence produces a new mapping M_k. As previously noted, M_k is identical to a mapping previously generated, that is if M_k has already been seen, it is ignored (since if it has already been generated, then it has already been evaluated too.)

M_k is sent to the evaluation engines to be evaluated for perceptual salience. Any violations of the perceptual guidelines results in new hints to improve M_k. These new hints are combined once again into hint chains, which are placed on the priority queue to be applied at some point M_k. In this way the algorithm searches along the most promising paths for improved data-feature mappings.

5.3 Search Termination

With respect to our three stages of search, evaluation of a sample mapping forms the first stage of the search process. The evaluation produces an evaluation weight and possible hints wherever necessary, to improve the perceptual salience of the mapping. The generation of hints, and the creation of hint-chains denotes the second stage of the search process.

The third stage involves the processing or application of hint chains to the mapping as they are removed from the priority queue, the generation of new mappings from applying hint chains, and the submission of the new mappings to the evaluation engines to repeat the evaluation process.

As mentioned earlier, a list containing the twenty-five best mappings to date (based on the evaluation weights returned for those mappings) is maintained. This list is constantly updated as new mappings are produced and evaluated.

This process of generating hints and hint-chains, and producing new mappings from application of the most promising hint chains continues until no new mappings are generated. At this point, the priority queue is empty. When all hint-chains have been exhausted, ViA presents the list of the twenty-five best mappings to the user, ordered by evaluation weight. This denotes the end of a ViA run.

In the next chapter we shall look at the application of ViA to practical environments in order to demonstrate its effectiveness in generating perceptually salient visualizations over multiple domains.

Chapter 6

Practical Applications

In the previous chapters, we described the two main components of ViA's architecture: the evaluation engines and the search engine. The evaluation engines evaluate data-feature mappings and offer hints on how to improve their perceptual salience. Hints are generated from guidelines that formalize the abilities of low-level human vision. The search engine uses the hints to search paths that are most likely to produce better mappings. We discussed in detail the design of the search engine and the way in which hints are are applied to build new mappings. Finally, we explained how hints are chained together to search as efficiently as possible for the best mappings.

ViA was designed to work in a domain-independent fashion. The perceptual guidelines were not constructed for a particular problem environment, rather they were built based on the fundamental strengths and weaknesses of the human visual system. ViA allows for domain-specific constraints in the form of user-chosen importance weightings for the different at-

tributes, a listing of the analysis tasks to be performed, and dataset-specific properties like domain-type and spatial frequency.

To test ViA's flexibility, we used it to suggest visualizations for two real-world datasets. First, we applied ViA to a collection of environmental and weather readings. Next, we applied it to an e-commerce auction agent dataset. The use of two different domain areas emphasizes ViA's domain independent nature. The datasets were visualized using two different rendering techniques. The weather datasets were displayed with a nonphotorealistic visualization model. This model uses a two-dimensional visualization design that simulates Impressionist painting styles to represent data with colored, textured brush strokes. The e-commerce auction agent datasets were visualized with the pexel method [HE98] discussed in earlier chapters.

6.1 Weather Dataset Visualization

Our first application area is one that we have used throughout this thesis: the visualization of weather datasets. These datasets represent average monthly weather conditions in $\frac{1}{2}^{\circ}$ latitude and longitude steps across the continental United States. The averages were provided by the Intergovernmental Panel on Climate Change (IPCC) using historical weather conditions for the years 1961 to 1990. The datasets contain eleven separate data attributes: *temperature, wind speed, pressure, precipitation, cloud cover, diurnal temperature range, ground-frost frequency, maximum temperature, minimum temperature, radiation, vapor pressure,* and em wet day frequency.

We used ViA to build visualizations for four different attributes: *temperature, wind speed,*

Table 6.1: Attributes from the weather dataset visualized with ViA; each attribute's domain type, spatial frequency, analysis tasks to support, and relative importance as specified to ViA are listed in the table; ViA's initial mapping M_1 was *temperature* $\rightarrow$ color, *wind speed* $\rightarrow$ height, *pressure* $\rightarrow$ density, and *precipitation* $\rightarrow$ orientation

Attribute	Domain	Frequency	Task	Importance
temperature	discrete (7 unique values)	high	search	1.0
wind speed	discrete (23 unique values)	low	boundary	0.75
pressure	discrete (296 unique values)	low	boundary	0.15
precipitation	discrete (82 unique)	high	search	0.75

pressure, and *precipitation.* ViA generated mappings for these data attributes using four visual features: color, height, density, and orientation.

6.1.1 Weather Dataset Mappings

Table 6.1 describes the weather dataset properties and initial user constraints. ViA chose a starting mapping M_1 of *temperature* $\rightarrow$ color, *wind speed* $\rightarrow$ height[1], *pressure* $\rightarrow$ density, *precipitation* $\rightarrow$ orientation. The evaluation engines evaluated M_1 based on the user constraints and dataset properties in the following manner:

1. *temperature* $\rightarrow$ color: Color is not appropriate for high spatial frequency data; penalty = -0.0625. ViA suggests two feature-swap hints: swap color with height (currently mapped to *wind speed)*, and swap color with orientation (currently mapped to *precipitation)*.

2. *wind speed* $\rightarrow$ height: The number of unique *wind speed* is 23, but height only supports a maximum of five discrete values; penalty = -0.0575. ViA suggests a discretize hint: discretize *wind speed* from 23 values to five values.

[1] In a two-dimensional visualization, height corresponds to the *size* of a 2D glyph

3. *pressure* $\rightarrow$ density: The number of unique *pressure* is 296, but density only supports a maximum of four discrete values; penalty = -0.0625. ViA suggests a discretize hint: discretize *pressure* from 296 values to four values.

4. *precipitation* $\rightarrow$ orientation: The number of unique *precipitation* is 82, but orientation only supports a maximum of twelve discrete values; penalty = -0.0625. ViA suggests a discretize hint: discretize *precipitation* from 826 values to twelve values.

Based on these results, M_1 returned a total normalized evaluation weight of 0.755:

- *temperature* (w/7 discrete values, importance = 1.0) $\rightarrow$ color, weight = 0.1875,
- *wind speed* (w/23 discrete values, importance = 0.75) $\rightarrow$ height, weight = 0.1925,
- *pressure* (w/296 discrete values, importance = 0.15) $\rightarrow$ density, weight = 0.1875, and
- *precipitation* (w/82 discrete values, importance = 0.75) $\rightarrow$ orientation, weight = 0.1875.

M_1 produced five hints, two generated by the color engine, and one each by the height, density and orientation engines:

1. $H_{1,1}$: Feature swap *temperature* (color) with *wind speed* (height), estimated improvement = 0.0625.

