FAEJ3D
A File Attribute Explorer in Java3D

Harald Koehler

Institute for Information Processing and
Computer Supported New Media (IICM)

Graz University of Technology
A-8010 Graz, Austria

February 2002

Masters Thesis
at
Graz University of Technology

Advisor: Univ.Ass. Dr. Keith Andrews

FAEJ3D
Ein File Attribut Explorer mit Java3D

Harald Koehler

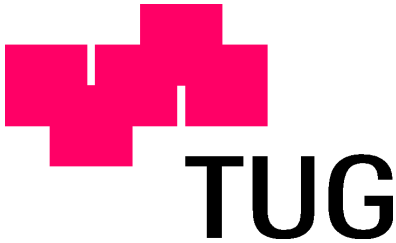
Institut für Informationsverarbeitung und
Computergestützte neue Medien (IICM),

Technische Universität Graz
A-8010 Graz, Austria

Februar 2002

Diplomarbeit
an der
Technischen Universität Graz

Betreuer: Univ.Ass. Dr. Keith Andrews



Abstract

Information visualization has many advantages compared to a pure textural representation, but why and how can information visualization be used for a certain goal?

This thesis gives an overview of the physiological and psychological aspects of seeing and perceiving data. Some popular approaches in information visualization are described.

A prototype file attribute visualizer was implemented in Java3D. The File Attribute Explorer in Java3D (FAEJ3D) displays files as glyphs in 3D space with extrinsic and intrinsic mappings. The user can freely explore the space and change mappings dynamically.

A secondary aim was to investigate the suitability of Java3D to such information visualizations. In this regard, it can be said that Java3D still needs to improve, in particular with regard to efficiency and bugs.

Kurzfassung

Informations Visualisierung hat viele Vorteile gegenüber einer rein textlichen Darstellung. Aber warum und wie kann man Informations Visualisierung benutzen um ein bestimmtes Ziel zu erreichen?

Diese Diplomarbeit gibt einen Überblick über physiologische und psychologische Aspekte des Sehens und Wahrnehmens. Einige bekannte Ansätze der Informations Visualisierung werden beschrieben.

Ein Prototyp eines File Explorers wurde in Java3D erstellt. Der File Attribute Explorer (FAEJ3D) zeigt Glyphen im 3D Raum mit extrinischen und intrinsischen Mappings. Der Benutzer kann sich frei im 3D Raum bewegen und die Mappings nach seinen Bedürfnissen dynamisch ändern.

Als sekundäres Ziel wurde untersucht ob Java3D geeignet ist für solche Informations Visualisierungen. In diesem Zusammenhang, kann man sagen das Java3D noch einige Verbesserungen besonders bezüglich Effektivität und Fehler benötigt.

Contents

List of Figures

Acknowledgments

First of all I want to thank my parents who supported me every time during my studies.

Many thanks also to my beloved girlfriend Pamela even when we were physically distant I felt her close to me which gave me strength in periods where I felt weak.

Special thanks to Keith Andrews who is not only my advisor who had always time for me and my questions he is also a friend and "responsible" for my interest in the human computer science.

Herzlichen Dank liebe Eltern für Eure Unterstüzung!

Pamela, grazie per essere così come sei!

Preface

In 1993 I started to study telematics at Graz University of Technology. This was a choice I never regretted. Studying was hard but always interesting. Computer graphics and its applications was always my favorite section in the wide field of telematics.

The final input for me came in April 2000 at a conferences from the ACM, CHI 2000 in The Hague. Thanks to Keith Andrews who encourages students to be volunteers at this conference. I attended it and was overwhelmed by all this new experiences. I visited the tutorial from Colin Ware, Rich Gossweiler, and Ed Chi: "Perception and Data Visualization".

Right after it I considered writing my thesis about this topic. I asked Keith Andrews if he wants to be my advisor, he agreed and we selected Java3D for the practical part of the thesis.

Now my thesis is ready and I can say that apart from some nights spend with Java3D bugs, I was always pleased to have selected this topic.

Harald Koehler, Graz October 2001

Preface from Ed Chi from XEROX Palo Alto Research

The lifeline of academia is the free exchange of ideas between scholars, students, and industrial researchers. We, as researchers, do this by attending conferences and publishing papers, after assured of the validity of the results. The conferences serve not only to disseminate ideas, but also to enable students to get a chance to talk to researchers around the world and expose to ideas that are foreign (literally and figuratively) to their own.

I met Harry at the CHI2000 conference in The Hague, where he eagerly sought out my advice on how to pursue his academic studies. He had a strong interest in the visuals, and was interested in the physiological mechanisms of visual perception. Being a visualization researcher myself, I spend some time with him to impart some of my understandings. To this end, I feel that we have succeeded in fulfilling the goal of exchange of ideas.

Visualization is built on the premise that the human visual channel is one of the most high bandwidth senses humans have. In order to test this premise and bring it to fruition, we must examine various avenues of research, including physiological basis of vision, perceptual illusions, computer graphics, human computer interaction, human motor control, and system sciences. Harry's thesis examines some of the system science issues by studying how to build information visualizations using the Java3D graphic system. While it is still unclear whether Java3D will succeed, Harry demonstrates the viability of the approach. To this end, he helps researchers understand a piece of this gigantic puzzle.

Ed

I hereby certify that the work presented in this thesis is my own and that work performed by others is appropriately cited.

Ich versichere hiermit, diese Arbeit selbständig verfaßt, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und mich auch sonst keiner unerlaubten Hilfsmittel bedient zu haben.

Credits

- Titlepage FOLDED CHESS, courtesy of Sandro Del-Prete Bern Switzerland

Chapter 1

Introduction

This thesis introduces information visualization from physiological and psychological backgrounds and gives an overview of existing applications. Information visualization is exploiting the power of graphical computers to enhance the users cognitive abilities. These graphics are normally interactive and the users can find new information or confirm prior made assumptions about their data.

Java3D was selected as the experimental platform because of its system independence and also to explore weather Java3D can be used to realize the theory of visual perception.

Chapter 2 gives an overview of human-computer interaction. Hardware and basic ergonomics are described. The physiological details of the human eye and visual pathway are discussed. Chapter 3 builds on this to discuss the wide field of visual perception and the psychological reasons why things are perceived as they are even if they are physically different. Chapter 4 covers information visualization techniques theory, defining the goals for visualizations and giving an overview of existing research visualization systems.

Chapter 5 is an introduction to the Java3D language and its concepts. It will be shown how the Java3D scene graph concept works and how it can be installed and used.

The File Attribute Explorer in Java3D (FAEJ3D), the practical part of the thesis is described in Chapter 6. This chapter demonstrates the implemented concept of reusability and how the goals of visualizing file attributes are achieved.

Chapter 7 discusses some of the implemented ideas and algorithms in detail, for example the scene graph of the FAEJ3D. Chapter 8 concludes the thesis with an outlook to possible future usage and additions to FAEJ3D. Appendix A contains a user guide for the FAEJ3D application.

Chapter 2

Human-Computer Interaction

"*Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them.*"
[ACM SIGCHI Curricula for Human-Computer Interaction]

Human-computer interaction (HCI) as a science comprises parts of many sciences [ACM92]. HCI (see Figure 2.1) emerged as a field from different roots in operating systems, computer graphics, human factors, ergonomics, industrial engineering, cognitive psychology. Computer graphics was born from the use of cathode ray tube (CRT) and pen devices early in the history of computers. This led to the development of numerous human-computer interaction techniques.

2.1 Hardware

Today the main interaction between users and computer hardware is with the screen, with the keyboard and mouse as input devices.

Typical monitors today are CRT having a diagonal size of 17 inches and a frequency range of about 30-96 kHz horizontal and 50-150 Hz vertical, which guarantees a flicker free image. A typical monitor at the moment is set to a resolution of 1024x786 pixels.

The typical computer user has a QWERTY keyboard as shown in Figure 2.2, and a two or three button mouse. The name QWERTY comes from the layout of the keys.

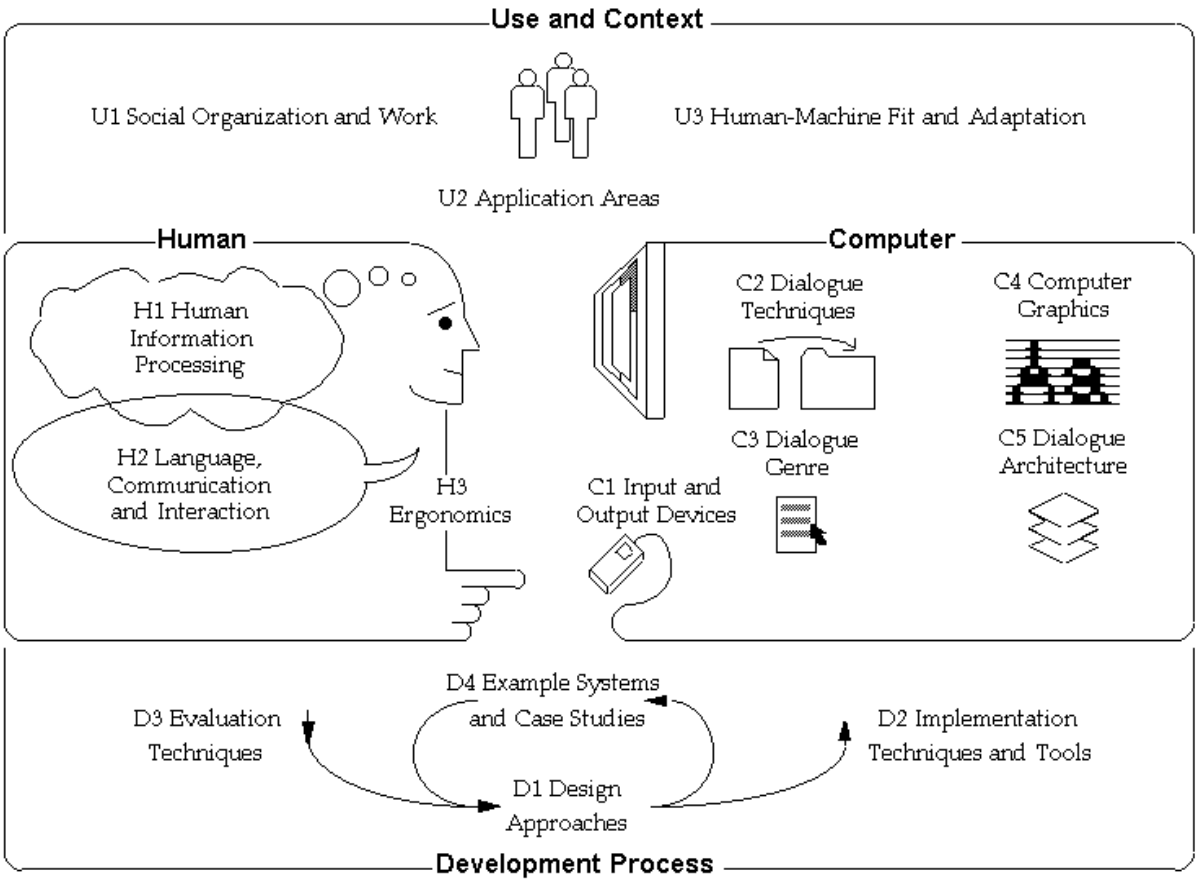


Figure 2.1: The field of human-computer interaction. (Source: [ACM92])

Figure 2.2: Typical computer hardware for interaction today: monitor, keyboard and mouse.

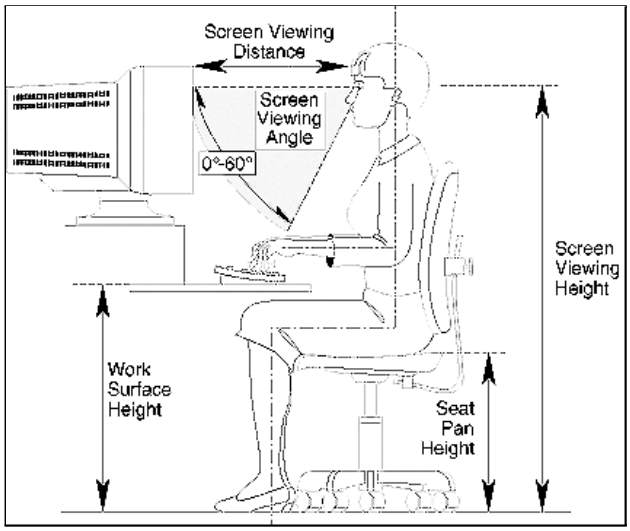


Figure 2.3: ANSI Standards for Visual Display Terminal (VDT) Workstations.

2.2 Ergonomics

There is an ANSI Standard for the Visual Display Terminal [ANSI88], for example for the operator shown in Figure 2.3. This standard is the result of research concerning perception and anatomy to produce optimal working conditions for the human user. A summary can be seen in Table 2.2.

Adjustment	**Mean cm**	**Range cm**
Seat pan height	54	50-57
Work surface height	73	70-80
Monitor center above floor	113	107-115
Screen viewing distance	70	59-78
Work surface tilt	8.6 degrees	2-13 degrees
Monitor tilt	-7.7 degrees	-15- +15 degrees

Table 2.2 ANSI Standards for Visual Display Terminal (VDT) Workstations.

Ergonomics are important since wrong placement or illumination of the monitor can easily disturb the use. See [ISO19] for equivalent ISO norms.

2.3 The Human Eye

The human eye is a sense organ capable of receiving visual images, which are then transported to the brain. It is essential to understand the eye and the visual pathway in order to understand many of the principles of perception. The eye receives the amazing amount of 10^7 bits of information in a second [Holz00a].

The eyeball is not a perfect sphere, it is the result of fusing a small portion of a small, curved sphere with a large portion of a large, less curved sphere as seen in Figure 2.4. The eye is made up of three coats, the outermost coat consists of the cornea and the sclera; the middle coat contains the main blood supply to the eye and consists, of the choroid, the ciliary body, and the iris [Snel98]. Looking directly into the eye from in front one sees the white sclera surrounding the cornea; but instead of seeing the cornea, a ring of tissue lying within the eye, the iris is seen. The

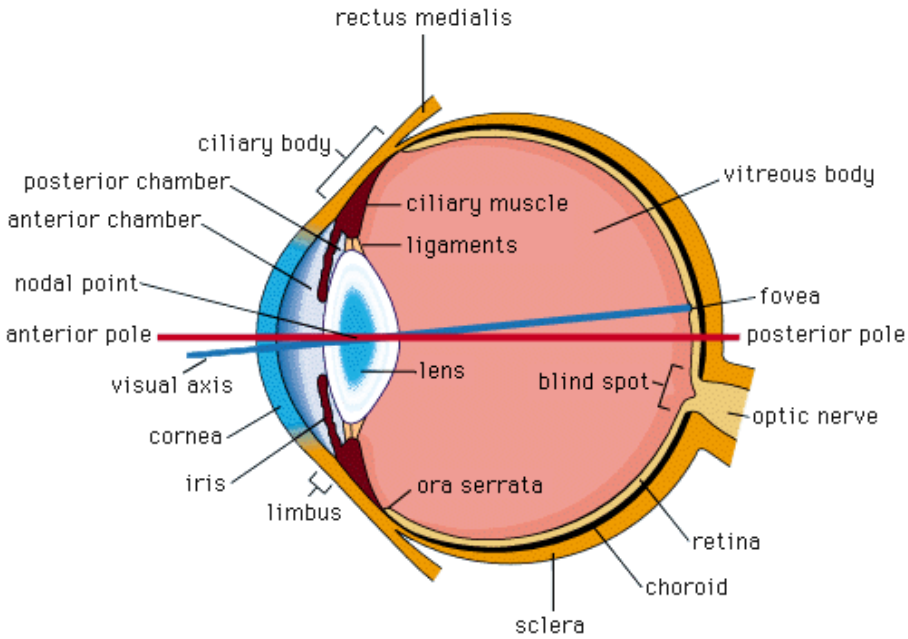


Figure 2.4: Horizontal section of the eye. (Source: [HHMI01])

iris determines the color of the eye. The centre of this ring is called the pupil, this centre appears dark because the light passing into the eye is only reflected back in a small amount [Greg90]. The iris changes in size as a function of the amount of light. Bright light makes the pupil smaller and the iris gets larger whereas and a small amount of light makes the pupil larger and the iris smaller. See Figure 2.5.

Behind the iris is the lens, the clear part of the eye that helps to focus light on the retina. With the lens humans can focus on both far and near objects so that they are perceived clearly and sharply. This focussing is done with the ciliary muscle which helps to change the shape of the lens. This changing of lens shape is called accommodation. See Figure 2.6.

The retina is the part of the eye which receives the light and converts it into chemical energy. The chemical energy activates nerves that conduct the messages out of the eye into the higher regions of the brain.

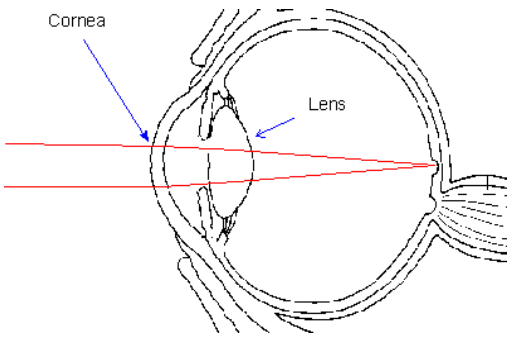


Figure 2.5: The function of the cornea and the lens. Adapted from [NEIN01].