2. $H_{1,2}$: Feature swap *temperature* (color) with *precipitation* (orientation), estimated improvement = 0.0625.

3. $H_{1,3}$: Discretize *wind speed* (height) from 23 to five, estimated improvement = 0.0575.

4. $H_{1,4}$: Discretize *pressure* (density) from 296 to four, estimated improvement = 0.0625.

5. $H_{1,5}$: Discretize *precipitation* (orientation) from 82 to twelve, estimated improvement = 0.0625.

These hints were combined to form fifteen hint chains of varying lengths. The total number of hint chains was constrained by the fact that the search engine does not allow conflicting hints on the same chain. All the hints are applied, but conflicting hints must appear on different hint chains. The hint chains that ViA generated were:

1. $\{ H_{1,1} \}$: feature swap color w/height; estimated improvement = 0.0875.

2. $\{ H_{1,1}, H_{1,4} \}$: feature swap color w/height, discretize *pressure* from 296 to 4; estimated improvement = 0.1125.

3. $\{ H_{1,1}, H_{1,4}, H_{1,5} \}$: feature swap color w/height, discretize *pressure* from 296 to 4, discretize *precipitation* from 82 to 12; estimated improvement = 0.1375.

4. $\{ H_{1,1}, H_{1,5} \}$: feature swap color w/height, discretize *precipitation* from 82 to 12; estimated improvement = 0.1125.

5. $\{ H_{1,2} \}$: feature swap color w/orientation; estimated improvement = 0.0875.

6. $\{ H_{1,2}, H_{1,3} \}$: feature swap color w/orientation, discretize *wind speed* from 23 to 5; estimated improvement = 0.1125.

7. $\{ H_{1,2}, H_{1,3}, H_{1,4} \}$: feature swap color w/orientation, discretize *wind speed* from 23 to 5, discretize *pressure* from 296 to 4; estimated improvement = 0.1375.

8. $\{ H_{1,2}, H_{1,4} \}$: feature swap color w/orientation, discretize *pressure* from 296 to 4; estimated improvement = 0.1125.

9. $\{ H_{1,3} \}$: discretize *wind speed* from 23 to 5; estimated improvement = 0.0825.

10. $\{ H_{1,3}, H_{1,4} \}$: discretize *wind speed* from 23 to 5, discretize *pressure* from 296 to 4; estimated improvement = 0.1125.

11. $\{ H_{1,3}, H_{1,4}, H_{1,5} \}$: discretize *wind speed* from 23 to 5, discretize *pressure* from 296 to 4, discretize *precipitation* from 82 to 12; estimated improvement = 0.1375.

12. $\{ H_{1,3}, H_{1,5} \}$: discretize *wind speed* from 23 to 5, discretize *precipitation* from 82 to 12; estimated improvement = 0.1125.

13. $\{ H_{1,4} \}$: discretize *pressure* from 296 to 4; estimated improvement = 0.0875.

14. $\{ H_{1,4}, H_{1,5} \}$: discretize *pressure* from 296 to 4, discretize *precipitation* from 82 to 12; estimated improvement = 0.1125.

15. $\{ H_{1,5} \}$: discretize *precipitation* from 82 to 12; estimated improvement = 0.0875.

Given a hint chain with k hints $\{ H_1, \ldots, H_k \}$, where each hint H_i has an estimated improvement weight w_i, the chain's total estimated improvement W is computed as:

$$W = w_{max} + c\,k \tag{6.1}$$

where $w_{max} \geq w_i \ \forall i = 1, \ldots, k$ is the largest estimated improvement, and c is a fractional constant used to reward longer hint chains.

The hint chains are stored on a priority queue ordered by each chain's estimated improvement W. Every chain encodes a description of the mapping on which it is based (i.e. M_1 in the current example). This is needed to reconstruct the mapping when the hints in the chain are applied.

Once the hint chains are placed on the queue, the chain at the front of the queue (i.e. the chain with the largest W) is removed and its hints are applied to build new candidate mappings. In our example, three hint chains had $W = 0.1375$: chain 3, chain 7, and chain 11 (the exact order of these three chains on the queue depends on the order in which they were generated). In our implementation, chain 3 was at the front of the queue:

- $\{ H_{1,1}, H_{1,4}, H_{1,5} \}$: feature swap color w/height, discretize *pressure* from 296 to 4, discretize *precipitation* from 82 to 12; estimated improvement = 0.1375

Data encoded in the hint chain was used to reproduce its original mapping M_1. ViA then tried to apply the three hints. Color was swapped with height. Both discretizations were disallowed, however, because the search engine considered them too extreme relative to the original number of discrete values in the target attributes (i.e. discretizing *pressure* from 296 values to 4 makes *pressure* too coarse; similarly for discretizing *precipitation* from 82 values to 12).

The result was a new mapping M_2 that was then evaluated for perceptual salience. The evaluation engines reported a total normalized evaluation weight of 0.7475:

- *temperature* (w/7 discrete values, importance = 1.0) $\rightarrow$ height, weight = 0.17,
- *wind speed* (w/23 discrete values, importance = 0.75) $\rightarrow$ color, weight = 0.2025,

- *pressure* (w/296 discrete values, importance = 0.15) $\rightarrow$ density, weight = 0.1875, and
- *precipitation* (w/82 discrete values, importance = 0.75) $\rightarrow$ orientation, weight = 0.1875.

Interestingly, M_2 evaluated slight lower than M_1 (0.7475 for M_2 versus 0.755 for M_1). An examination of the results from each evaluation engines explains this result. The color evaluation engine evaluated *temperature* $\rightarrow$ color in M_1 to 0.1875, and suggested switching to height, since this visual feature is better suited to high spatial frequency data. Switching to *temperature* $\rightarrow$ height fixed the spatial frequency issue, but it introduced two new problems:

1. *temperature* $\rightarrow$ height: A less important attribute *(wind speed)* is being displayed with a more salient visual feature (color), which will cause visual interference.
2. *temperature* $\rightarrow$ height: The number of unique *temperature* is seven, but height only supports a maximum of five discrete values.

This caused the evaluation weight for *temperature* $\rightarrow$ height in M_2 to decrease to 0.17. Notice that *wind speed* $\rightarrow$ pressure actually increased 0.2025 (in was 0.1925 in M_1); this is because color supports more discrete values than height. Unfortunately, this increase was not enough to offset the decrease from the *temperature* $\rightarrow$ height pair.