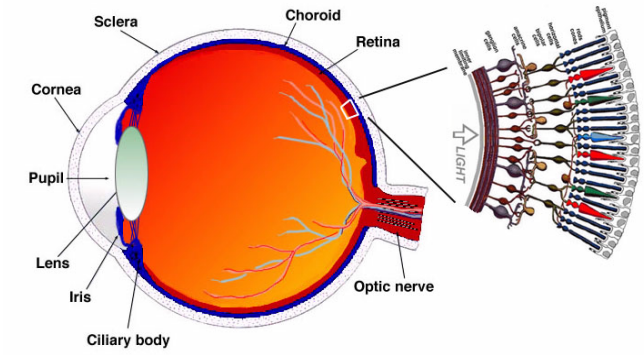


Figure 2.6: Schematic section of the eye, with an enlargement of the retina. (Source: [HHMI01])

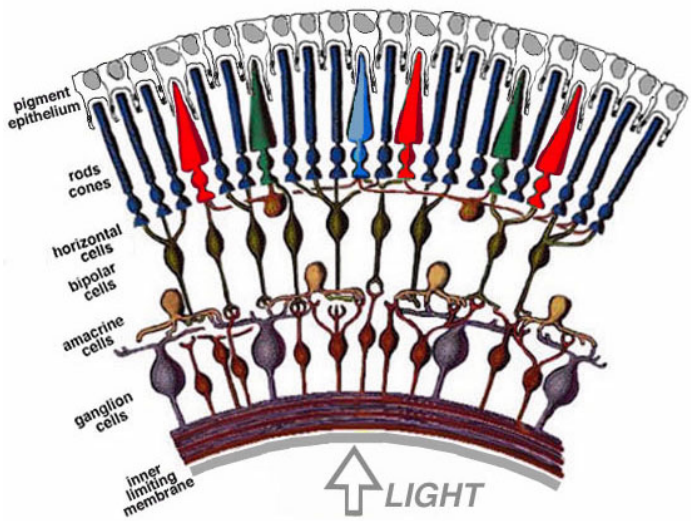


Figure 2.7: The retina in detail. (Source: [HHMI01])

The small sensitive area of the retina that gives central vision is called the macula and contains the fovea. This foveal area is covered with a yellow pigment and referred as the "yellow spot". When fixating or looking directly at an object it is imaged on the fovea and so on the center of the macula humans have the sharpest vision.

The retina consists of different layers of cells which can be seen microscopically. In general, there are four main layers. Next to the choroid is the pigment epithelium. Below the epithelium are the light-sensitive cells, the layer of *rods* and *cones*. Light changes in the rods and cones are transmitted to a layer of neurons (nerve cells) called the bipolar cells, these bipolar cells connect with the innermost layer of neurons, the ganglion cells. These transmitted messages are carried out of the eye along their projections, or axons, which constitute the optic nerve fibres [Thal95]. See Figure 2.7.

In the whole human retina there are about 7,000,000 cones and up to 150,000,000 rods with a spatial distributed as shown in Figure 2.8. The blind spot in the retina corresponds to the optic papilla, the region on the nasal side of the retina through which the optic nerve fibres pass out of the eye. There are at least 150,000,000 receptors but only about 1,000,000 optic nerve fibres, so that there must be strong convergence of receptors during the optic pathway, which means that

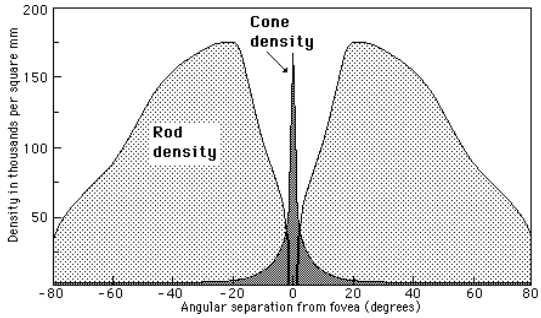


Figure 2.8: The Rod and Cone Density on Retina. (Source [Wand95])

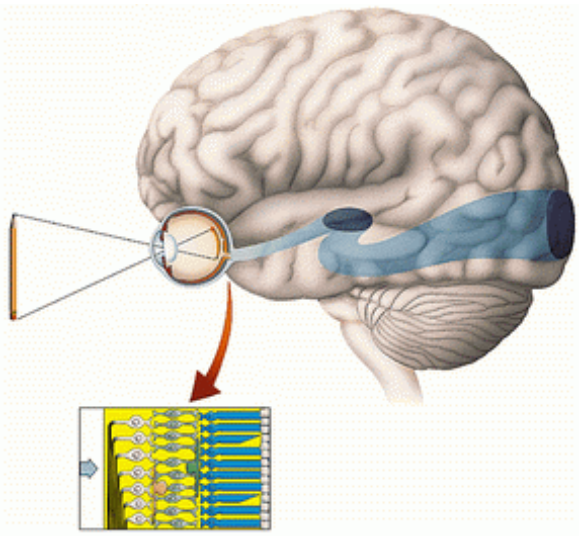


Figure 2.9: The visual pathway. (Source: [HHMI01])

there will be considerable mixing of messages [Wand95].

2.4 The Visual Pathway

Figure 2.9 shows how light from an object (here for example, a pencil) enters the eye and passes through its lens. As the light rays in Figure 2.9 show the image is projected inverted the retina at the back of the eye. The signals produced by rods and cones then start on their way into the brain, through the optic nerve and reach a major communication station, named the LGN (lateral geniculate nucleus). This route is called the visual pathway [HHMI01].

The produced signals from particular elements from the seen object then travel to selected areas of the primary visual cortex, or V1, which curves around a deep fissure at the back of the brain. Then the signal is split up and processed in "higher" areas of cortex that deal with more global aspects of the object such as its color, shape, or motion.

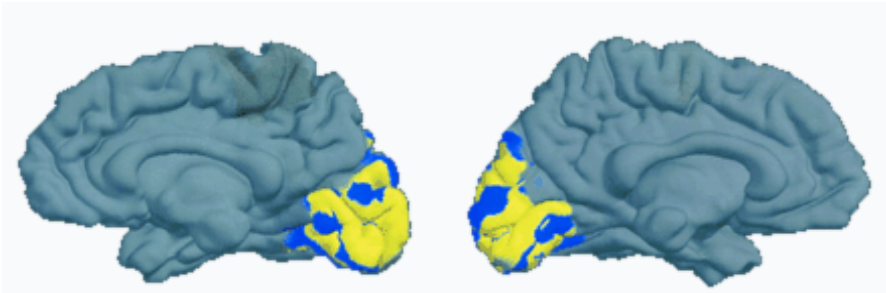


Figure 2.10: The Visual Cortex colored blue and yellow. (Source: [Toot98])

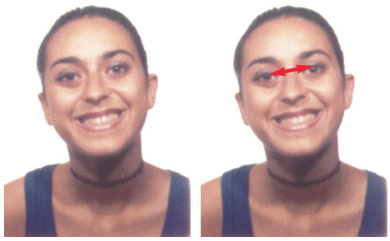


Figure 2.11: The eye distance.

2.5 The Visual Cortex

For a better understanding of the function of the visual cortex, the neurons of this region are distinguished by the stimulus features that each detects. These so-called *feature detectors* are divided into three major groups *simple cells, complex cells*, and *hypercomplex cells*.

The most specific are the simple cells, they are responding to lines of particular width, orientation, angle, and position within the visual field. Similar to the simple cells are complex cells, they respond to the proper stimulus in any position within their receptive field and in addition some complex cells respond particularly to lines or edges moving in a specific direction across the receptive field. Finally, hypercomplex cells are responsive to lines of specific length.

Then the information from all feature detectors combine to result in the perception of visual stimulation. Figure 2.10 shows the topography of the primary visual cortex and surrounding areas [Toot98].

2.6 Visual Fields

The human has a pair of eyes with a typical distance between them of 6 cm [Wand95] and halfway down the head as shown in Figure 2.11. Closing one eye will show that the range of vision in the remaining eye is mainly limited by the nose. With the fovea as center there can be defined two halves of the retina, the temporal half next to the temple and the nasal half next to the nose. See Figure 2.12.

The lens of the eye inverts the visual images due to the optical laws. Figure 2.12 shows that now in the right eye, the nasal retina receives the right half ,and the temporal retina receives the left half of the world. Information about the world enters both eyes with a great deal of overlap it can be seen that the right nasal retina and the left temporal retina receive almost the same thing. Drawing a line through the world at the nose, results into two *hemifields*. Each eye sees information from both hemifields and this is essential for the perception of depth [Thal95].

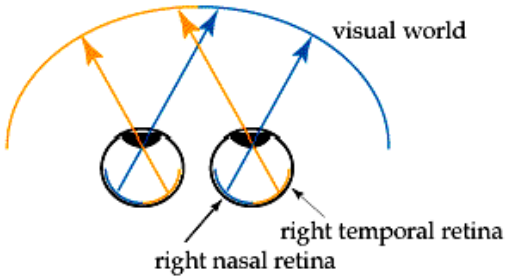


Figure 2.12: Visual fields. (Source: [Thal95])

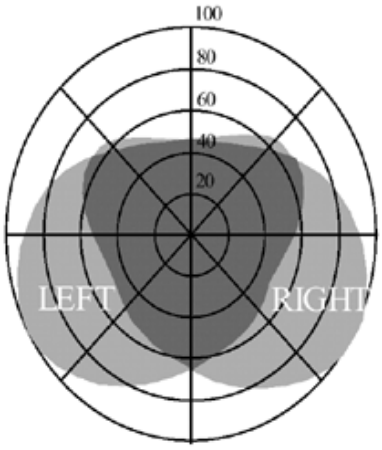


Figure 2.13: The human visual field. (Source: [Ware00])

The human eye accepts light through a wide field of view as shown in Figure 2.13. The irregular boundaries are caused by facial features such as the nose. The dark area shows the region of binocular overlap.

2.7 The Visual Angle

A major property of the eye is the visual angle. Visual angles are normally defined in degrees, minutes, and seconds of an arc. As rule of thumb it can be said that a thumbnail held at arm's length is about 1 degree.

Visual acuity can be defined with the 6/6 acuity, this means that at a distance of 6 meters a person with normal view is able to decipher a letter that subtends a visual angle of 5 minutes of arc Experiments have shown that people can localize the position of a line to a spatial resolution of 6 seconds of visual angle. The visual angle is labeled Θ in Figure 2.14.

2.8 Color Blindness

About 10% of men and 0.4% of women have the red/green form of color blindness and 0.001% of the male population the blue/yellow form. Color blindness has to be considered if the visualization is critical and particularly if it is mainly used by men [Rutg01].

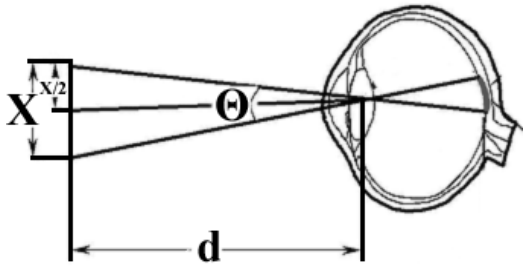


Figure 2.14: The visual angle.

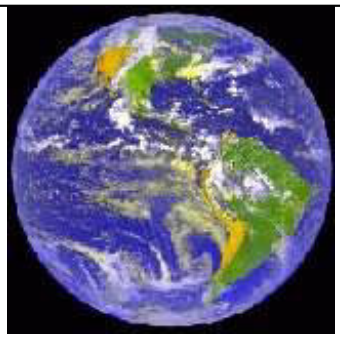
The world	How the world looks to a person with a red/green color deficit (deuteranopia).	How the world looks to a person with a blue/yellow color deficit (tritanopia).
		
		

Table 2.4 Types of color blindness. (Source: [VISC01])

2.9 Light and Optics

The theory of evolution states that the visual system must have survival value, and adopting this perspective allows an understanding of visual mechanisms in the broader context of useful skills. Visible light is only a small part of the electromagnetic spectrum, as shown in Figure 2.15. Wavelengths directly below 400 nm are called ultraviolet light and longer then 700 nm is infrared light. See Figure 2.16.

During interaction with object light is absorbed, reflected, refracted, and diffracted. This optical flow changes over time and with movement which helps to interpret the 3D location. The key point here is that visual images of the world are dynamic. Textured surfaces are also a

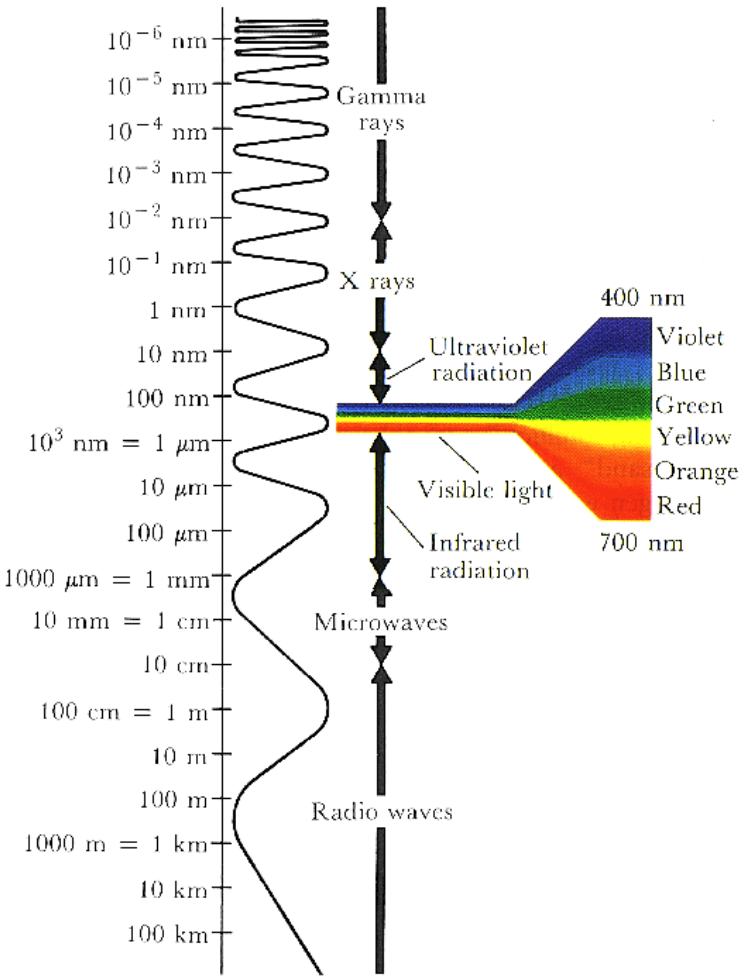


Figure 2.15: The electromagnetic spectrum. (Source: www.eb.com)

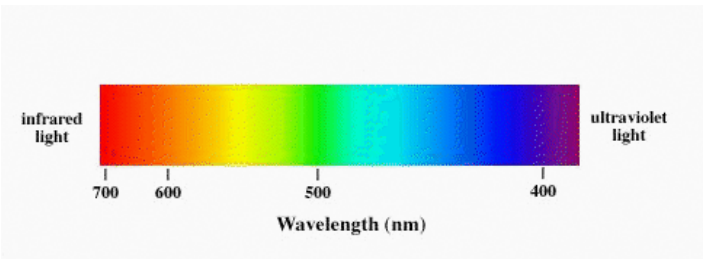


Figure 2.16: The visual spectrum. (Source: www.eb.com)

Figure 2.17: A natural texture.

fundamental visual property of an object and assist in the judgement of distances, as shown in Figure 2.17.

Shading models include the Lambertian shading model alone or with specular shading and ambient shading, or all together with the cast shadows model. The Lambertian shading model (see Figure 2.18) has a directional light source and the reflected light depends on the angle between the normal vector and the incoming light.

The specular shading model adds highlights on surfaces and shows a white reflecting spot on the object which helps orientating. Figure 2.19 shows Lambertian shading only on the left then with ambient and specular shading, and on the right ambient specular and cast shadows. For more details about shading models see [Fole90].

From knowledge of the real world, it is known how shadows are cast because the sun is above the observer. This is assumed also for visualizations and has to be considered. Shading and especially specular light give the observer information about fine details and depth.Figure 2.20. shows the same image twice, but the right image is turned 180 degrees, bumps are now perceived as hollows, because light is assumed to come from above.

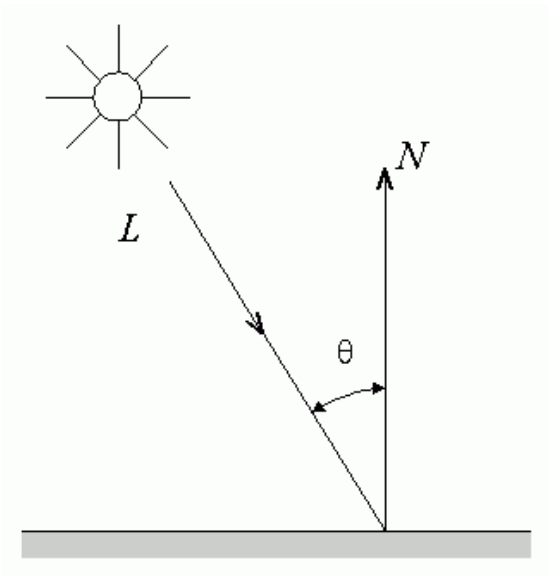


Figure 2.18: The Lambertian shading model.

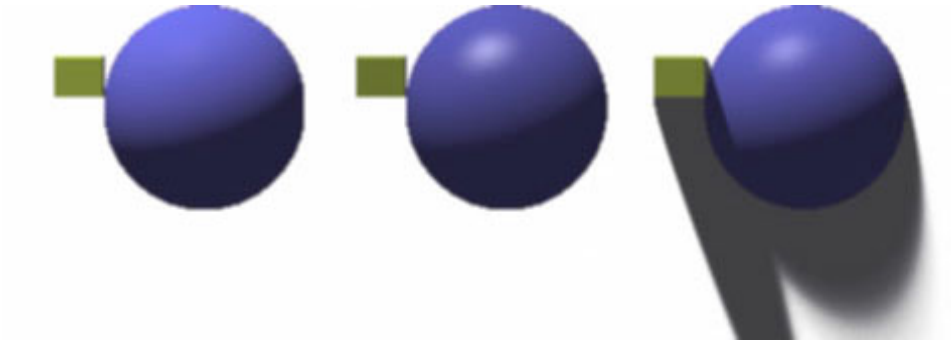


Figure 2.19: Lambertian shading only, ambient and specular, ambient and specular with shadows.

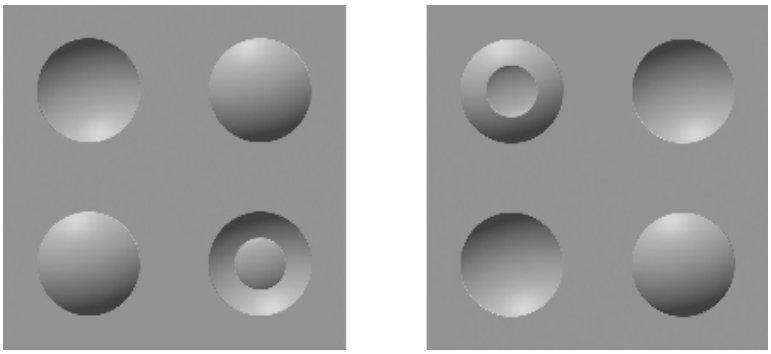


Figure 2.20: Bumps and hollows with specular light. Right image turned 180°. (Source [Ware00])

Chapter 3

Visual Perception and Visualization

"The eye...

the window of the soul,
is the principal means
by which the central sense
can most completely and
abundantly appreciate
the infinit works of nature."

Leonardo daVinci 1452-1519

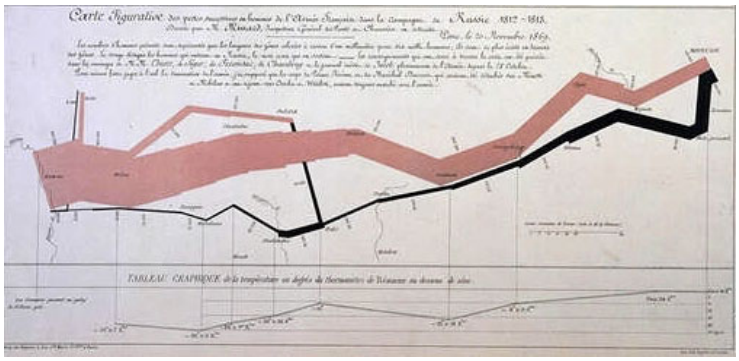


Figure 3.1: Napoleon's march on Moscow. Orignial Title: Carte figurative des pertes successives en hommes de l'armée française dans la campagne de Russie. (Source: [Mina12])

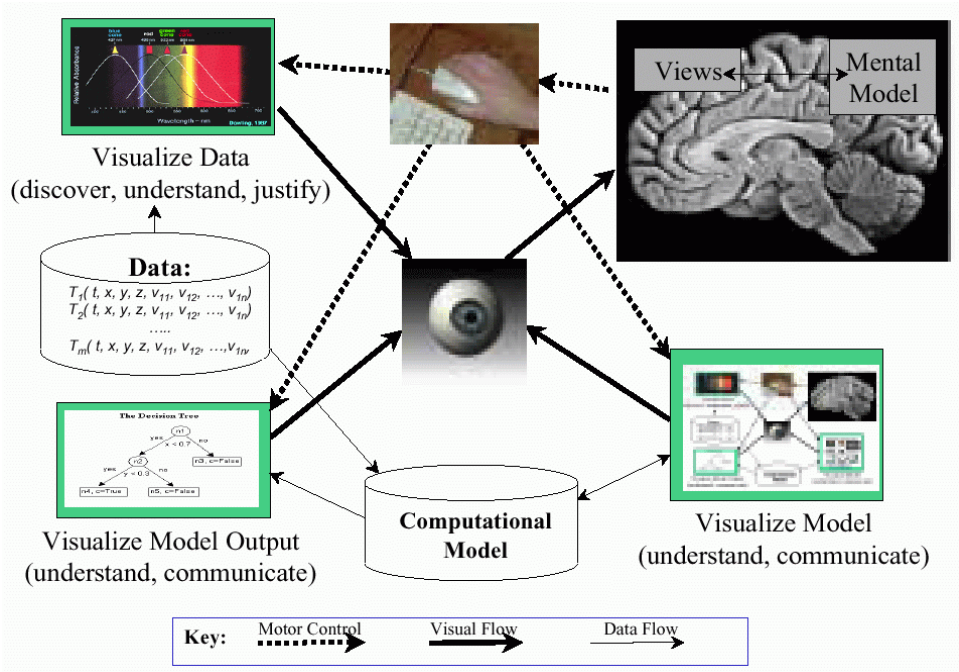


Figure 3.2: Diagram of the visualization process. (Source: [Crap00])

One of the greatest benefits of data visualization is the sheer quantity of information which can be rapidly interpreted if it is presented well. Figure 3.1. [Mina12] shows a visualization of Napoleon's march on Moscow. The thick, brown band starting at the left shows the troop strength of 422,000 soldiers at the beginning of the campaign in June 1812. The band width shows the approximate numbers of troops at various locations during the advance to Moscow. The retreat from Moscow is shown as a black band. The retreat band is also linked to a temperature scale and dates at the bottom of the chart. Instead of reading and comparing long lists and columns, all this data can be perceived at once [Tuft82].

The advantages of visualization can be summarized as follows:

- Huge amounts of data can be comprehended.
- Properties, even those that were not anticipated, emerge.
- Problems with data like errors or artifacts pop out immediately.
- Visualization helps to understand of large-scale and small-scale features of the data.
- Visualization facilitates hypothesis formation.

The process of visualization needs various experts such as a data seeker, a psychologist, and a graphic designer. See Figure 3.2. Data is gathered and its interpretation in the social and physical environment is determined. It has to be figured out which is the best way to visualize the data and then algorithms are selected to find the best graphical representation [Hump00].

3.1 Semiology

Visualization is made up of symbols and the meaning of a symbol is understood depending on convention and the social interaction.

$$\nabla(\oint(x,y,z))\Gamma\gamma\frac{\frac{a}{b}}{c}\arccos(\Psi)$$

Figure 3.3: A visual language.

Figure 3.4: Another visual language.

- *semantics*: is the relationship of signs and their meaning.
- *syntactics* (or syntax): the formal or structural relations between signs.
- *pragmatics*: the relation of signs to interpreters.

3.1.1 Semiotics of Graphics

The study of symbols and how they convey meaning is called semiotics. It is often claimed that visual languages are easy to learn and use, but they can be just as hard to learn as written language. Figure 3.3 shows for example the visualization of an equation which needs some experience to understand [Chan01]. Other symbols instead are immediately clear such as the symbol in Figure 3.4. A sign can be understood as a two-part model. The form of the sign, the *"signifier"*, and the concept it represents the *"signified"*.

The Saussurean diagram shown in Figure 3.5 indicates with arrows the relation between signifier and signified, this relation is called *"signification"*. The horizontal line in Figure 3.5 is called "the bar". Taking a linguistic example, the word "Tree" is a sign consisting of:

- A signifier: the word "Tree".
- A signified concept: If the interpreters speaks English they will think for example about a oak tree.

There cannot be a sign with a meaningless signifier or a formless signified. A sign must have both to be understood. One signifier can stand for different signified and therefore have a different sign. The word "open" for example can stand once for the concept of a "open shop" if the sign is placed at a store entrance or on top of a carton together with an arrow it means "open this end", but the word "open" on a push-button in an elevator means "open the door". The signifier "open" stands in each example for a different signified [Fior98].

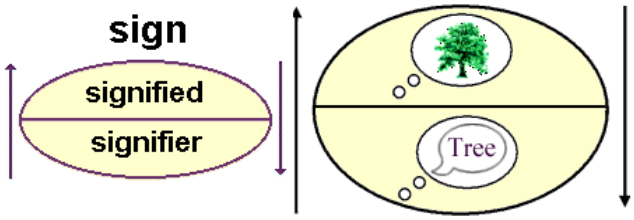


Figure 3.5: Saussurean diagram. The four Latin letters TREE are the signifier and signify the concept of a tree for persons speaking English. (Source: [Chan01])

3.1.2 Classification of Signs

According to Chandler [Chan01], signs can be classified as follows:

-  *Symbolic*: An arbitrary or conventional mapping between signifier and signified. The relationship must be learned: e.g. language in general, numbers, Morse code, traffic lights, national flags.
-  *Iconic*: The signifier is a resembling or an imitation of the signified, it has a recognizable looking or sensing (look, feel, taste, or smell): e.g. a portrait, a cartoon, a scale-model, metaphors, imitative gestures.
-  *Indexical*: The signifier and the signified are directly connected in a physically or causally way. Examples are footprints, smoke, directional signposts, a photograph, and indexical words ("that", "this", "here", "there").

3.1.3 Arbitrary Representations

Some basic properties of arbitrary codes can be stated in general terms. They are hard to learn, the graphical code of the alphabet and their rules of combinations must be laboriously learned and they are embedded in culture and applications. They are also easy to forget, if not overlearned such as written numbers arbitrary conventional information can by easily forgotten and finally they are also capable of rapid change. Arbitrary representations are formally very powerful. For example mathematicians have created hundreds of powerful graphical languages but something expressed in visual code does not mean that it is easy to understand.

3.1.4 Sensory Representations

Sensory representations are representations for which the meaning is perceived without additional training and certain signs are perceived immediately. To study representations of sensory types

different research methodologies are appropriate [Ware00]:

- *Psychophysics:* The physical measuring of human sensations like how fast light has to flicker to be perceived as steady is called psyhophysics. Psychophysical techniques are normally used for studies intended to reveal early sensory process.
- *Cognitive Psychology:* In cognitive psychology, the brain is modelled as a set of interlinked processing modules and the memory is divided into a short-term or working-memory and the long-term memory. Methods in cognitive psychology are for example the measuring of the reaction time.
- *Structural Analysis:* Interpreters are asked to assign numbers to subjective things like effectiveness or clearness. In this way similar to a formal experiment, numerical data for comparing representations is obtained.
- *Cross-Cultural Studies:* Symbols that will be used all over the world should be tested for cross cultural boundaries. For example, in Western cultures the color green means nature and health, but in Asiatic cultures it means death and ghost.

3.2 Gibson's Affordance Theory

James J. Gibson defined the term "ecological psychology" to emphasize his belief that more traditional psychologies of the "mind" or of "behavior" were too narrowly conceived. He assumed that observers perceive in order to operate on the environment and perception is designed for action.

"The environment not only serves as the surfaces that separate substances from the medium in which animals live, but also affords animals in terms of terrain, shelters, water, fire, objects, tools, animals, human displays, etc.; and there is not only information in light for the perception of the environment, but also information for the perception of what the environment affords. He proposed a radical hypothesis: the composition and layout of the surfaces in the environment constitute what they afford." [Norm99].

Gibson's affordance has the following properties: [Gibs79]

- Affordance is perceived directly.
- Affordances provided by the environment are what it offers, what it provides, and what it invites.
- The "values" and "meanings" of things in the environment can be directly perceived. "Values" and "meanings" are external to the perceiver.
- Affordances can only be measured in ecology, not in physics.
- An affordance is an invariant.
- Affordances are holistic. What is perceived when someone looks at objects are their affordances, not their dimensions and properties.

Consider the ambiguous door designs shown in Figure 3.6. A knob affords turning, but does it afford pushing or pulling? A horizontal bar affords pushing, but which side should be pushed? Figure 3.7 shows an example of good affordances in door designs. A flat panel affords pushing and the broadness indicates which side to push. A vertical handle affords grasping and pulling.

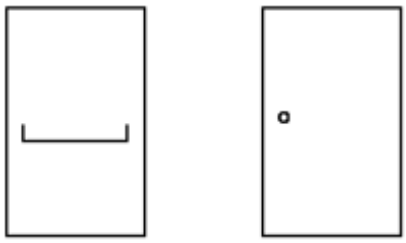


Figure 3.6: Ambiguous door design.

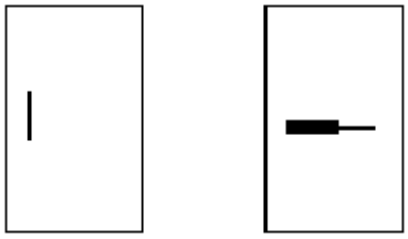


Figure 3.7: Clear door design.

3.3 Perceptual Processing

Perceptual processing can be modeled in two stages. In the first stage, visual information is processed in parallel by large arrays of neurons in the eye and in the primary visual cortex at the back of the brain. Individual neurons are selectively tuned to certain kinds of information, such as orientation of edges or the color of a patch of light. This processing includes the extraction of features, orientation, color, texture, and movement patterns and the transitory information in the iconic buffer

In the second stage, of sequential processing, visual attention and memory become important. To identify an object, visual characteristics are compared to properties of an object stored in memory. The task which the observer is performing also influences what is perceived. Sequential processing is slow, involving both working memory and long-term memory.

3.4 Types of Data

In information visualization the classification of data is important and this issue is closely related to the classification of knowledge. From the history of database design, the concept of entity, relationship, and attribute are well known [Maur96].

Entities are generally the objects of interest. Computers, humans, and cats can be entities. There are different kinds of relationships from the structure of entities. A chip is **part of** a computer, a cat is a **member of** animals. Relationships can be structural, physical, causal, or temporal [Stro91].

Attribute types of entities and relationships [UBCO99]:

- *Nominal:* Numbers are used to represent identity. For example the phone number 6151772 stands for one unique telephone but this number is not more or less then a different phone number like 5893899. As another example of the nominal type can be mentioned the numbers of racers, they are only to distinguish the racer from others.

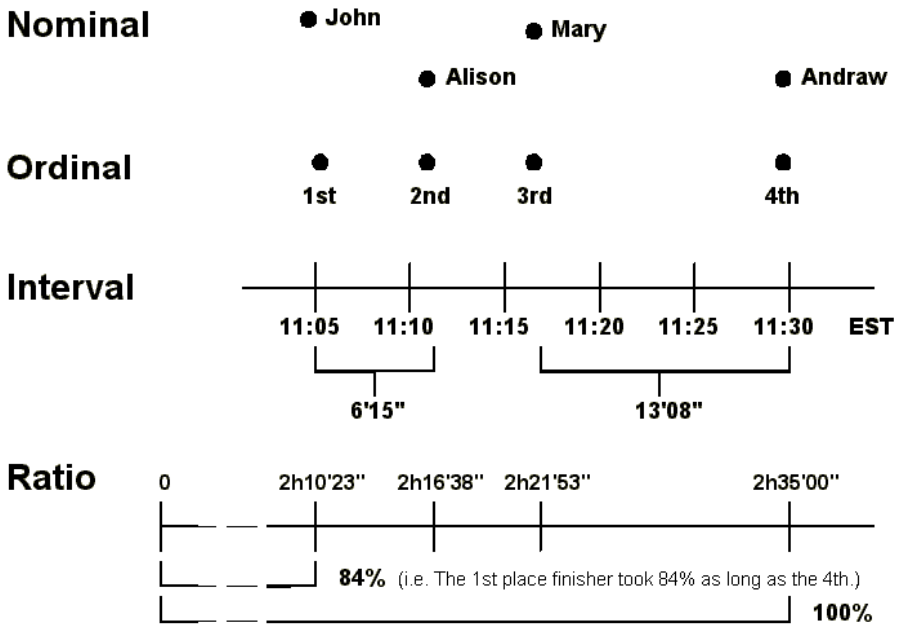


Figure 3.8: Different types of attributes. (Source: [UBCO99])

- *Ordinal:* An example of ordinal attributes are the results in a race, i.e. 1st place, 2nd place, 3rd place, etc.
- *Interval:* The difference between two numbers, e.g. on a temperature scale represent an interval, it can be said that 30°C is 10 degrees warmer than 20°C.
- *Ratio:* The ratio scale has an absolute zero point so the difference between two numbers can be measured absolutely. On a weight scale for example it can be found that 100kg is twice the weight of 50kg.

The different types of attributes are illustrated in Figure 3.8.

3.5 Brightness, Lightness, and Contrast

How bright is a patch of light? What is white, what is black, and what is dark gray? The answers to these simple questions is more complex than it seems and lead to many of the fundamental mechanisms of perception. The Difference of Gaussians model (DOG function) is a widely used mathematical description of the concentric receptive field. In this model, the firing rate of the cell is the difference between two Gaussians. One of the Gaussians represents the center the other the represents the surround. The DOG receptive field can be used to explain a variety of brightness contrast effects.

In the Hermann-Hering grid, shown in Figure 3.9. black spots appear at the intersections of bright lines. These illusions are seen because there is more inhibition at the points between two squares, and hence they seem brighter than the points at the intersections. Mach bands appear where there is an abrupt change in the first derivative of brightness model. In Figure 3.10 the red line shows the perceived brightness and the blue line the actual brightness.

In the Craik-Cornsweet Illusion, brightness is not a simple function of intensity (see Figure 3.11). This illusion lets the left side of the pattern looks darker than the right, both sides are physically identical, occluding the central edge with a thin object proves this. When the edge

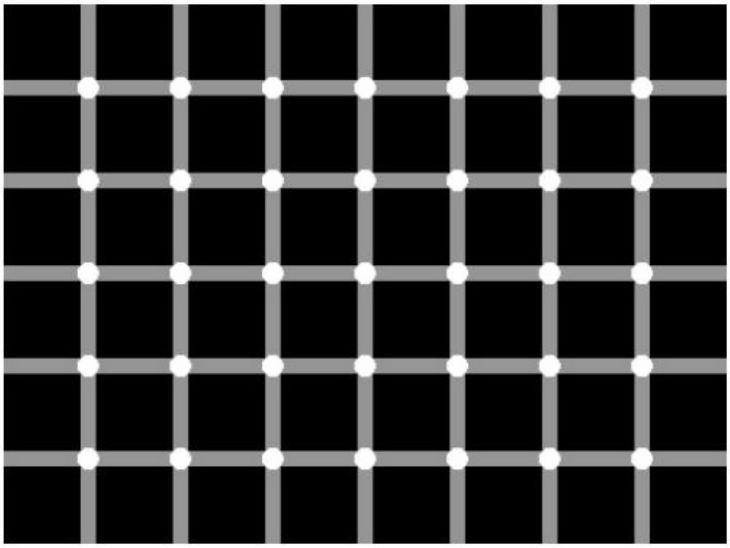


Figure 3.9: The Hermann-Hering Grid.