This situation demonstrates an important design property: hints work in a completely independent fashion. They are meant to improve one specific property of the mapping being tested. Hints do not consider how their application can negatively impact other parts of the mapping. Although it might be possible build smarter hints, the number of potential interactions between groups of hints within a mapping is enormous. Trying to enumerate appropriate actions in all

possible cases would be very complicated. Instead, we rely on the search engine to selectively prune evaluation paths that do not lead to significantly better mappings. This allows us to keep the hints very simple, and significantly reduces the complexity of ViA's search algorithm. Results show that this strategy still produces high-quality mappings, and that these mappings are usually found early in the search process.

Evaluation of M_2 generated seven hints to address problems in the mapping:

1. $H_{2,1}$: Discretize *wind speed* (color) from 23 to seven, estimated improvement = 0.0475.
2. $H_{2,2}$: Discretize *temperature* (height) from seven to five, estimated improvement = 0.0175.
3. $H_{2,3}$: Feature swap *temperature* (height) with *wind speed* (color), estimated improvement = 0.0175.
4. $H_{2,4}$: Feature swap *temperature* (height) with *wind speed* (color), estimated improvement = 0.0625.
5. $H_{2,5}$: Importance weight modify *temperature* from 1.0 to 0.875 and *wind speed* from 0.75 to 0.875.
6. $H_{2,6}$: Discretize *pressure* (density) from 296 to four, estimated improvement = 0.0625.
7. $H_{2,7}$: Discretize *precipitation* (orientation) from 82 to twelve, estimated improvement = 0.0625.

The height evaluation engine returned the same hint twice ($H_{2,3}$ and $H_{2,4}$, feature swap height with color), with two different estimated improvement weights (0.0175 and 0.0625, respec-

tively). These hints were generated from two different tests on the *temperature* $\rightarrow$ height pair. The first test examined attribute type, and suggested that height and color swap since color supports more discrete values than height (seven versus five). The second test examined visual interference, and again suggested that height and color swap since *temperature* (currently bound to height) is a more important attribute than *wind speed* (currently bound to color). Both hints were used, but in different hint chains.

The seven hints from M_2 were combined into new hint chains that were added to the priority queue. The hint chain at the front of the queue was based on M_2, and contained four hints:

- $\{ H_{2,1}, H_{2,2}, H_{2,6}, H_{2,7} \}$: discretize *wind speed* from 23 to 7, discretize *temperature* from 7 to 5, discretize *pressure* from 296 to 4, discretize *precipitation* from 82 to 12; estimated improvement = 0.1625

M_2 was modified based on these hints. $H_{2,2}$ (discretize *temperature* from seven to five) was allowed; the other three hints were rejected for proposing too coarse a discretization. The result was a new mapping M_3 that evaluated to 0.7644:

- *temperature* (w/5 unique values, importance = 1.0) $\rightarrow$ height, weight = 0.1875,
- *wind speed* (w/23 unique values, importance = 0.75) $\rightarrow$ color, weight = 0.2025,
- *pressure* (w/296 unique values, importance = 0.15) $\rightarrow$ density, weight = 0.1875,
- *temperature* (w/82 unique values, importance = 0.75) $\rightarrow$ orientation, weight = 0.1875,

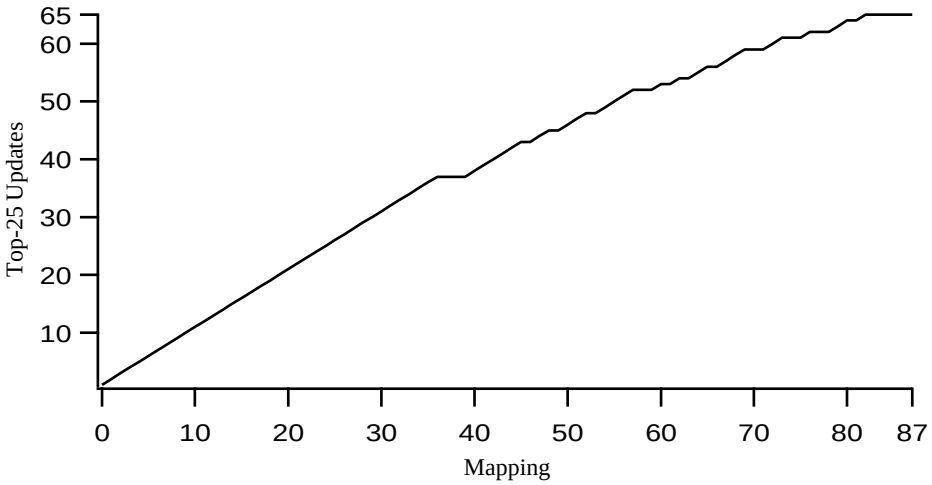


Figure 6.1: A histogram that shows when the top 25 mapping list is updated with improved mappings. Notice that much of the activity in the list occurs early in the search process; ViA is finding the best mappings quickly, suggesting that the hints chains and priority queue are working as expected to promote the most promising hints first

M_3 generated the highest evaluation weight to date, $W = 0.7644$. Discretizing *temperature* into five ranges allowed height to support it without penalty, producing a significant increase in evaluation weight compared to M_2.

Removing hint chains from the priority queue continues to generate new mappings, some of which evaluate even higher than M_3. In all, 87 unique mappings were tested by ViA. Recall that ViA maintains a list of the top 25 mappings to date. Figure 6.1 shows a histogram of when new mappings were added to the top 25 list. The histogram shows that much of the activity in the list occurs early in the search process. This means that ViA found many of the top mappings quickly, and suggests that the combination of hint chains and priority queue does an good job of identifying and applying the most promising sets of hints to generate the most salient mappings.

The final top 25 list returned by ViA contained a number of intriguing mappings, for example:

- M_4: *temperature* (w/5 unique values, importance = 0.875) $\rightarrow$ height, *wind speed* (w/12 unique values, importance = 0.875) $\rightarrow$ color, *pressure* (w/296 unique values, importance = 0.15) $\rightarrow$ density, *precipitation* (w/82 unique values, importance = 0.75) $\rightarrow$ orientation; total evaluation weight = 0.8444,

- M_5: *temperature* (w/5 unique values, importance = 0.875) $\rightarrow$ orientation, *wind speed* (w/12 unique values, importance = 0.875) $\rightarrow$ color, *pressure* (w/296 unique values, importance = 0.15) $\rightarrow$ density, *precipitation* (w/82 unique values, importance = 0.75) $\rightarrow$ height; total evaluation weight = 0.8444,

- M_6: *temperature* (w/5 unique values, importance = 0.875) $\rightarrow$ height, *wind speed* (w/12 unique values, importance = 0.75) $\rightarrow$ orientation, *pressure* (w/296 unique values, importance = 0.15) $\rightarrow$ density, *precipitation* (w/82 unique values, importance = 0.875) $\rightarrow$ color; total evaluation weight = 0.8162, and

- M_7: *temperature* (w/7 unique values, importance = 0.875) $\rightarrow$ color, *wind speed* (w/12 unique values, importance = 0.812) $\rightarrow$ orientation, *pressure* (w/296 unique values, importance = 0.15) $\rightarrow$ density, *precipitation* (w/82 unique values, importance = 0.812) $\rightarrow$ height; total evaluation weight = 0.8125.

We visualized three of the mappings, M_1, M_4, and M_5, using a two-dimensional nonphotorealistic visualization technique. Each weather reading was displayed using one or more simulated

brush strokes. The color, size, coverage (amount of canvas covered by the strokes), and orientation of the strokes corresponded to the color, height, density, and orientation features suggested by ViA.

Figure 6.2 shows data for February over the eastern United States visualized using M_1. In this mapping, color represents *temperature*. Bright pinks and reds denote higher temperatures in the south, while dark greens and blues show lower temperatures in the north. We can easily observe the expected higher-to-lower temperature variation as we move south-to-north across the map. The size of the strokes represents *wind speed*. Notice the area of stronger winds (with corresponding larger strokes) in the center-left area of the map. The coverage of the strokes represents *pressure*. The entire map is sparse (i.e. large amounts of the background canvas show through) since pressure is low across the entire country for the month of February. Finally, *precipitation* is represented by orientation: vertical strokes for low rainfall, and horizontal strokes for high rainfall. Areas of increased rainfall can be seen as tilted strokes in southern states like Georgia, Alabama, and Florida.

The use of color and orientation make *temperature* and *precipitation* relatively easy to identify. On the other hand, *wind speed* and *pressure* variations are harder to see. A mapping with a higher evaluation weight, M_4, attempts to address some of this imbalance by swapping color for *wind speed* and size for *temperature* (Figure 6.3). It is now very easy to identify the strong winds in the center of the country (shown as bright pink strokes). As well, *temperature* variations are still visible as large strokes in the south, and small strokes in the north.

Figure 6.2: Visualizing average weather conditions for February over the eastern United State using ViA mapping M_1: *temperature* $\rightarrow$ color, *wind speed* $\rightarrow$ size, *pressure* $\rightarrow$ density, and *precipitation* $\rightarrow$ orientation; all attributes are displayed in their original form

Figure 6.3: Visualizing average weather conditions for February over the eastern United State using ViA mapping M_4: *temperature* $\rightarrow$ height, *wind speed* $\rightarrow$ color, *pressure* $\rightarrow$ density, and *precipitation* $\rightarrow$ orientation; *temperature* was discretized to five ranges, and had its importance weight reduced to 0.875; *wind speed* was discretized to twelve ranges, and had its importance weight increased to 0.875

Figure 6.4: Visualizing average weather conditions for February over the eastern United State using ViA mapping M_5: *temperature* $\rightarrow$ orientation, *wind speed* $\rightarrow$ color, *pressure* $\rightarrow$ density, and *precipitation* $\rightarrow$ height; *temperature* was discretized to five ranges, and had its importance weight reduced to 0.875; *wind speed* was discretized to twelve ranges, and had its importance weight increased to 0.875

Finally, M_5 swaps orientation for *temperature* and size for *precipitation* (Figure 6.4). In this visualization *temperature* variations can be quickly distinguished as clearly visible difference in orientation. The use of size to denote *precipitation* does an excellent job of highlight the region of high precipitation in the south as a group of large strokes. As in M_4, *wind speed* is well represented with color. Finally, low *pressure* is shown as a sparse packing of strokes across the map.

This example provides a detailed description of how ViA was used to produce perceptually salient data-feature mappings for a collection of environmental weather conditions. In the next section we emphasize ViA's domain-independent nature by using it to visualize a completely different type of dataset containing results from a simulated e-commerce auction competition.

6.2 E-Commerce Visualization

The Trading Agent Competition[2] (TAC) is a simulated e-commerce auction environment run on the Michigan Internet AuctionBot platform[3]. The AuctionBot is a TCP-based auction server that implements numerous types of auction rules. This allows the simulation of a wide variety of market games. Intelligent auction agents are designed and tested within these markets to study different buying and selling strategies.

During the TAC, each agent acts as a "travel advisor" whose goal is to assemble a travel package for eight fictitious customers. A travel package consists of:

[2] `http://tac.eecs.umich.edu`
[3] `http://tac.eecs.umich.edu/auction`

- a round-trip flight from TACtown to Boston,
- a hotel reservation, and
- tickets to certain entertainment events (a baseball game, the symphony, and the theater).

Each customer specifies preferences for the different aspects of his or her trip. For example, customers will tell the agent which days they want to be in Boston, the type of hotel they prefer (economy or luxury), and the entertainment events they want to attend. There are obvious dependencies that must be met, for example, customers need hotel rooms for the duration of their trip, and can only attend entertainment events during that interval. The goal of the agent is to maximize the total satisfaction of its customers (i.e. the sum of their utility functions).

All three products (flights, hotels, and entertainment) are traded in separate markets with different auction rules. For example, the auction for airline tickets runs as follows:

1. A single airline (TACAIR) operates a single flight every day between TACtown and Boston; from the point of view of an agent, the supply of available seats is unlimited.
2. TACAIR runs a separate auction for each flight (i.e. for each day flights are being sold), with prices ranging from \$150 to \$600 dollars; a stochastic function with a uniform distribution permutes the price by $\pm$\$10 every 20 to 30 seconds,
3. The auctions run continuously until the simulation ends.
4. A buy bid by an agent is held within an auction until: (1) a sell price at or below the agent's bid is issued for the given auction, or (2) the auction ends.
5. Agents can withdraw or revise their bids at any time prior to a bid being accepted.

6. Agents *cannot* sell their own (previously purchased) tickets within the auction; only TACAIR can sell tickets.

Other auctions run with slightly different rules. For example, two hotels are available during the TAC: an economy hotel (Le FleaBag Inn) and a luxury hotel (Boston Grand Hotel). Both hotels offer sixteen rooms for each evening, with every hotel-evening pair run as a separate auction. The sixteen highest bids for an auction determine who receives rooms. An agent bids for one or more rooms at a chosen price (obviously, this price must exceed a minimum bid price, which is the sixteenth highest bid seen to date). Bids cannot be withdrawn, and only the hotel can offer to sell rooms. An auction ends when: (1) the simulation ends, or (2) a randomly chosen period of inactivity with no new bids passes (this was intended to penalize agents that try to wait until the end of the simulation, check the minimum bid price, then bid slightly above that price to secure the rooms they want). All the rooms are sold at the price of the sixteenth bid (i.e. agents with the highest bids will often pay less than they offered for their rooms).