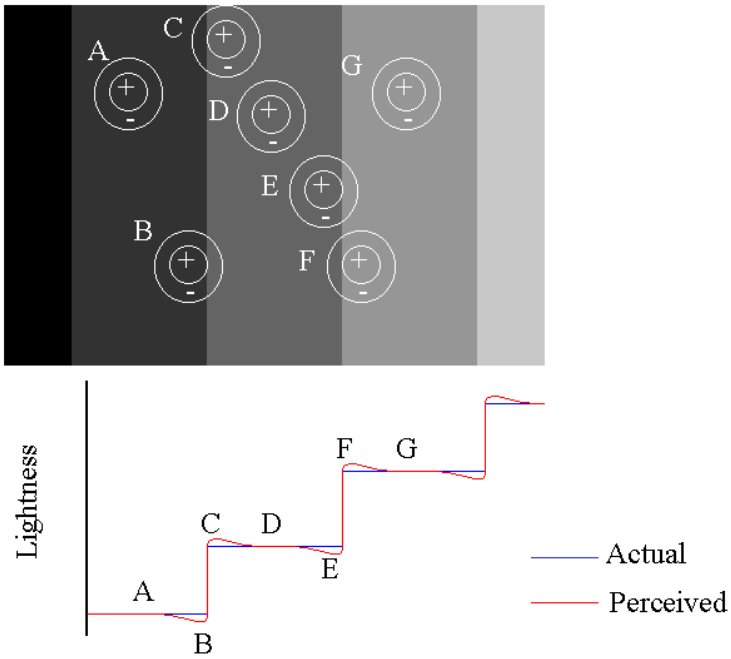


Figure 3.10: The Mach band illusion. Perceived and actual brightness.

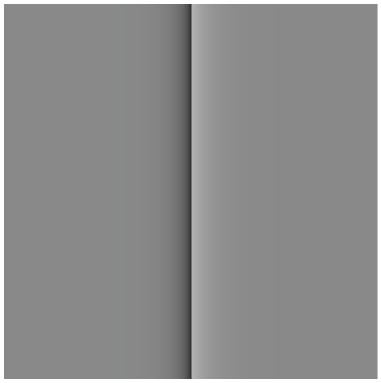


Figure 3.11: The Craik-Cornsweet illusion.

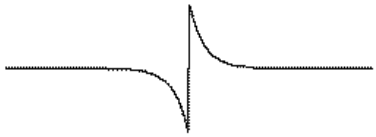


Figure 3.12: The luminance profile of Figure 3.11.

is covered, the sides of the pattern have the same perceived lightness. Figure 3.12 shows the luminance profile of the pattern.

3.6 Color

Color is more than a visible electromagnetic wave, it also has psychological aspects. There exists relationships between colors and emotional sensations, e.g. colors like reds, yellows, and browns are perceived as "warm", whereas blues, greens, and grays are described "cold".

Depending on the social and cultural environment colors represent certain feelings like green for nature, red for aggression or love and so on; this has to be considered in the selection of colors for visualizations. One method to present color is the colorwheel as seen in Figure 3.13. This Figure shows also the names for the most famous colors, but color names are not universal since in some languages there are no separation between orange and yellow. Despite of some exceptions it can be said that comparing different cultures results that there are certain patterns. Figure 3.14 shows that all languages have designations for black and white followed by red, and yellow and green or green and yellow. Blue is the fifth color named, and brown is the sixth thereafter is no particular sequence [Ware00]. One of the most successful theories of color vision, trichromacy theory, was first proposed around 1801 by Thomas Young, an English physician, and refined about 50 years later by the German scientist Hermann von Helmholtz. It postulates that there are three distinct color receptors in the retina (see Figure 3.15). The three curves in Figure 3.15 shows the normalized cone sensitivity functions. This is the response of an average human eye to various amounts of ambient light. The shift in sensitivity occurs because two types of photoreceptors absorb the light at different wavelengths.

The curve on the right shows the eye's response under normal lighting conditions and this is called the photopic response. This curve peaks at 555 nanometers which means that under normal lighting conditions, the eye is most sensitive to a greenish-yellow color. With ambient lighting conditions as in an office the eye's response shifts the curve to the left. This is presented as the heavier curve in middle of Figure 3.15, it peaks at 550 nanometers, which means the eye

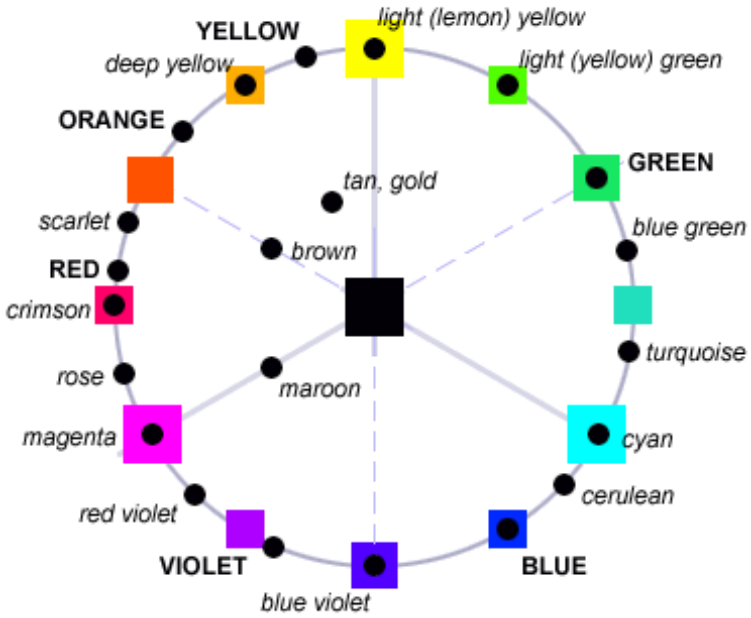


Figure 3.13: The color wheel . Adapted from http://www.arce.ukans.edu .

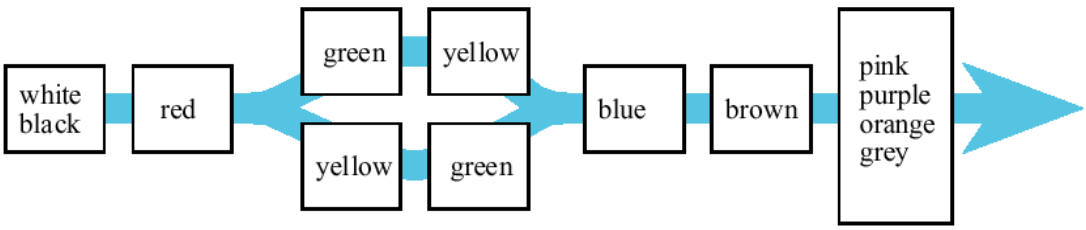


Figure 3.14: Color Names in Languages. (Source: [Ware00])

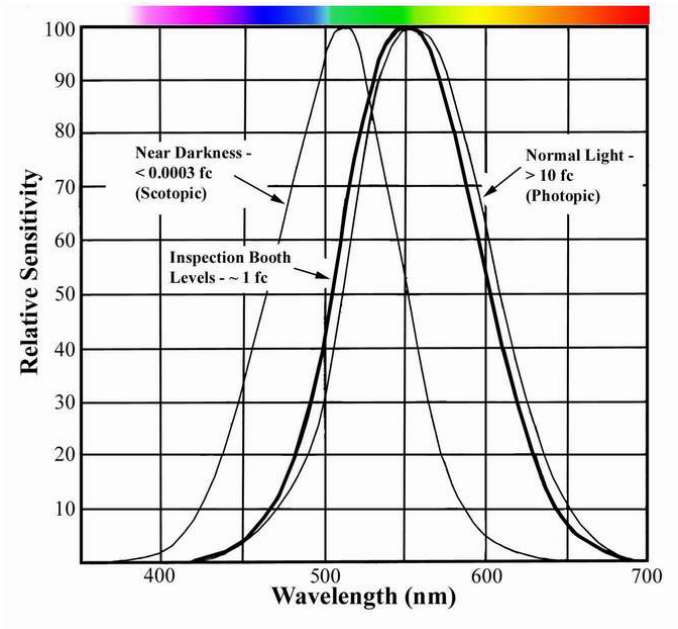


Figure 3.15: The relative sensitivity of the color receptors in the retina under various lighting conditions. (Source: [YORK01])

Figure 3.16: The additive primaries: red, green, and blue.

is most sensitive to yellowish-green color at this light level.

Since only three different receptors are involved in color vision, it is possible to match a color with only three primaries. This leads to color measurement. Color spaces, or color order systems, are methods of organizing the set of possible human color perceptions in a systematic way. With these methods, colors or their attributes are expressed numerically. A color space is a three-dimensional geometric representation of the colors that can be produced using a certain color model. Figure 3.18 shows two different models of color-spaces.

The additive primaries, (red, green, and blue) when projected together as beams of colored light, as on a computer monitor or television screen, will mix or overlap to produce the colors as shown in Figure 3.16. The subtractive primaries (cyan, magenta, and yellow), are used to create color separations for photography and printing. In subtractive color mixing, pigments such as ink or paint absorb or subtract all the colors of the spectrum *except* the color that the pigment reflects to the eye, as shown in Figure 3.17.

In the color opponent process model shown in 3.19, cone signals are transformed into black-white (**luminance**), red-green, and yellow-blue channels (**chromatic**). From the perspective of data visualization, the different properties of the color channels have important implications for the use of color. The most significant differences are between the two chromatic channels and the luminance channel. See Table 3.1.

Figure 3.17: The subtractive primaries; cyan, magenta, and yellow.

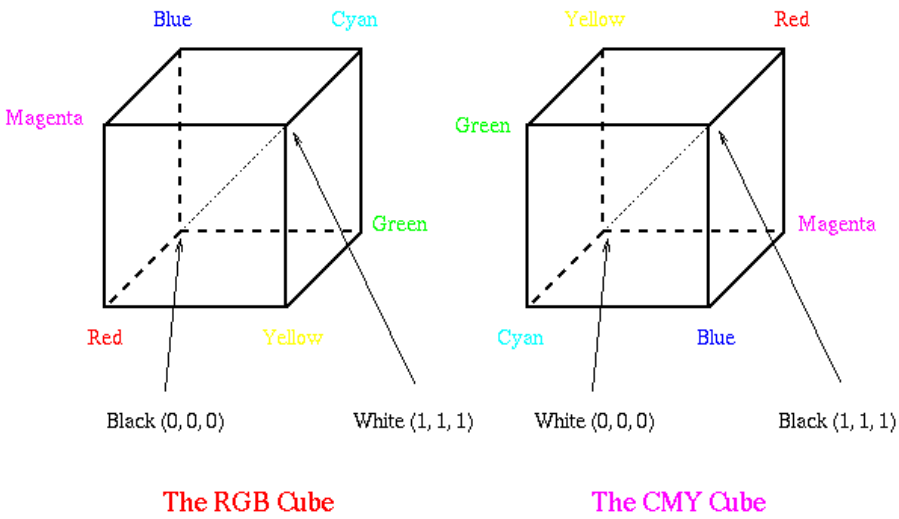


Figure 3.18: The RGB and CMY color cubes.

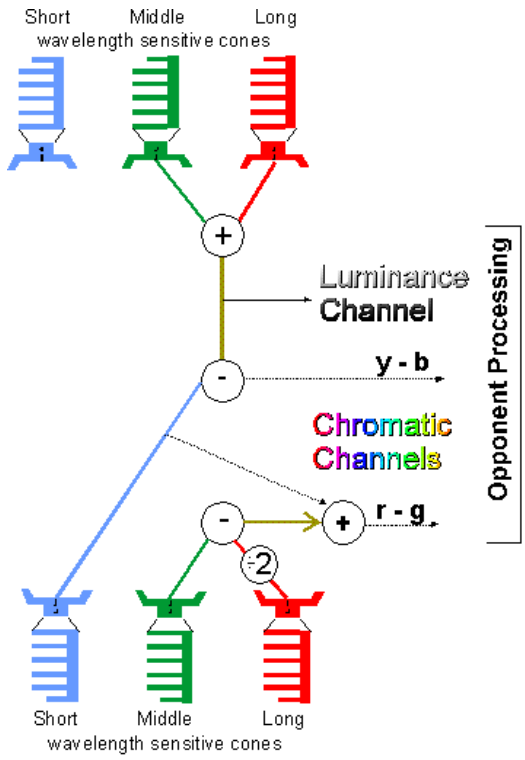


Figure 3.19: Color vision model. (Source: [YORK01])

Luminance Channel	Chromatic Channels
Detail	Surfaces of things
Form	Labels
Shading	Categories (about 6 – 10)
Motion	Red, green, yellow and
Stereo	blue are special hues

Table 3.1 The differences between the luminance and the chromatic channels. Source: [Ware00]

Spatial sensitivity is very important since the red-green and yellow-blue chromatic channels carry only one third of the amount of detail carried by the black-white channel. In Figure 3.20, in the areas where there is only chromatic difference, the text becomes hard to read.

Figure 3.21 shows a color contrast illusion. Stimulation of neighboring parts of the visual system influence one another. The tiles in each palette are actually identical, but the colors of the tiles may appear very different when they are displayed on the different backgrounds.

3.7 Color in Visualization

Color is often extremely effective for labeling when a distinct color code for objects is given. It has to be remembered that humans can see only about 24 different shades of gray, whereas more than 3000 different colors can be perceived. A typical example is shown in Figure 3.22 where the original gray magnetic resonance image pseudo-colored depending on the medical problem [Wilt99]. Pseudo-coloring is a technique where colors are mapped to a certain value not depending on the true color.

Figure 3.20: Yellow text on a blue gradient.

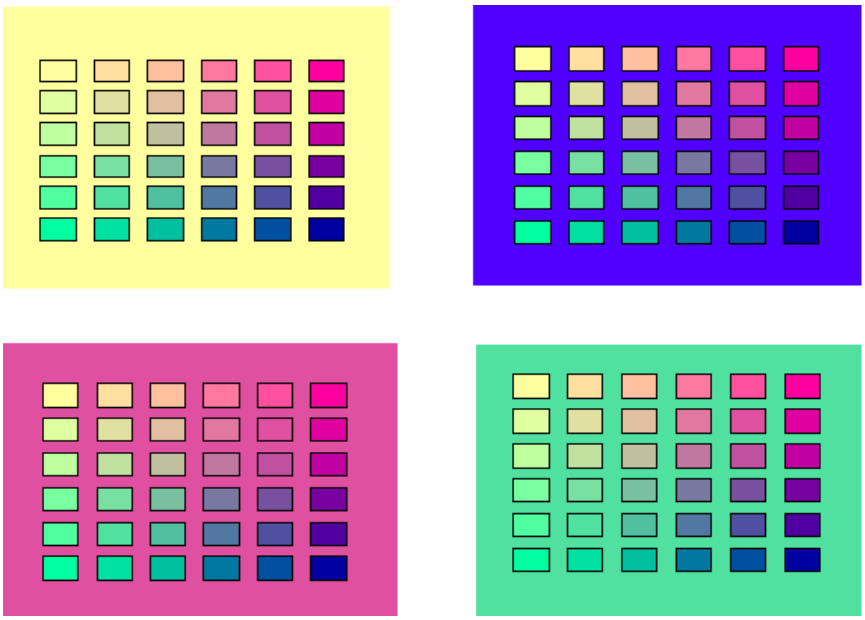


Figure 3.21: Color palettes on different backgrounds.

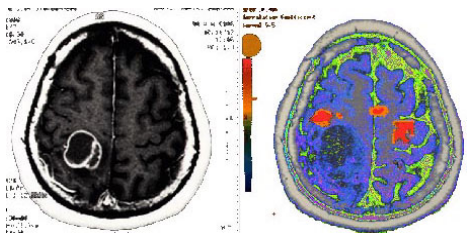


Figure 3.22: Magnetic resonance (MR) image without and with pseudocoloring.

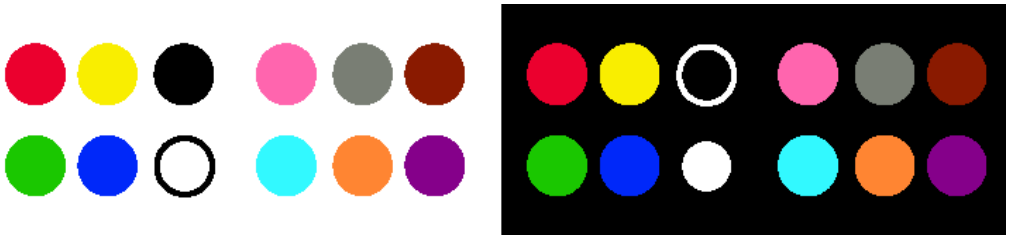


Figure 3.23: Breaking isoluminance with small borders.