Finally, every agent receives an initial allotment of tickets for each entertainment event. They can then buy and sell these tickets with other agents. As with hotels, a separate auction is held for each evening-event combination. The auctions run in a manner similar to the stock market: buy or sell requests that match an existing bid are executed immediately; otherwise they are held within the auction until an appropriate bid is received, or until the auction ends.

Although certain aspects of the TAC are simplified, it still provides an excellent simulation of a real-world e-commerce auction environment. Products are bought and sold in real-time, both by external suppliers and by the agents themselves. Careful planning is needed to manage

Table 6.2: Attributes from the TAC simulation visualized with ViA; each attribute's domain type, spatial frequency, analysis tasks to support, and relative importance as specified to ViA are listed in the table

Attribute	Domain	Frequency	Task	Importance
agent ID	discrete (8 unique values)	high	search	1.0
price	continuous	low	boundary	0.5
quantity	discrete (10 unique values)	high	estimate	0.5

the cost of certain items versus their potential unavailability (e.g. hotel rooms). Different auctions are run using different rules, requiring an agent to implement a variety of tactics to guarantee overall success.

The TAC has been used to study different auction strategies through head-to-head competitions. For example, teams of students in our undergraduate e-commerce course design and implement TAC agents, then compete against one another at the end of the term. In July 2000, twelve teams participated in a TAC run at the Fourth International Conference on Multiagent Systems (ICMAS-00). The agents at ICMAS were selected from an initial group of twenty teams from six countries that competed in a set of qualifying rounds conducted prior to the conference.

6.2.1 E-Commerce Mappings

Visualizing the datasets generated by the auction agents would allow participants and observers to track the performance of an agent as the simulation unfolds. Also, one would be able to observe underlying strategies of the different agents.

Based on feedback from the TAC designers and participants, the following attributes were

selected for visualization: *time, auction ID, agent ID, price* and *quantity* for every bid made during the simulation. We chose to use *time* and *auction ID* to define a bid's x and y position on an underlying two-dimensional grid (where the columns corresponds to a specific time step, and the rows corresponds to a specific auction). Table 6.2 shows the properties of the remaining data attributes to be visualized. These properties are used by ViA to identify appropriate data-feature mappings. For visualization purposes, and since the dimensionality of the dataset was fairly small, hue and luminance were combined into a single visual feature color. *q*uantity was allowed to be re-discretized, if necessary, down to as few as three discrete ranges. The attributes *agent ID* and *quantity* were not changed in any way. It must also be mentioned that hint chains were not incorporated into the version of ViA that was used for this application. Hints were applied individually to search for improved results. In spite of this, ViA was still able to suggest numerous effective mappings.

ViA evaluated a total of nineteen different mappings. This number was somewhat low, due in part to the inflexibility in the initial constraints specified by the user (i.e. only *quantity* was allowed to undergo any modification), and in part to the low number of attributes to be visualized. Also, any mapping that included regularity was discarded, since experimental results suggested that users would have difficultly distinguishing this feature in the visualization. Based on the perceptual salience of the mappings it generated, ViA returned the following promising candidates:

1. M_1: *agent ID* $\rightarrow$ color, *price* $\rightarrow$ height, *quantity* $\rightarrow$ density, with *quantity* re-discretized into four ranges of equal width; evaluation weight = 0.841.

2. M_2: *agent ID* $\rightarrow$ color, *price* $\rightarrow$ density, *quantity* $\rightarrow$ height; evaluation weight = 0.725.

3. M_3: *agent ID* $\rightarrow$ height, *price* $\rightarrow$ density, *quantity* $\rightarrow$ color; evaluation weight = 0.694.

For each of the above mappings, the evaluation engines generated warnings to highlight potential violations of the perceptual guidelines by different data-feature pairs within the mappings. These violations produced a corresponding reduction in the overall evaluation weight of each mapping. For example, evaluation of M_1 generated the following warnings and reductions:

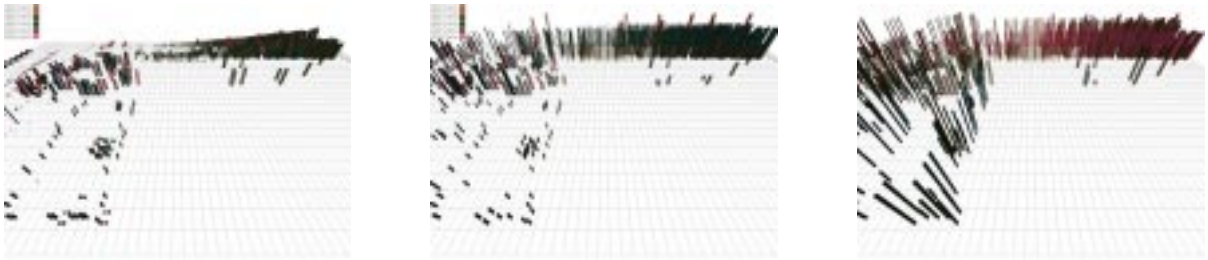
1. *agent ID* $\rightarrow$ color: The number of unique *agent ID* is eight, but color only supports a maximum of seven discrete values; penalty = -0.013.

2. *agent ID* $\rightarrow$ color: Color is not appropriate for high spatial frequency data; penalty = -0.083.