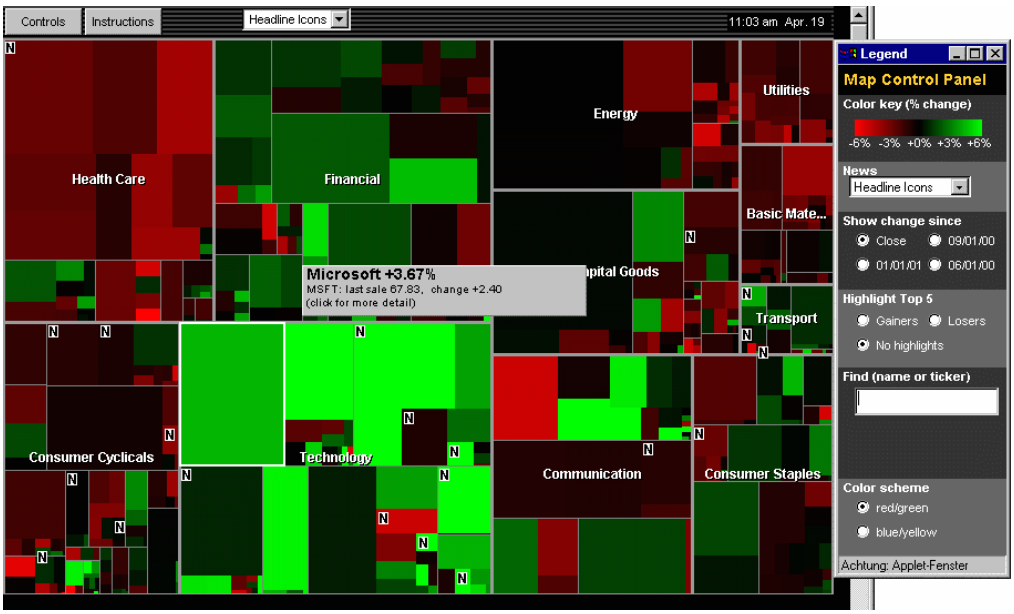


Figure 3.24: Market map with red-green color coding. (Courtesy of Smartmoney www.smartmoney.com/marketmap)

Chromatic coding can often be employed in a way that only minimally interferes with data presented on the luminance channel. Perceptual factors to be considered in choosing a set of color labels, include [Ware00]:

- *Distinctness:* The difference between two colors which are placed together should have a certain degree of perceived difference.
- *Unique hues:* Red, green, yellow and blue, as well as black and white, are special because they are perceived as opposites.
- *Contrast with background*: The selected color-coding should always differ from the background. A small black or white border can achieve this effect, as shown in Figure 3.23.
- *Color blindness:* Color blindness has to be considered in some applications, but unfortunately this reduces the available design choices. An example of a visualization application catering for red-green color blindness can be seen in Figure 3.24 and Figure 3.25.
- *Number:* Only a small number of different colors can be perceived rapidly.

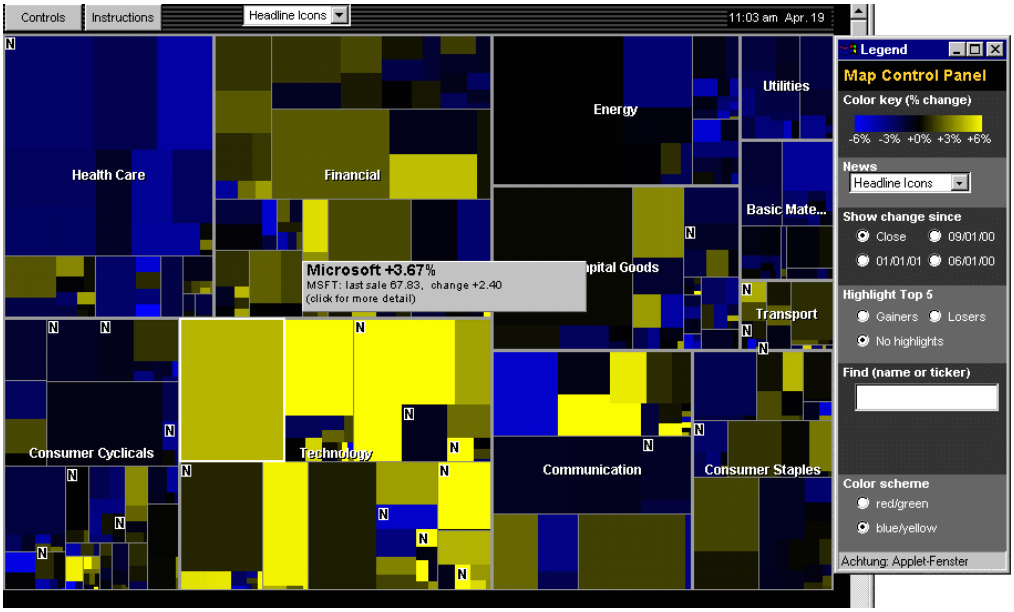


Figure 3.25: Market map adapted for color blind people with blue-yellow color coding. (Courtesy of Smartmoney www.smartmoney.com/marketmap)

Figure 3.26: Color sequences for pseudo-coloring shown on a brain visualization.

- *Field Size:* Small areas are of color should be avoided. Any small areas should be at least half a degree of visual angle and have high saturation. Large fields of color should be instead of low saturation.
- *Conventions:* Color-coding conventions such as green = go, red = hot, and blue = cold should be considered, but one should keep in mind that there are cultural differences.

The most common color scheme used in scientific visualization is a color scheme that approximates the physical spectrum. In pseudo-coloring, two issues are generally important: perceiving the shapes of features and classification on the basis of color. See Figure 3.26 for an example of two different color sequences.

3.8 Attention and Pre-Attention

Visual search provides one of the great benefits of visualization. To detect a single dark pixel in a 500 x 500 array of white pixels users need normally less than a second. This section is about searching for, and rapidly identifying, information.

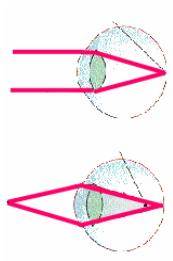


Figure 3.27: Accommodation, the object remains focused.

The eye is in constant motion seeking for information. In reading this page the eye makes movements and moves the focus of interest this movements are called saccadic movements. There are three important types of eye movement.

- *Saccadic eye movements* are very fast, ballistic, eye movements, separated by fixation periods during which the eyes are relatively still. This movements are time optimal: the eyes move as fast as physiologically possible. Speeds of up to about 1000 degrees per second are possible. Saccades are ballistic in the sense that, once initiated, they cannot be stopped or modified. The duration of the neural impulse determines the size of the movement.
- *Pursuit eye movements* are slow eye movements, continuous, and conjugate and occur when a fixed object is moved slowly.
- *Convergent eye movements* can be either saccadic or smooth, when an object moves toward the observer the eyes converge.

One of the important capabilities of the human eye is its ability to change its focal length by using the ciliary muscles to change the curvature of the flexible lens. This allows the eye to adjust so that objects at various distances from the eye can be focused and hence seen clearly, as shown in Figure 3.27. This ability to adjust the focus is called *accommodation*. As the eye ages, it gradually loses some of its ability to accommodate and for that more older people than young people require corrective lenses.

Certain simple shapes or colors "pop-out" from their surroundings and are easier to identify after a brief exposure. If processing is pre-attentive, the time taken to find the target is independent of the number of distractors [Luck98]. Figure 3.28 illustrates some features which can be processed pre-attentively, such as orientation, shape, enclosure.

Features that are pre-attentively processed can be organized into a number of categories based on form, color, motion and spatial position [Bonf99].

- *Form:* line orientation, line length, line width, line collinearity, size, curvature, spatial grouping, added marks, and numerosity.
- *Color:* Hue and intensity.
- *Motion:* Flicker and direction of motion.
- *Spatial Position:* 2D position, stereoscopic depth, and convex/concave shape from shading.

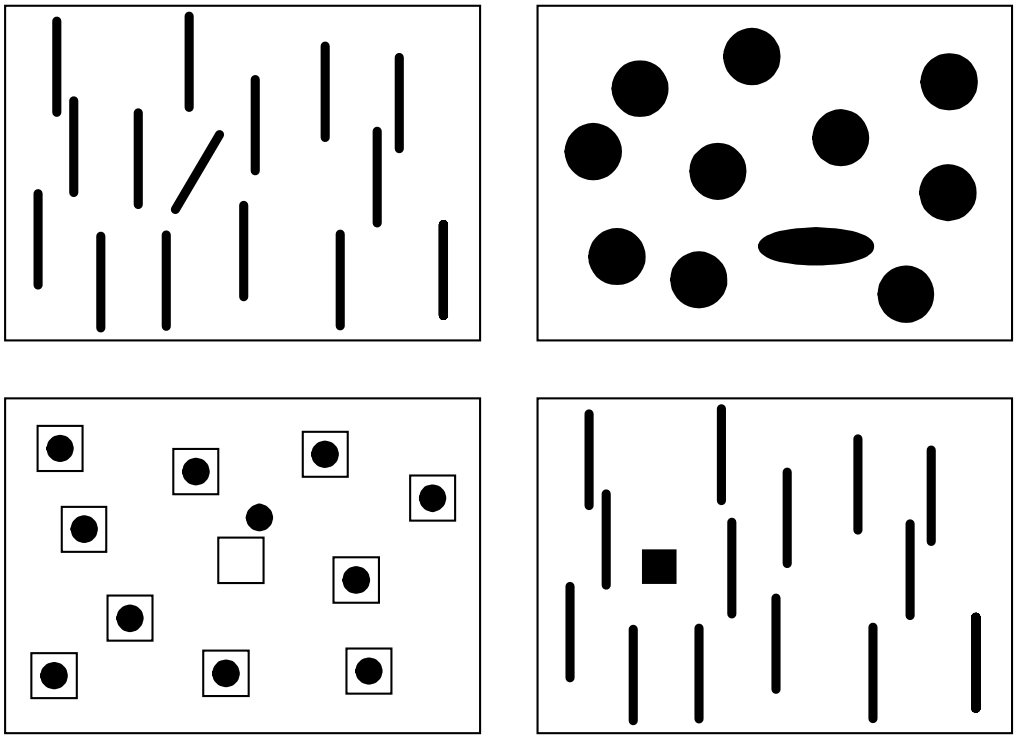


Figure 3.28: Examples of pre-attentively features. From top left to bottom right: orientation, shape, enclosure, and shape.

3.9 Glyphs in Visualization

One way of representing multivariate discrete data is by using a glyph. A glyph is a single graphical object that represents a data object. To create a glyph, multiple data attributes are mapped in a systematic way to show different aspects of the appearance of the graphical object. All the results discussed previously, related to pre-attentive detection of size orientation and color coding of data, apply to the design of glyphs. Figure 3.29 shows a Chernoff face as an example of a glyph, each data value in this multidimensional data element controls an individual facial characteristic. Examples of these characteristics include the nose, eyes, eyebrows, mouth, and jowls [Cher73].

Figure 3.29: Example of pre-attentive processing: A Chernoff face.

3.10 Gestalt Psychology

The study of form must involve the study of perception of form. The Gestalt law says that the whole of anything is greater than its parts. A great amount of work on this subject has been inspired by the theories of the Gestalt psychologists Max Wertheimer, Wolfgang Köhler, and Kurt Koffka.

The attributes of the whole of anything are not deducible from analysis of the parts in isolation. The word Gestalt is used in modern German to mean the way a thing has been gestellt; i.e., "placed," or "put together." There is no exact equivalent in English. In psychology the word is often rendered "pattern" or "configuration." [Holz00c].

The main Gestalt laws:

- *Law of Proximity:* Elements that are closer together will be perceived as a one coherent object. On the left in Figure 3.30, there appears to be horizontal rows, while on the right, the grouping appears to be columns since here the distance between the faces has changed.
- *Law of Similarity:* Similar looking elements will be perceived as part of the same form. There seems to be a triangle in the square formed by the little triangle faces. See Figure 3.31.
- *Law of Good Continuation:* There is a tendency to continue contours whenever the elements of the pattern establish an implied direction. Humans tend to draw a good continuous line as shown in Figure 3.32. The two blue unhappy faces will not be perceived as continuation since they mean a abrupt change of direction.
- *Law of Prägnanz:* A stimulus will be organized into a good a figure if possible. Good means in this context: symmetrical, simple, and regular. The Figure 3.33 appears to the eye as a square overlapping triangle, not a combination of several complicated shapes.
- *Law of Closure:* Humans tend to enclose a space by completing a contour and ignoring gaps, Figure 3.34 lets the observers perceive two triangles.

Figure 3.30: Law of Proximity. (Source: Jenny Fultz, Anderson University)

Figure 3.31: Law of Similarity. (Source: Jenny Fultz, Anderson University)

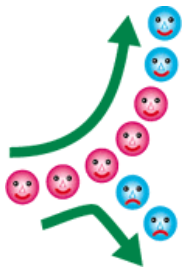


Figure 3.32: Law of Good Continuation. (Source: Jenny Fultz, Anderson University)

Figure 3.33: Law of Prägnanz.

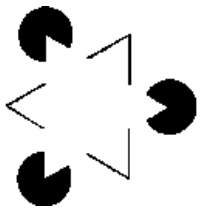


Figure 3.34: Law of Closure.

Chapter 4

Data and Information Visualization Techniques

The visual data and information exploration process can be viewed as a hypothesis generation process, whereby visualizations of the data allow users to gain insight into the data and come up with new hypotheses. Verification of the hypotheses can also be accomplished via visual data exploration, as well as through automatic techniques derived from statistics and machine learning.

4.1 Data and Information Visualization

Different goals of data and information visualization adapted from [Keim00]:

- *Explorative Analysis:*

Starting point:	Data without hypotheses about the data
Process:	Interactive, usually undirected search for structures, trends, etc.
Result:	Visualization of the data, which provides hypotheses about the data

- *Confirmative Analysis:*

Starting point:	Hypotheses about the data
Process:	Goal-oriented examination of the hypotheses
Result:	Visualization of the data, which allows the confirmation or rejection of the hypotheses

- *Presentation:*

Starting point:	Facts to be presented are fixed a priori
Process:	Choice of an appropriate presentation technique
Result:	High-quality visualization of the data presenting the facts

Figure 4.1 shows the comparison of the abilities of humans and computers. Creativity, general knowledge, and perception are significant human abilities. So, how should large data sets be prepared to be immediately understood by humans? The pioneering work of Tufte [Tuft82] and Bertin [Bert81] focuses on visualization of data with inherent 2D-/3D-semantics general rules, for layout, color composition, and attribute mapping.

The visualization techniques were classified by [Keim00] into one or more of the following categories, also identified in Figure 4.2:

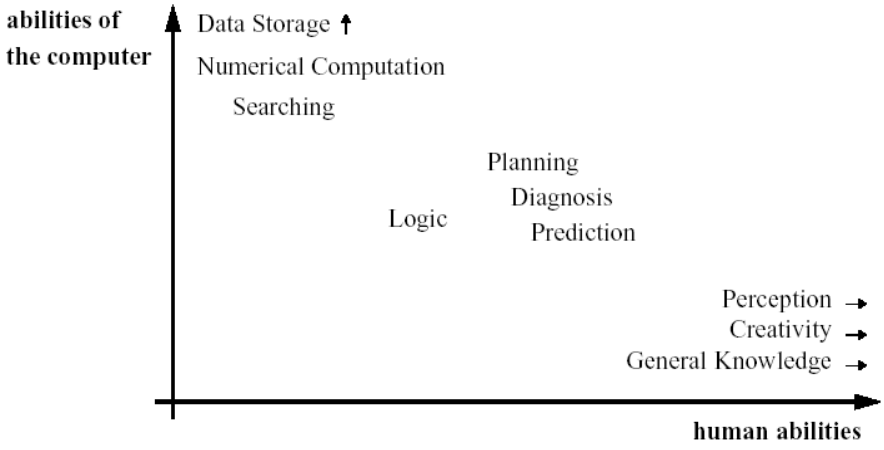


Figure 4.1: Comparison of the abilities of humans and computers.

- 2D/3D displays such as x-y plots and landscapes for visualizing the data.
- Geometric techniques; displays using geometric transformations and projections.
- Icon-based displays; Each icon represents a data item and the dimension values as features of the icons. For example stick figures and Chernoff faces.
- Pixel-oriented techniques; dense pixel displays that visualize each dimension value as a color pixel and group the pixels belonging to each dimension into an adjacent area.
- Graph-based techniques; visualization of large graphs using techniques to convey the meaning of the graph clearly and quickly.
- Stacked displays that visualize the data partitioned hierarchically. The most important dimensions have to correspond to the first levels of the hierarchy, and with the other metadata as dimensions a hierarchy is build up.
- Hybrid techniques; arbitrary combinations of the above

Visual data exploration, also known as the **"information seeking mantra"**, usually follows a three-step process [Shne96]:

"overview, zoom and filter, and details-on-demand"

In the overview step, the user identifies interesting patterns, focusing on one or more of them. To analyze the patterns, the user drills down to access details of the data. Visualization technology may be used for all three steps, presenting an overview of the data and allowing the user to identify interesting subsets. In analyzing the patterns, it is important to maintain the overview visualization while focusing on the subset using another visualization technique. An alternative is to distort the overview visualization in order to focus on the interesting subsets. It has to be noted that visualization technology provides not only the base visualization techniques for all three steps, but also bridges the gaps between the steps.

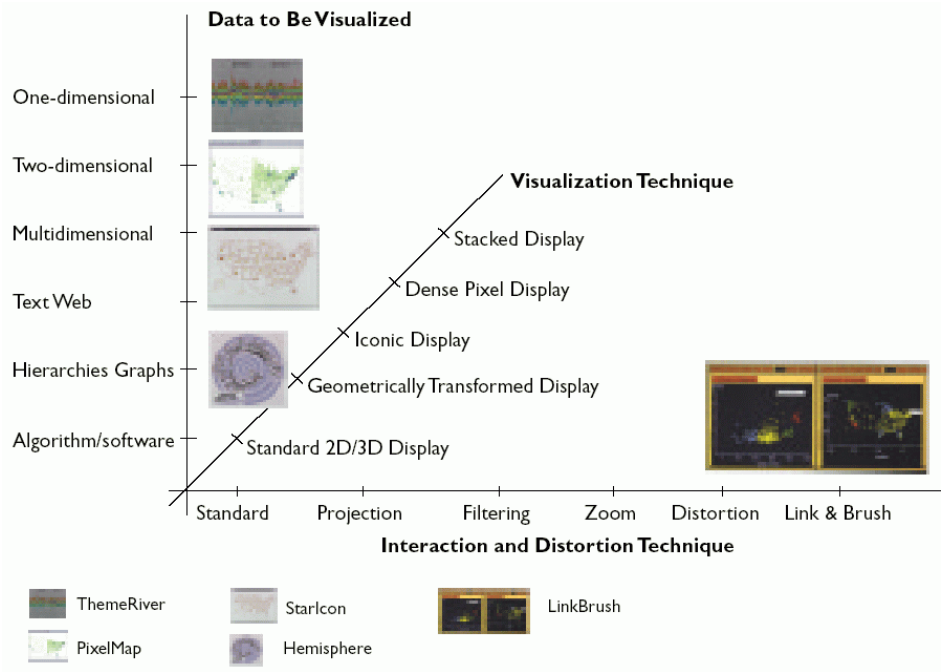


Figure 4.2: Classification of visual data exploration techniques. (Source: [Keim00])

4.2 General Principles of Visualization

The techniques described in this section were classified by Young [Youn96] according to their mappings from the data domain to the visualization space. These techniques use some aspect, property or value of the data items to produce a mapping to objects within the visualization, they range from surface plots, cityscapes, and Benediktine space, to the room metaphor.