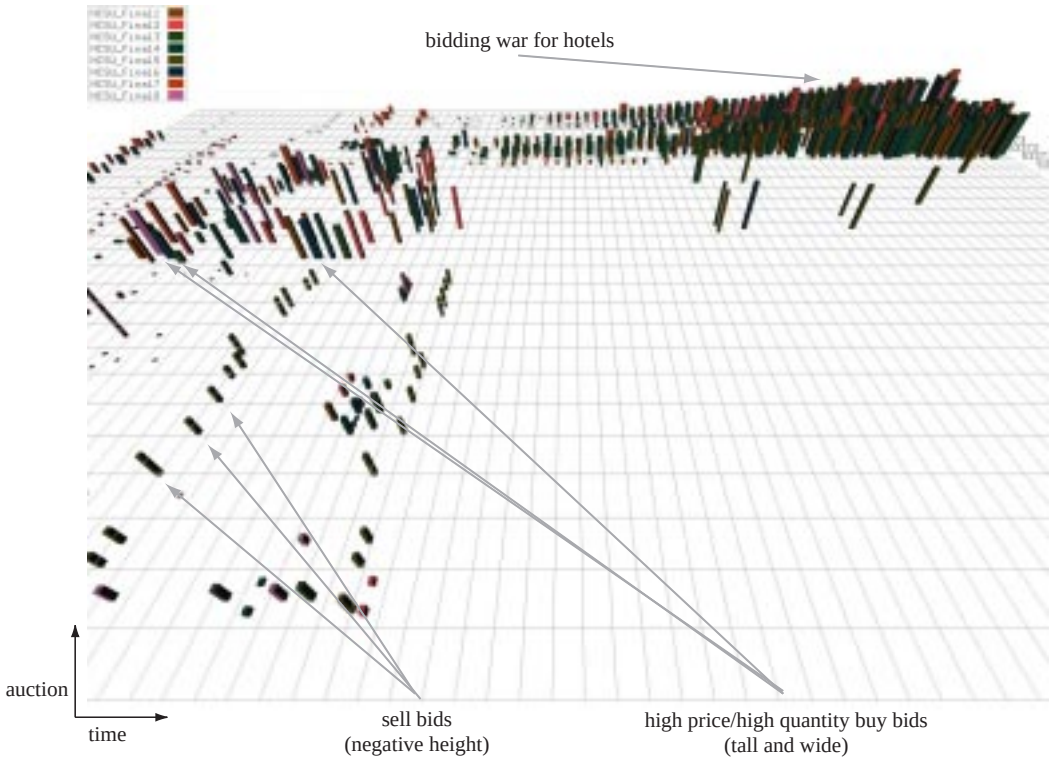
3. *quantity* $\rightarrow$ density: Density is not appropriate for high spatial frequency data; penalty = -0.063.

Figures 6.5a–6.5c show the visualizations for M_1, M_2, and M_3, respectively. Pexels [HE98] were used as the rendering model for visualizing these datasets. *time* is represented on the horizontal axis, increasing from left to right. Each row corresponds to a specific auction (each with a unique *auction ID)*.

In Figure 6.5a (representing M_1) viewers can identify the different colors representing the different agents. The price of each bid is represented by the height its pexel. This allows a viewer to compare the prices of different bids using height. Positive heights represent buy



(a) (b) (c)



(d)

Figure 6.5: Student TAC data visualized with four different M: (a) M_1: *agent ID* $\rightarrow$ color, *price* $\rightarrow$ height, *quantity* $\rightarrow$ density; (b) M_2: *agent ID* $\rightarrow$ color, *price* $\rightarrow$ density, *quantity* $\rightarrow$ height; (c) M_3: *agent ID* $\rightarrow$ height, *price* $\rightarrow$ density, *quantity* $\rightarrow$ color; (d) M_{final}: *agent ID* $\rightarrow$ color, *price* $\rightarrow$ height, *quantity* $\rightarrow$ width

bids (bids lying above the plane), and negative heights represent sell bids (bids lying below the reference plane). Notice in the upper right-hand corner a group of pexels with steadily increasing height. Since the upper rows of in the visualization represent hotel auctions, the pexels in this area of the visualization show a bidding war for hotel rooms. Viewers can clearly see the (poor) strategy of the agents as they repeatedly increase their bids in an attempt to secure hotel rooms by out-bidding one another.

M_2 and M_3 evaluated lower than M_1. The lower evaluation weights for these mappings resulted from slightly more severe violations of ViA's perceptual guidelines within M_2 and M_3.

M_2 had warnings identical to M_1 for its *agent ID* $\rightarrow$ color pair. Two additional warnings were generated:

1. *price* $\rightarrow$ density: ViA cannot map a continuous attribute to a discrete visual feature, unless the attribute is discretized; penalty = $= -0.083$.

2. *quantity* $\rightarrow$ height: The number of unique *quantity* is ten, but height only supports a maximum of five discrete values; penalty = -0.033.

Re-discretization of *quantity* to five or fewer ranges would eliminate the second violation in M_2, resulting in an improved evaluation weight of 0.758.

Figure 6.5b shows a visualization of the same data, but with M_2 as the data-feature mapping. In this representation different agents are still distinguished by different colors. However, *price* values are more difficult to differentiate, since they are now represented by the density of the pexels. This shows one potential disadvantage of M_2. Certain advantages also result from

this representation, however. Since *quantity* is represented by height, we can now see constant height values for each agent during the hotel bidding war. This shows that each agent was continually bidding for a constant number of rooms. The same pattern was visible with M_1 (in Figure 6.5a) as a constant density of pexels, but it was much less noticeable. This shows that there can be significant advantages to visualizing a dataset in more than one way.

M_3 had a lower evaluation weight as compared to both M_1 and M_2. This is due to the following violations that were detected by ViA:

1. *agent ID* $\rightarrow$ height: The number of unique *agent ID* is eight, but height only supports a maximum of five; penalty = -0.027.

2. *agent ID* $\rightarrow$ height: A less important attribute *(quantity)* is being displayed with a more salient visual feature (color), which will cause visual interference; penalty = -0.083.

3. *price* $\rightarrow$ density: ViA cannot map a continuous attribute to a discrete visual feature, unless the attribute is discretized; penalty = $= -0.083$.

4. *quantity* $\rightarrow$ color: The number of unique *quantity* is ten, but height only supports a maximum of seven; penalty = -0.030.

5. *quantity* $\rightarrow$ color: Color is not appropriate for high spatial frequency data; penalty = -0.083.

Looking at Figure 6.5c representing the visualization with data-feature mapping M_3, it is clear why it was evaluated as perceptually less salient than M_1 or M_2. Identification of the agents using height is much more difficult than with color. Constant colors for each participating

agent results from the fact that each agent only bids in a limited number of different quantities as per its customers requirements. Since density reflects the price of each bid, it is difficult to distinguish price ranges or compare differences in this value. The use of density would have be more effective if *price* were discretized into a small number of ranges.

A modified version of M_1 was chosen for final representation during live TAC sessions. Pexel width was used to represent *quantity*, rather than density. This allowed *quantity* to be represented without re-discretizing. It also allows us to sue spatial density to represent areas of high activity within the auction (i.e. specific time steps where numerous agents bid simultaneously in a common auction). Figure 6.5d shows a visualization with this mapping M_{final}. Since the visualizations were rendered in real-time as the TAC simulations unfolded, M_{final} allowed viewers to track different agents and their strategies, giving them the ability to analyze and explore the TAC in a rapid and effective manner.

Figure 6.6 shows a dataset from one of the ICMAS-00 TAC simulations. The same M_{final} is used to visualize the data. Finalists at ICMAS used much more sophisticated agent strategies, some of which are clearly visible in our visualizations. For example:

1. Some agents would periodically make very low buy bids for hotel rooms to ensure the hotel auctions did not close prematurely.

2. Most agents deferred purchasing hotel rooms and airline tickets until just before the simulation ended, since they felt there was no advantage to early purchase (particularly for hotel rooms, where attempts at early purchase can drive up the final price).

3. If hotel rooms for a given customer cannot be found, the customer's entire trip is can-

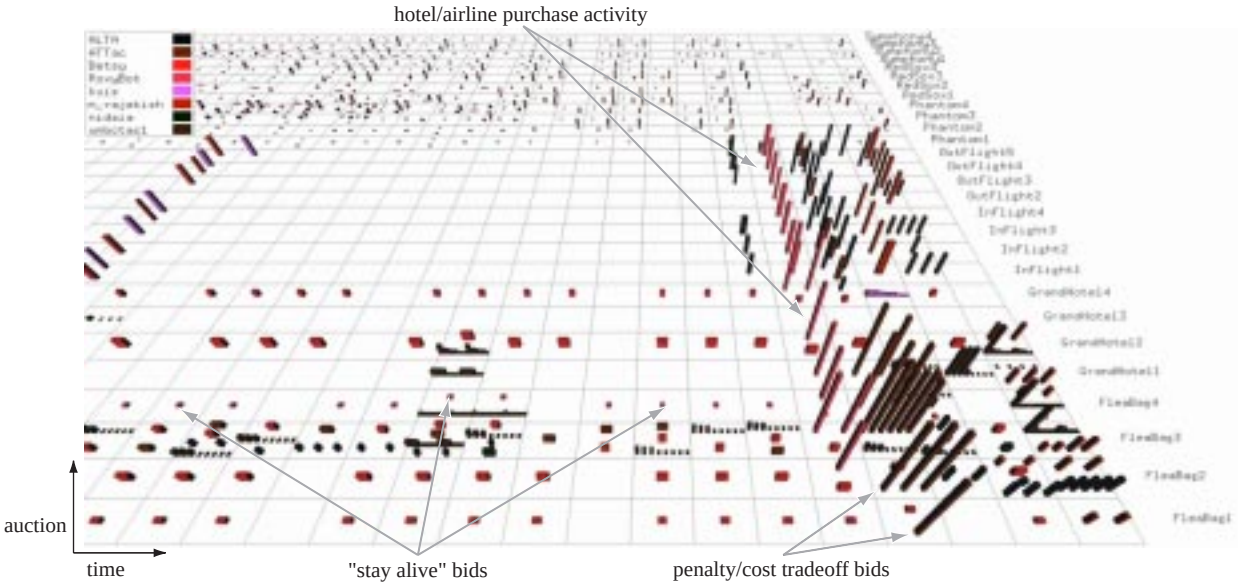


Figure 6.6: ICMAS TAC data visualized with M_{final} (*agent ID* $\rightarrow$ *color*, *price* $\rightarrow$ height, *quantity* $\rightarrow$ width); more sophisticated agent strategies are visible in these displays

celled, and the agent is penalized the cost of any airline and entertainment tickets they may have purchased on the customer's behalf. Some agents estimated the cost c of this penalty, then made late bids for hotel rooms at a buy price of c. These agents decided that paying c for a hotel room was no worse than paying a penalty of c for unused airline and entertainment tickets. More importantly, there is a good chance that the hotel rooms will sell for less than c (that is, the sixteenth winning bid for a room is made by some other agent with a buy price less than c). If this happens, the agent will make a profit relative to the scenario of not securing the hotel room.

This chapter discussed applying ViA to two real-world visualization environments: weather datasets, and e-commerce auction agent datasets. We examined the different stages that make up ViA's execution, and discussed the construction of the various mappings that were built,

evaluated, and suggested for use by ViA. We also identified the differences between these mappings, and explained how these differences affected ViA's evaluation procedures. The Conclusions chapter summarizes our findings, and suggests potential improvements to ViA that could result in a more effective and more efficient search process.

Chapter 7

Conclusions and Future Study

Multi-dimensional dataset visualization is fast emerging as an important area in the study of computer graphics and scientific visualization. The ability to explore and analyze large datasets is a critical requirement in our efforts to continually manage rapidly expanding collections of information from numerous problem domains. Supporting this need requires visual representations that aid users in comprehending and interacting with their data. Effective visualization techniques can significantly enhance a user's ability to conduct rapid, accurate, and effortless analysis.

This thesis proposes a combination of guidelines from human vision and intelligent search strategies to assist in constructing effective multidimensional visualizations. Our system, called ViA, is a flexible visualization tool that is used to build perceptually salient visualizations for multidimensional datasets. ViA supports a wide variety of dataset types and user-specified constraints on the visualizations it suggests. Use of human perception as a foundation for

design allows ViA harness the strengths and avoid the limitations of low-level human vision. The result is a small collection of data attribute to visual feature mappings that best represent the user's data for the given analysis tasks.

ViA works by constructing candidate data-feature mappings, then evaluating the mappings to produce evaluation weights and hints. An evaluation weight represents a measure of a mapping's appropriateness for a given dataset and analysis tasks. Evaluation weights also allow us to compare different mappings.

Hints offer suggestions on how to correct weaknesses in a mapping. Hints are used to guide the search along paths that have the highest probability of producing better mappings. Individual hints are combined into hint chains, and applied in a single step to a candidate mapping. This produces rapid improvements in the mappings ViA generates. Results suggest this strategy is effective at quickly moving ViA to areas of the search space with the best possible mappings.

ViA was designed to be domain-independent. In particular, neither the perceptual guidelines nor the search algorithms depend on a particular type of dataset or problem environment. We applied ViA to two different domain areas, weather datasets and e-commerce datasets, to demonstrate this flexibility. In both cases ViA produced high-quality data-feature mappings that support the exploration and analysis tasks specified by our viewers. We expect that ViA's use of perceptual guidelines and intelligent search strategies will be effective for a wide range of real-world problem domains.

There are other promising areas of study that may further improve ViA's ability to act as a

flexible, robust visualization assistant. Two areas of particular interest are:

1. Research is currently underway to identify the perceptual properties of new visual features (*e.g.*, properties of motion like flicker, direction, and velocity). Support for addition features can easily be added to ViA through the addition of new evaluation engines. Having access to more visual features would allow ViA to suggest higher-dimensional, and potentially more effective visualizations.