4.2.1 Surface Plots

Similar to the topography of a landscape are surface plots. The surface plot uses three dimensional metadata with a regular structure. This regular metadata is then be mapped on the X and the Z axis, whereas the variable data is mapped to the Y axis, which is the height of the surface. With these mappings the relationship of the three variables is seen as a continuous surface and it makes it easy for users to identify important features of the data. Surface plots are very common and are available in mathematics packages but also in office applications. The principle of the surface plot can be seen in Figure 4.3.

4.2.2 Cityscapes

The simplest cityscape looks look like a extension of two dimensional bar charts, where the bars are placed in a horizontal plane. In the cityscape visualization information is encoded in three dimensional glyphs which appear like buildings, as shown in Figure 4.4. Cityscapes represent hierarchical information with the height, color and eventually architecture styles. Network information is presented as lines or to help the users cognitive model, as streets.

Observers of a cityscape can perceive the data classified into districts and neighborhoods connected with streets as in real towns.

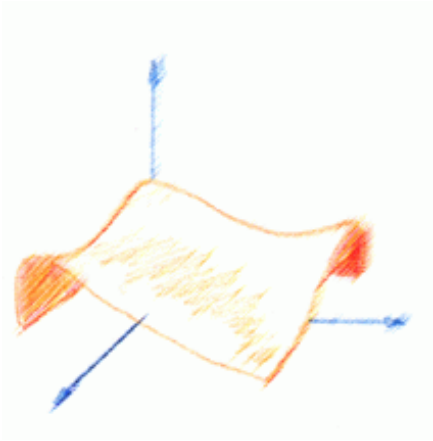


Figure 4.3: Surface plot principle.

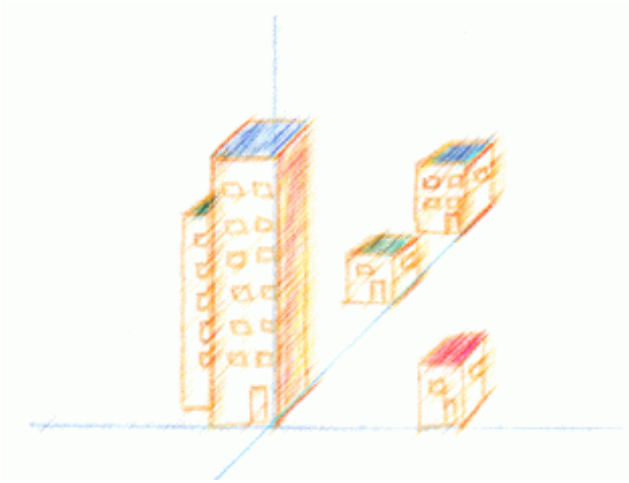


Figure 4.4: Cityscape principle.

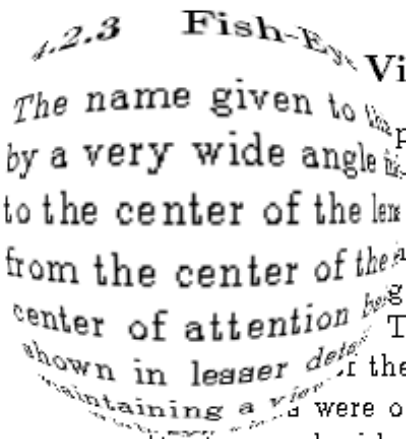


Figure 4.5: Fisheye Effect shown with text.

4.2.3 Fish-Eye Views

In his paper from 1982 Furnas [Furn86] proposed fish-eye views to keep information in focus and displays peripheral information in reduced detail. The simplest principle is shown in Figure 4.5. It is similar to the fish-lens effect of a optical camera and this gave also the name to this kind of visualization. Furnas defined a *degree-of-interest* (DOI) function, so parts of the information space are scaled or even hidden depending on their distance to the current focus of interest, the *focus point* (FP), to force a user-centered perspective. Fish-eye views can be used for example with large graphs where only the focus point is shown in detail, whereas only closely related objects are shown already smaller and therefor more objects can be visualized. This allows the viewer to concentrate on the interesting nodes while still maintaining a picture of these nodes positions within the whole structure.

4.2.4 Benediktine space

The name Benediktine space comes from Michael Benedikt, he defined a structure of Cyberspace where attributes are mapped to *extrinsic* and *intrinsic* spatial dimensions [Bene91]. Metadata that is mapped to the X, Y, and Z coordinates is called extrinsic mapping, on this position in the three dimensional space is drawn a glyph. This glyph has the intrinsic mappings on its size, shape, texture, etc. Benedikt stated a principle rule for this visualization to avoid occlusion:

"Two non-identical objects having the same extrinsic dimensions and dimension values, whether at the same time, or including time as an extrinsic dimension from the outset, is forbidden, no matter what other comparisons may be made between their intrinsic dimensions and values."

The FAEJ3D uses the Benediktine approach to visualize the attributes of the files, for example as extrinsic dimensions can be used the name, extension, and date of the file and the size can be mapped as intrinsic dimension to the glyph size.

4.2.5 Glyphs

Glyps are a single graphical objects that represent the data in form and/or in color. Glyphs can have multivariate discrete data and this data is mapped as attribute in a systematic way to the glyph. Famous examples are Chernoff faces, Starstruk. The FAEJ3D uses cones as glyphs with the data mapped to the height and the radius of the cones as intrinsic dimensions as described before.

4.2.6 Graphs

Graphs like a tree, are one possibility to visualize data and its common properties by placing the data in a hierarchically form in two dimensional or three dimensional space with nodes and arcs. One problem of graphs is that depending on the mapping algorithm the nodes can become cluttered and so hard to distinguish from each other. A method of organizing the distance between the nodes automatically are self-organizing graphs.

Self-organizing graphs use a special layout technique used in automatically laying out graphs. These techniques involve a function which attempts to perform a suitable layout on a given graph. One technique applies a physical model to the arcs and nodes and inserting a new node makes the system equations unstable and the graph response is a new spacial arrangement until the system is in a stable equilibrium. With such a physical model it can happen that after inserting a new node that the whole graph is changed and the users loose their cognitive model of the data.

4.2.7 Perspective

Using perspective in visualizations has the advantage that the center of interest is clearly focused and related data is closely, whereas other data is shown smaller because of the perspective distortion. The perspective can be realized by using a surface with a fixed viewpoint or by mapping the data onto a sphere or other geometric objects.

4.2.8 Rooms

Users are used to the three dimensional world and things like the printer are placed in a certain room. The 3D-rooms visualization is a metaphor for the real world and extends the normal desktop to a third dimension. The rooms are structured depending on their function like a internet room, an office, a multimedia room, etc. and so users can build easily a cognitive model to find the item they search.

Within each room their will be a variety of information sources depending on the type of task allocated to a particular room. To represent for example documents or applications a 3D object can be used, like a calculator to start the systems calculator program. The walls of the room can be used in a more conventional 2D manner, effectively using the walls as 2D displays for information, e.g. a calender.

The 3D-Rooms are connected like in the real world by doors and leaving one room again leads to the previous room. As a navigational aid for these structures an overview or floor plan of the rooms is provided, including some indication to the content or tasks of the individual rooms. This allows the user "teletransportation" between unrelated rooms or tasks without going through several other rooms like in the real world. In some implementations the users have as an useful addition to the rooms metaphor a notion of pockets which allows them to carry information or objects between rooms or keep important items close at hand. An example application is Win3D from Clockwise3D which can be used as complete substitution for the Windows desktop [CLOW99] since it includes all existing applications into the 3D-rooms.

4.3 Selected Visualization Systems

This section describes a number of information visualization systems which make use of the principles described in the previous section.

4.3.1 Cone Trees

Xerox PARC proposed 1991 the cone trees as a three-dimensional extension to two dimensional tree structures [Card91]. Cone trees are part of the Information Visualizer and allow interactive visualization of hierarchical structures. The construction of cone trees starts with the root node

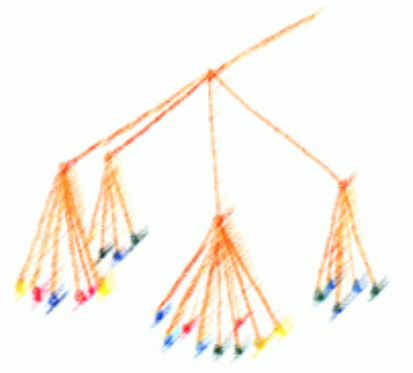


Figure 4.6: Cone tree principle.

near the top of the display. This root is the apex of a translucent cone. Starting from this root, child nodes are placed in equal distances along the base of the cone. To visualize all levels the process is repeated for every node in the hierarchy with a reduced base diameter for the cones. The principle is shown in Figure 4.6.

After the creation of the cone tree the users can bring the items of interest to the front by clicking on them, this results in a rotation of the cone tree which has to be done smoothly to maintain the viewers cognitive model of the structure. Cam trees are identical to cone trees except they grow horizontally as opposed to vertically.

4.3.2 Perspective Walls

The perspective wall uses a distortion technique on linear structured information similar to fish-eyes, so that details and context are integrated [Mack91]. Perspective walls maps a two dimensional layout onto a three dimensional wall as shown in Figure 4.7. The center can be seen by the users with the data mapped as colored rectangles and a time line on the bottom. Data closely related to the point of interest is mapped on the left and right side of the center where the wall is distorted. The wall can be animated by the users to bring other objects to the front where no perspective distortion occurs.

Perspective walls from Xerox-PARC offer two strategies to view large amounts of metadata, the first one is the described layout technique, and the second one is the time strategy. Since sometimes the layout can get overloaded and the users can not find detail information, they can now break up this view into a number of separate views, which can be displayed anytime. This allows to jump between different views but suffers by that it is easy to get lost within the data. Perspective wall resolves this problems by extending the time strategy with contextual cues. A single selection on the wall can be extended to be viewed in detail with adjacent sections being folded back on either side of the view to give cues to the position of the current section. Changing the detail views results in a smooth sliding of the wall to maintain the users cognitive model.

4.3.3 Sphere Visualization

Another method to use perspective, is the sphere visualization is described by Fairchild et al. [Fair93]. Sphere visualization is part of the VizNet visualization system, and it shows the relationships of multimedia objects mapped on a sphere. One object is selected and becomes the *object of interest* (OOI), this OOI is mapped on the center of the sphere and is therefore visible in full detail for users.

Objects that are closely related can still be seen in detail whereas unrelated objects are less visible since the mapping on the sphere implements a perspective distortion as shown in Figure 4.8.

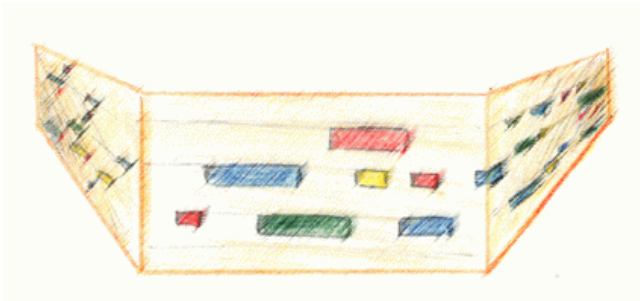


Figure 4.7: Perspective wall principle.

Figure 4.8: Sphere visualization principle.

To display hierarchical information nested spheres are used which become darker with increasing depth of nesting to help the users cognitive model of the metadata. Users can navigate the sphere visualization by rotating the sphere to bring other OOI to the front or by traversing links to the nested spheres.

4.3.4 BEAD

BEAD is a document visualization system that uses a physically-based force model to map multidimensional metadata into a three dimensional space [Chal92]. BEAD uses a spatial proximity algorithm to represent the relationships between documents.

The algorithm generates a landscape by placing the documents according to their similarity. The first implementation of BEAD used for the physical model only the metric from the co-occurrence of words for the placement. This leads to ball or cloud-like shapes which are hard to be interpreted for the users because of their complexity.

BEAD changed to a terrain visualization where the algorithm is forced to build a flatter, island-like shape as shown in Figure 4.9. The visualization lets the users now perceive apart the highlighted search results other close related objects by being grouped closely and also as overview over the whole data by the roughness of the terrain. BEAD allows also multiple users which are represented as cuboids with “eyes” to see their orientation. Users can interact by selecting an object or by performing a search within highlighted results.

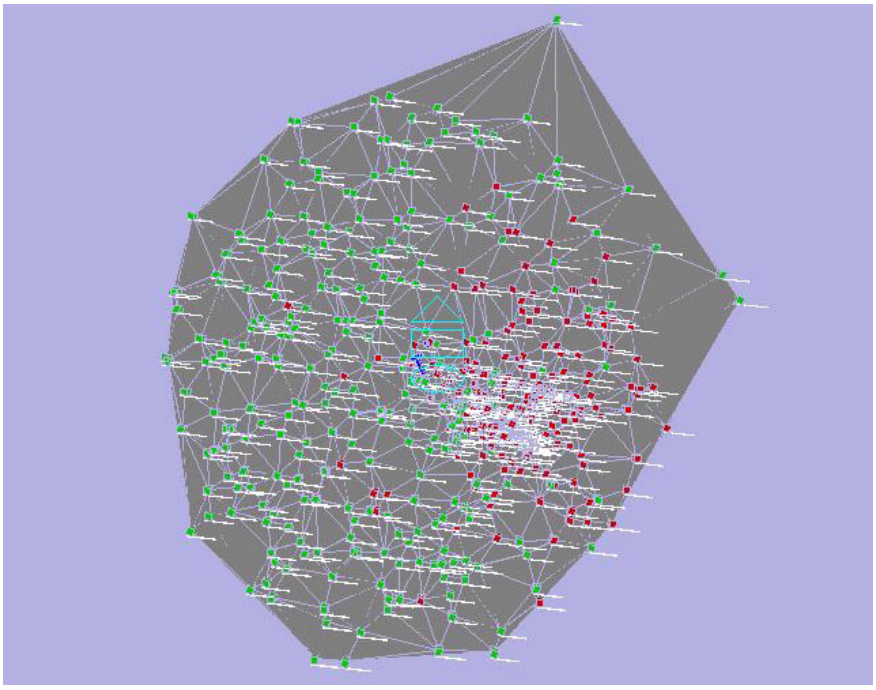


Figure 4.9: BEAD showing an overview of a data landscape from over 500 bibliographic references. (Source: [Chal92])

4.3.5 Populated Information Terrains (PITs)

Populated Information Terrains (PITs) use the Benediktine approach to display metadata with the main difference to other visualizations that the concept of co-operative work is implemented. Combining databases and virtual reality techniques, PITs allow multiple users to interact withe the visualization. Users are aware of the presence of other users and they are displayed within the data as three dimensional objects. This permits the sharing of information and co-operative work irrespective to the physical locations of the users.

Q-PIT by Benford is a prototype implementation of this concept which can display name tuples from a database[Benf94]. The metadata is mapped to extrinsic and intrinsic dimensions, and users are presented as cuboids with different colors to distinguish their identity as shown in Figure 4.10. Users can interact with the data selecting an object or with a new query. If a user changes the data itself the object moves in the visualization smoothly so all users are aware of this change.

4.3.6 VR-VIBE

An example implementation of PITs is VR-VIBE which is a three dimensional extension to VIBE [Benf95]. VR-VIBE visualizes bibliographies which are placed depending to the search keywords in the three dimensional space. Each keyword is a *Point of Interest* (POI) marked as octahedron so the documents are placed between the POIs as shown in Figure 4.11. Users can navigate freely in the visualization to see the structure of the information with the possibility of multiple views. The POIs can be moved to see which documents are pulled after them.

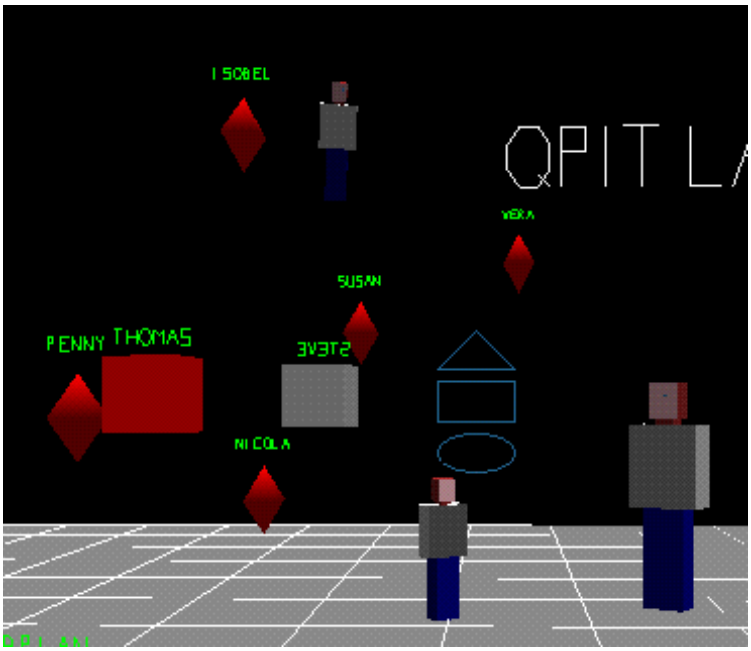


Figure 4.10: Q-Pit: Several users in a DIVE Q-Space. (Source: [Qpit94])

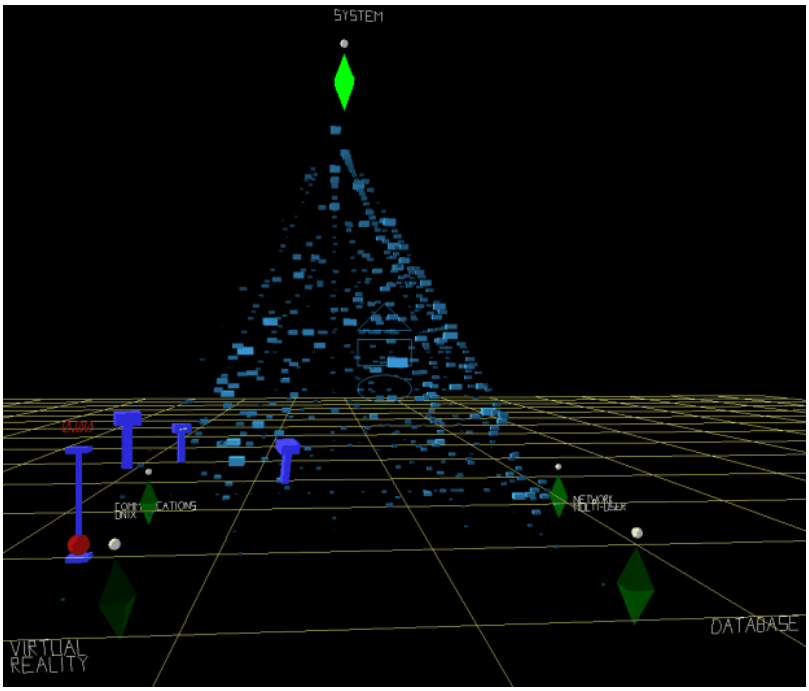


Figure 4.11: VR-VIBE showing a PIT containing five POIs. (Source: [Benf95])

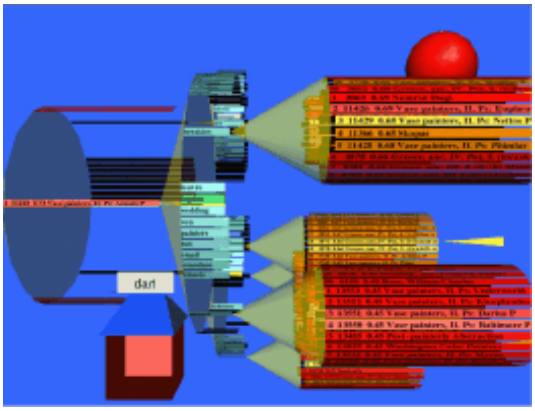


Figure 4.12: LyberWorld. (Source: [Hemm94])

4.3.7 LyberWorld

LyberWorld is an information retrieval user interface which uses cam trees for the visualization of documents [Hemm94]. The main goal of LyberWorld is the creation of a cognitive spatial model of the information, in a form that users can easily explore the data.

The visualization itself consists of two parts the *NavigationCone* and the *RelevanceSphere*. The NavigationCone visualizes the retrieval history and the RelevanceSphere displays the retrieved documents and their relevance to the search queries as seen in Figure 4.12. Cone or cam trees are normally only for hierarchically data, LyberWorld visualizes also networks by inserting redundant nodes and hidden paths. Users can navigate by unfolding nodes of interest or by rotating the whole tree.

4.3.8 Vineta

Vineta is a visualization tool to browse and query large bibliographic data [Kroh96]. Documents are presented in a three-dimensional space according to their semantic relevance. The two main metaphors for producing visualizations in Vineta are the galaxy metaphor and the landscape metaphor. As shown in Figure 4.13. the galaxy metaphor represents the navigation space as a galaxy of stars, documents are presented as fixed stars while terms are presented as "shooting stars". In the galaxy closely related documents are encoded by the proximity of the stars.

The landscape metaphor shown in Figure 4.14 represents the navigation space as a flat surface containing flowers with stems and petals. For a better depth perception the surface is textured, this also helps to judge the distance of the documents (flowers). The landscape metaphor comes from the study of ecological optics which emphasizes that perception of objects should never be considered apart from a textured ground surface. Flowers nearer to the users viewpoint are of more relevance to the initial query than flowers further from the viewpoint. The direction and color of petals on the flowers represent the search terms and their relevance to each document. To extrapolate the flower's position a stem is used which helps the users to find the position on the ground plane.

4.3.9 Narcissus

Hyper-structures are formed by typically large information stores with an arbitrary number and configuration of explicit relationships or links between data items. A typically example would be the graph formed from documents in hypermedia systems, where the documents and the links between them form the hyper-structure. The arbitrary arrangement of such structures and the

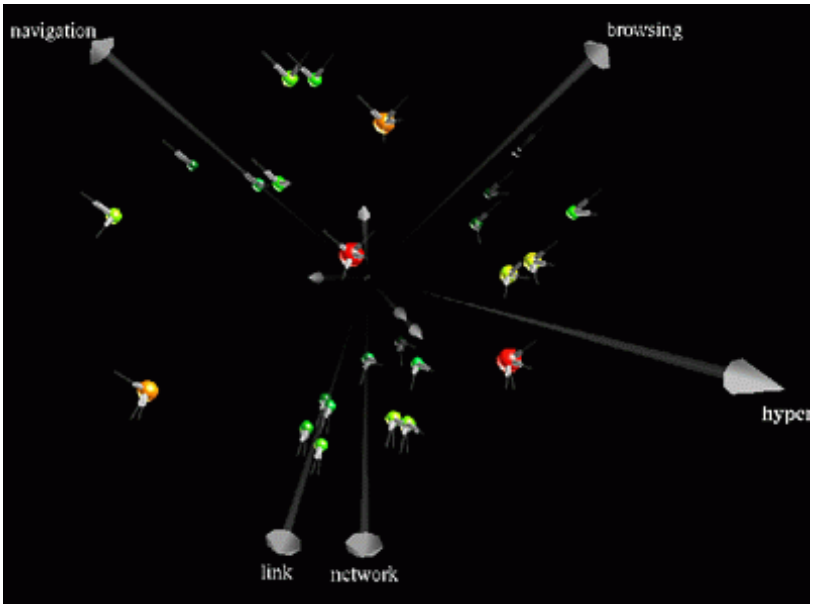


Figure 4.13: Vineta showing the galaxy visualization. (Source: [Kroh96])

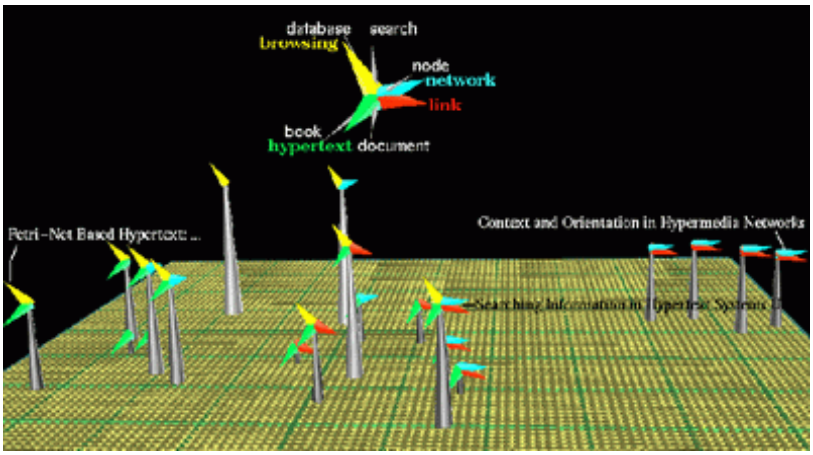


Figure 4.14: Vineta showing the landscape visualization. (Source: [Kroh96])

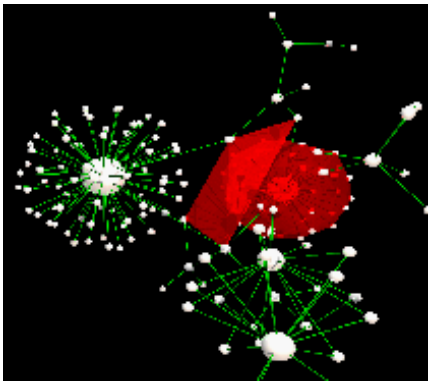


Figure 4.15: Ray-traced output from Narcissus. (Source: [Narc95])

inherent complexity makes visualization an extremely difficult task. Even simple structures can rapidly clutter the visualization and make the visualization useless. Narcissus is an information visualization system for creating three dimensional hyperstructures [Hend95]. One of the development goal of Narcissus was to display the structure of the dependencies between the various components of a software system. Figure 4.15 shows the visualization in Narcissus, it is created as wireframe three-dimensional graphs with the webpages displayed as spherical nodes and the links as directed edges. The layout itself is produced using a self-organizing technique, with new nodes introduced to the structure being placed randomly.

This form of visualization has the problem that the structure produced by the graph changes greatly when a node is inserted. Even the visualization of the same metadata can have a different structure. This inconsistency between runs effectively destroys the user's cognitive model of the structure thus requiring frequent re-orientation as the graph is updated.

4.3.10 FSN (FUSION)

FSN is file system navigator developed by Silicon Graphics. The visualization uses the landscape metaphor and shows the hierarchical information of the UNIX file system. Starting with the root of the tree closest to the user the branches are receding into the distance, away from the user's viewpoint. Each directory within the tree is represented as a pedestal, the height of this pedestal represents the combined size of the files contained within that directory as shown in Figure 4.16. The paths or links connect this directories and they can be traversed. Files within each directory are represented by boxes, placed on top of the directory pedestal. To represent the type of the file the box is adorned with a graphical image mapped onto the top surface Figure 4.17. The height of these file boxes represents their size, whereas the color of the boxes represents the age of the file. Users can combine now these features and the landscape metaphor to easily find for example prominent features such as large files or directories.

FSN is a fully functional file manager it allows apart of the visualization of the file structure, also the manipulation of the contents. Typically operations on files such as copying or moving them, also starting an external applications depending on the file type is possible.

4.3.11 Harmony Browser

The Harmony Internet browser [Andr95] provides 3D information landscape visualizations of Web sites as shown in Figure 4.18. Harmony can be used just as FSN but in addition a set of 3D icons represents the current document type to give a quicker visual impression.

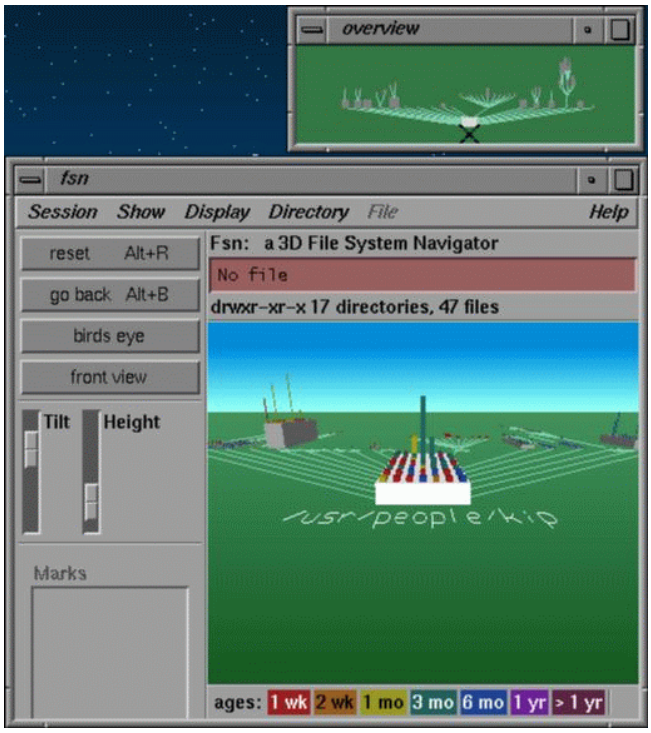


Figure 4.16: FSN visualization of a UNIX file system. (Source: SGI Webpage)

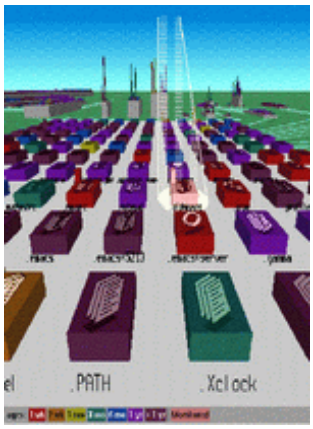


Figure 4.17: Detail FSN Visualization. (Source: SGI Webpage)

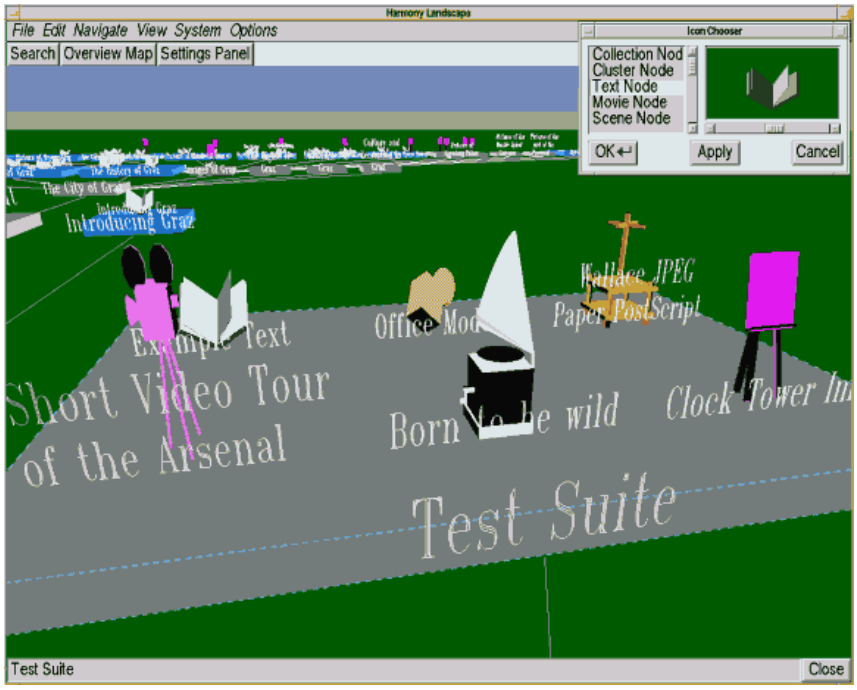


Figure 4.18: Harmony textured information landscape. (Source: [Andr95])

4.3.12 Measuring the Web

Tim Bray developed a three dimensional visualization for the web [Bray96]. The metadata is taken form the Open Text database. The goal was to visualizes connections between web sites, sites pointing to other sites, the numbers of pages it contains, and so on. The main mapping for sites distributed in space was in a fashion that reflects the strength of their connectivity. Figure 4.19 shows that UIUC including NCSA is the Web's most visible site. The scene in Figure 4.20 shows a wider view of the world wide web. Commercial sites are rendered in blue, and network infrastructure in cyan. Some of the principles Tim Bray adopted include:

- The "site" is the appropriate unit of display.
- Sites should be distributed in space in a fashion that reflects the strength of their connectivity
- The appearance of a site should reflect its visibility, as measured by the number of other sites that have pointers to it.
- The appearance of a site should reflect its size, as measured by the number of pages it contains
- The appearance of a site should reflect its luminosity, as measured by the number of pointers with which it casts navigational light off-site.