2. We want to take advantage of a user's domain expertise and understanding of context to help ViA decide how to best relax constraints during the search for effective visualizations. In particular, we hope to implement a mixed-initiative strategy that asks about changing the dataset's properties or the user's initial constraints. In order to be effective, such a strategy must make only a very small number of queries, with each query resulting in a significant improvement in ViA's ability to search for better mappings. Work currently underway is building a theoretical framework to control user interactions in a way that will satisfy these goals.

In summary, ViA represents the first step towards designing a domain-independent tool to assist users in constructing perceptually salient multidimensional visualizations. Results to date have been positive. We expect future work will only improve upon these successes.

Bibliography

[AC97] R. St. Amant and P. L. Cohen. Interaction with a mixed-intiative system for exploratory data analysis. *Proceedings Intelligent User Interfaces '97*, pages 15–22, 1997.

[BCE+92] K. W. Brodlie, L. A. Carpenter, R. A. Earnshaw, J. R. Gallop, R. J. Hubbard, A. M. Munford, C. D. Osland, and P. Quarendon. Scientific visualization: Techniques and applications. In *An Introductory Guide to Scientific Visualization*, New York, New York, 1992. Springer-Verlag.

[BF93] C. G. Beshers and S. K. Feiner. AutoVisual: Rule-based design of interactive multivariate visualizations. *IEEE Computer Graphics and Applications*, 13(4):41–49, July 1993.

[BJC96] B. Bauer, P. Jolicoeur, and W. B. Cowan. Visual search for color targets that are not linearly-separable from distractors. *Vision Research*, 36:1439–1446, 1996.

[D'Z91] M. D'Zmura. Color in visual search. *Vision Research*, 31(6):951–966, 1991.

[FvF+93] J. D. Foley, A. van Dam, S. K. Feiner, J. F. Huges, and R. L. Philips. *Introduction to Computer Graphics*. Addison-Wesley Publishing Company, Reading, Massachusetts, 1993.

[Gal94] J. Gallop. Underlying data models and structures for visualization. In L. Resenblum, editor, *Scientific Visualization: Advances and Challenges*, pages 87–102, San Diego, California, 1994. Academic Press.

[GPW89] G. Grinstein, R. Pickett, and M. Williams. Exvis: An exploratory data visualization environment. *Proceedings Graphics Interface '89*, pages 254–261, 1989.

[HE98] C. G. Healey and J. T. Enns. Building perceptual textures to visualize multidimensional datasets. *Proceeding Visualization '98*, pages 111–118, 1998.

[HE99] C. G. Healey and J. T. Enns. Large datasets at a glance: Combining textures and colors in scientific visualization. *IEEE Transactions on Visualization and Computer Graphics*, 5(2):145–167, 1999.

[Hea96] C. G. Healey. Choosing effective colours for data visualization. *Proceeding Visualization '96*, pages 263–270, 1996.

[Hea98] C. G. Healey. Perceptual colors and textures for scientific visualization. In *Applications of Visual Perception in Computer Graphics*, pages 205–224, 1998.

[HM90] R. B. Haber and D. A. McNabb. Visualization idioms: A conceptual model for scientific visualization systems. In G. M. Nielson, B. Shriver, and L. J. Rosenblum, editors, *Visualization in Scientific Computing*. IEEE Computer Society Press, 1990.

[JB83] B. Julész and J. R. Bergen. Textons, the fundamental elements in preattentive vision and perception of textures. *The Bell System Technical Journal*, 62(6):1619–1645, 1983.

[KML99] R. M. Kirby, H. Marmanis, and D. H. Laidlaw. Visualizing multivalued data from 2d incompressible flows using concepts from painting. *Proceedings Visualization '99*, pages 333–340, 1999.

[KUU95] M. Kawai, K. Uchikawa, and H. Ujike. Influence of color category on visual search. *Annual Meeting of the Association for Research in Vision and Ophthalmology*, 1995.

[LAK+98] D. H. Laidlaw, E. T. Ahrens, D. Kremers, M. J. Avalos, R. E. Jacobs, and C. Readhead. Visualizing diffuse tensor images of the mouse spinal cord. *Proceedings Visualization '98*, pages 127–134, 1998.

[LP88] D. Lubinsky and D. Pregibon. Data analysis as search. *Journal of Econometrics*, 38:247–268, 1988.

[LRBW90] G. L. Lohse, H. Rueter, K. Biolsi, and N. Walker. Classifying visual knowledge representations: A foundation for visualization research. *Proceedings Visualization '90*, pages 131–138, 1990.

[Mac86] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Transaction Graphics*, 5(2):110–141, 1986.

[RD94] P. K. Robertson and L. De Ferrari. Systematic approaches to visualization: Is a reference model needed? In L. Resenblum, editor, *Scientific Visualization: Advances and Challenges*, pages 239–250, San Diego, California, 1994. Academic Press.

[RL93a] A. R. Rao and G. L. Lohse. Identifying high level features of texture perception. *CVGIP: Graphics Models and Image Processing*, 55:218–233, 1993.

[RL93b] A. R. Rao and G. L. Lohse. Towards a texture naming system: Identifying relevant dimensions of texture. *Proceedings Visualization '93*, pages 220–227, 1993.

[RN95] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Inc., Upper Saddle River, New Jersey, 1995.

[Rob90] P. K. Robertson. A methodology for scientific data visualization: Choosing representations based on natural scene paradigm. *Proceeding Visualization '90*, pages 114–123, 1990.

[Rob91] P. K. Robertson. A methodology for choosing data representations. *IEEE Computer Graphics and Applications*, 11(3):56–67, 1991.

[RT90] P. Rheinigans and B. A. Tebbs. A tool for dynamic explorations of color mappings. *Computer Graphics*, 24(2):145–146, 1990.

[RT93] B. A. Rogowitz and L. A. Trienish. An architecture for rule-based visualization. *Proceeding Visualization '93*, pages 236–243, 1993.

[SI94] H. Senay and E. Ignatius. A knowledge-based system for visualization design. *IEEE Computer Graphics and Applications*, 14(6):36–47, 1994.

[TB96] G. Turk and D. Banks. Image - guidened streamline placement. *SIGGRAPH 96 Conference Proceedings*, pages 453–460, 1996.

[Tri85] A. Triesman. Preattentive processing in vision. *Computer Vision, Graphics and Image Processing*, 31:156–177, 1985.

[WK95] C. Ware and W. Knight. Using visual texture for information display. *ACM Transactions on Graphics*, 14(1):3–20, 1995.

[WL90] S. Wehrend and C. Lewis. A problem oriented classification of visualization techniques. *Proceedings Visualization '90*, pages 139–143, 1990.