4.3.13 The SOMLib Digital Library System

The SOMLib library is based on the self-organizing map (SOM) [Raub99]. A program called libViewer was developed, it visualizes information like the size of the underlying document, its type, the date it was created, when it was accessed for the last time and how often it has been

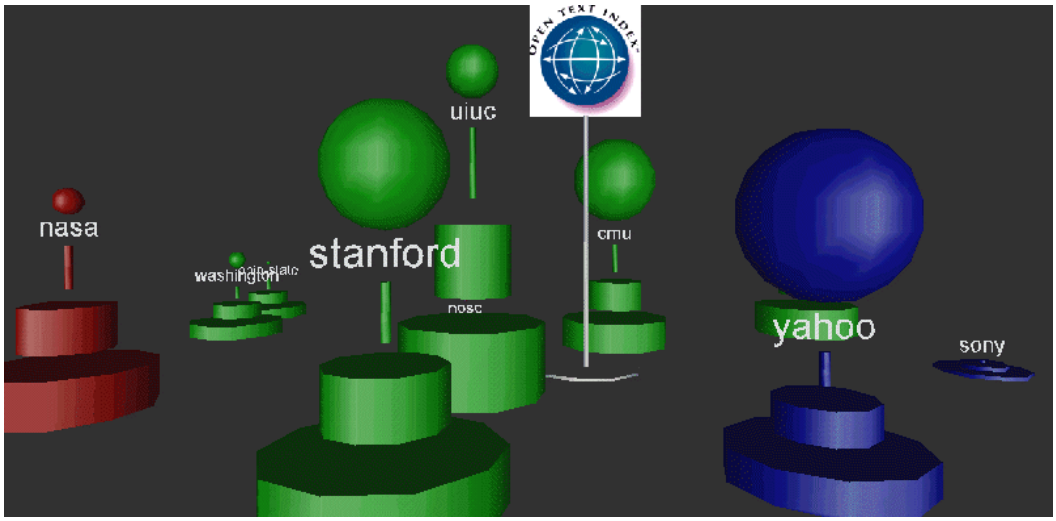


Figure 4.19: Visualization of some major web sites. (Source: [Bray96])

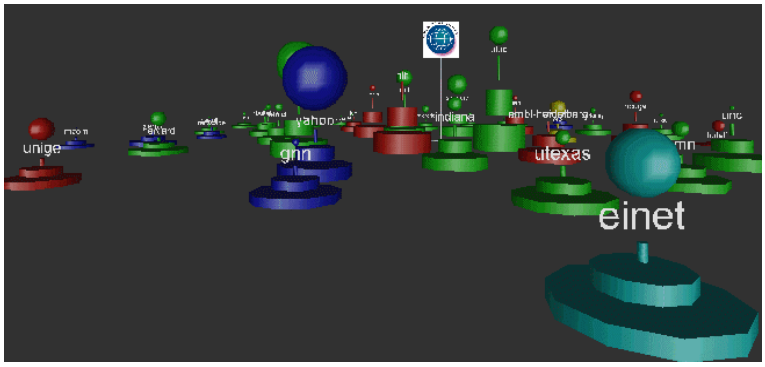


Figure 4.20: Tim Bray's visualisation of sites in the Web Space. (Source: [Bray96])

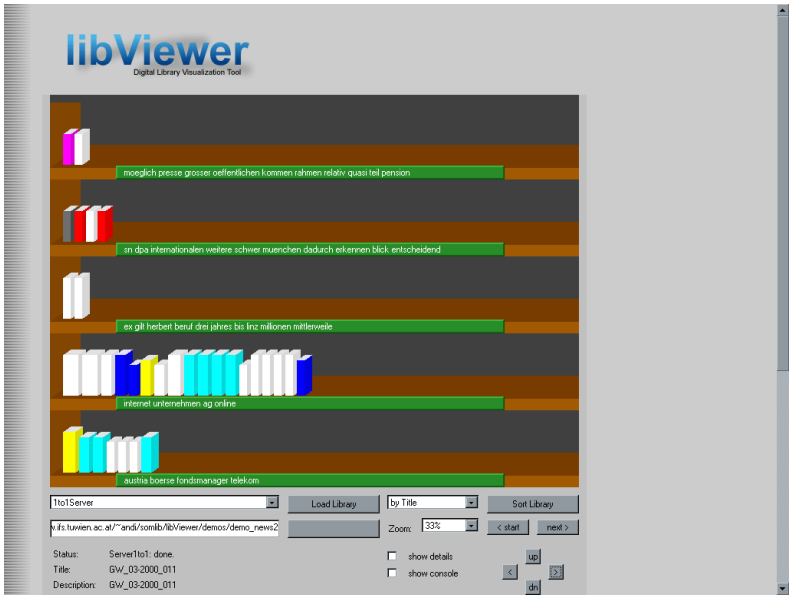


Figure 4.21: A bookshelf in libViewer. (Source: [Raub99])

accessed at all, its language etc. A set of metaphors is implemented in the libViewer to allow a flexible mapping of metadata attributes to graphical representations in order to best suit the requirements of the user as well as the resources present in the library.

Users can define a number of mappings to optimize the representation for the requirements of a digital library, ranging from a rather realistic representation of the items in the library to a more abstract one designed for special exploration purposes. The following metaphors have been realized: Each piece of work in a digital library needs a physical representation. A set of templates is defined to represent, for example, hardcover books, paperbacks, binders, manuscripts, boxes for audio, video and software components or links to other libraries in order to provide a rather realistic visualization of library resources. As most dominate feature after physical location color is used which can easily be detected at long distance. Thus, color can be used to represent a variety of attributes in a very distinctive way, such as language, publication series, genre, topical classification etc. The amount of information available in a book or magazine is intuitively judged from the size of the physical object but also some genre information can be visualized, for example, oversize format books such as an atlas or art collection books vs. small paperbacks.

Figure 4.22 provides a sample representation of a digital library containing a number of books, technical reports, papers and multimedia resources as well as hypertext links. Figure 4.21 shows some books in the libViewer in detail, from the color and the position of the books the users can immediately find for example the German-English dictionary.

Figure 4.22: Details in libViewer. (Source: [Raub99])

Chapter 5

Java and Java3D

5.1 Java 2

Java is a simple, robust, object-oriented, platform-independent multi-threaded, dynamic general-purpose programming environment. Java is different from other development environments, it does not generally execute on the host machine, even when it is compiled. Instead, it executes in a virtual machine, which in turn executes on the host computer, as shown in Figure 5.1. This means that code is written once and can be run without re-compilation on other hardware platforms. In effect, the Java code always runs on the same machine: the Java Virtual Machine (JVM).

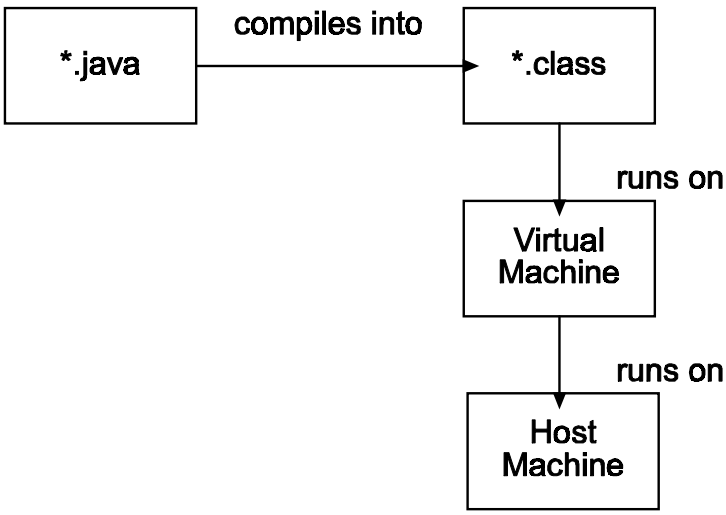


Figure 5.1: The standard Java environment.

5.2 Java3D

Java3D is part of the Java media packages as shown in Table 5.1. The Java3D API is an applications programming interface to create three-dimensional applications as well as applets. For both applications and applets the users need to install the Java3D Runtime package. Java3D has the advantage that Java code is *"written once, and runs anywhere"* and the API exists for several platforms.

In the API is a collection of high-level constructs for the creation and manipulation of three-dimensional visualizations. With Java3D the programers can concentrate on the geometry and structure of the visualization or its animation, because low-level details are handled inside Java3D. The main construct in Java3D is the *virtual universe*, which can be created precise in a variety of actual physical sizes, from astronomical to subatomic.

The goal of the Java3D API is to combine the best ideas of existing low-level graphic API's like OpenGL or Direct3Draw with the high-level concept of scene graph-based systems. The API is straightforward to use, for example the details of rendering are hidden from the programer and are handled automatically. The renderer of Java3D takes also advantage of the threads in Java and is capable of rendering in parallel.

Package	**Description**
Java 2D	Provides an abstract imaging model that extends the JDK 1.3.1 AWT package, including line art, images, color, transforms, and compositing.
Java Media	Specifies a unified architecture, messaging protocol, and programming.
FrameWork	Interface for media players, media capture, and conferencing; Comprises three separate APIs (Java Media Player, Java Media Capture, and Java Media Conference)
Java Collaboration	Allows for interactive, two-way, multi-party communications over a variety of networks.
Java Telephony	Integrates telephones with computers and provides basic functionality for first-party and third-party call control.
Java Speech	Provides Java-based speed recognition and speech synthesis (text-to-speech).
Java Animation	Provides for motion and transformations of 2D objects while utilizing the Java Media Framework for synchronization, composition, and timing.
Java3D	Provides an abstract, interactive imaging model for behavior and control of 3D objects.

Table 5.1: The Java media packages. (Source: [Java01])

The Java3D is object-oriented and uses the scene graph model as its internal structure. The description for the virtual universe is contained in this scene graph. In the API there are several classes for constructing and manipulating the scene graph. With this approach Java3D includes all necessary information, like the geometrical data, the attributes, and the rendering options in this structure.

Another important issue in Java3D is that the programers can think about three-dimensional objects and not about triangles. The core classes of the Java3D API are in the `javax.media.j3d` package. At the moment there are more then hundred classes, but to create for example a simple universe only few classes are needed. Figure 5.2 shows some of the important classes of the Java3D API.

In addition to the core package, other packages are delivered with Java3D. The most important is the utility package `com.sun.j3d.utils`. In this package there are extensions to the core and it can be divided into four categories: content loaders, scene graph construction aids, geometry classes, and convenience utilities. Planed utilities for future versions of Java3D will be in this package, e.g. free-form curves. Java3D programers can reduce the amount of source code drastically when they use the utility classes.

From the `java.awt` package Java3D uses the Abstract Windowing Toolkit (AWT) to create the rendering window. The mathematical operations, necessary for computer graphic are in the package `javax.vecmath`, as the name of this package suggests it defines classes for points, vectors, matrices, and mathematical operations. See Figure 5.3.

5.3 The Java3D Scene Graph

In Java3D the virtual universe is created from a scene graph. A scene graph is a tree structure which is assembled from objects, sound, lights, location, orientation, and appearance. As all trees in computer science the scene graph is made up of nodes and arcs. Nodes represent the data element, and the arcs the relationship between two data elements. Arcs can represent two kinds of relationships between the elements, the parent-child relationship and the reference relationship.

Java3D has group nodes which can have any numbers of children but only one parent, leaves can not have children. Node component objects are associated to a node by a reference, e.g. the node could be a color cube and in the node component object is saved how to render the color cube. The scene graph structure in Java3D is built up in a way, that for every leaf there is a unique scene graph path down from the root node to this leaf. Every scene graph has to be a Directed Acyclic Graph (DAG). A DAG is data structure composed of elements and directed arcs in which no cycles are formed. In the Java3D world, the elements of the DAG are Group and Leaf objects, and the arcs are the parent-child relationships between Groups and Leaf objects. This means a structure where no element can be its own parent. Figure 5.4 shows an example of a scene graph, each object has a shape and a 3D position relative to its parent node. The scene graph path for one leaf includes all the state information like location, orientation, and size of the visual object. Depending on the attributes for the scene graph path the Java3D renderer can build up his own rendering order to be most efficient. The Java3D programmer normally *does not have control* over the rendering order of objects. Every scene graph for a Java3D program has two thematically separated parts. The first one seen on the right side of Figure 5.5 deals with the way the view will be shown on the output media e.g. the monitor. The remaining part of the scene graph is called the content graph here the programer adds the objects to be inserted in the virtual universe as shown in Figure 5.6.

The graphic representation of the scene graph can be used as a design tool and for documentation. In the design phase the standard symbols as shown in Figure 5.7 are used to create the virtual universe which is then be used as specification for the program. When the program is implemented following the specifications, the drawing of the scene graph can then be used for documentation.

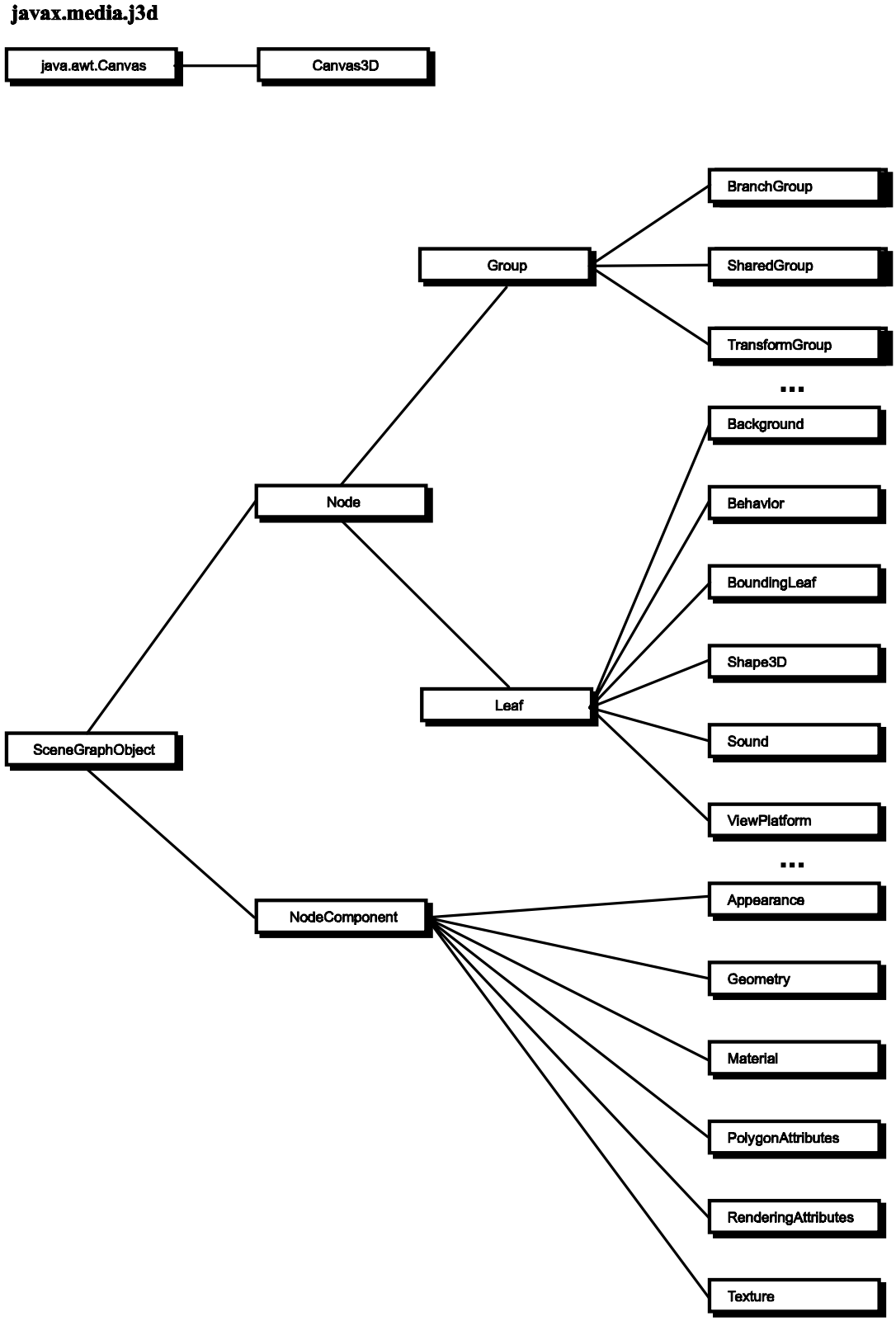


Figure 5.2: The main J3D classes.

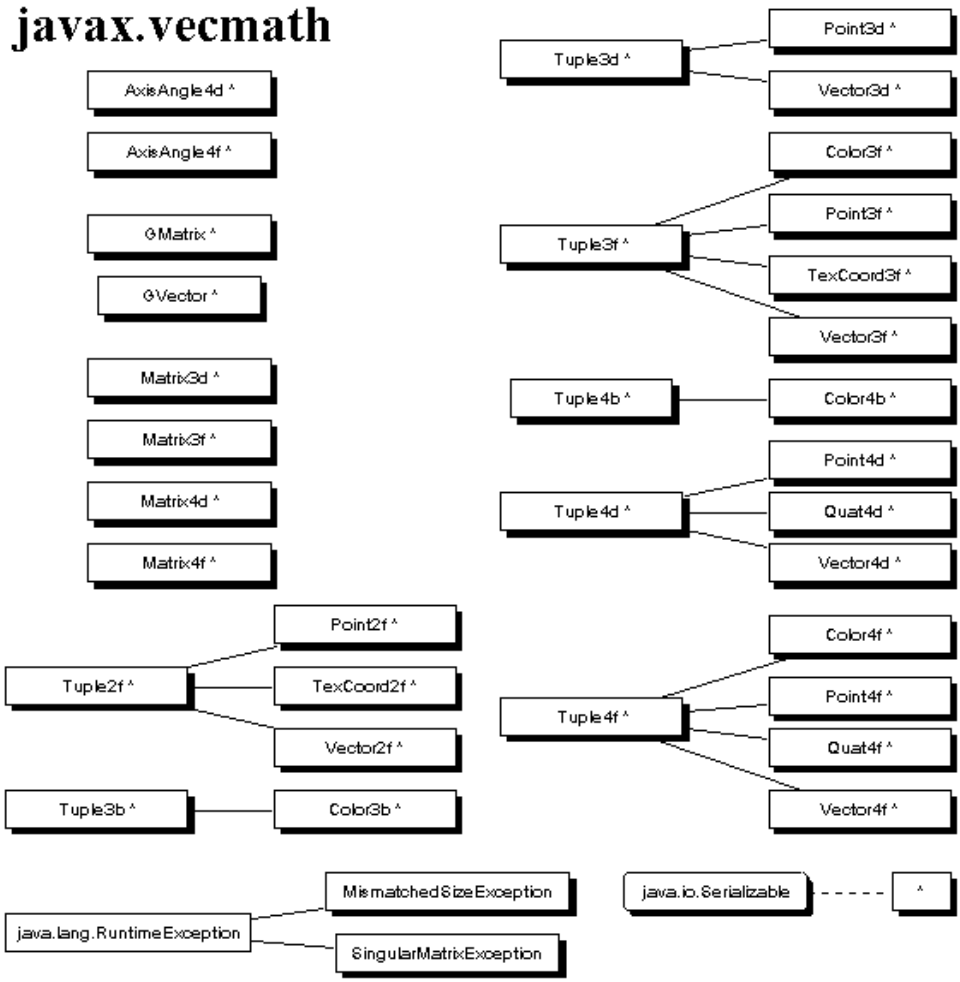


Figure 5.3: The J3D Vecmath class hierarchy diagram. (Source: [Java01])

5.4 Java3D Rendering

As shown in the pseudo code in Figure 5.8 the Java3D renderer starts in an infinite loop when a branch graph containing an instance of View becomes live in a virtual universe.

5.5 Scene Graph Objects

The Java3D scene graph is built up from node objects, these node objects are called *node component objects*. All of these objects are derived from a common scene graph object class, which is abstract and defines common methods for the scene graph objects. Attaching a scene graph object to an existing scene graph, which is attached to a virtual universe, is called making the object *live*. Before attaching this scene graph object, the scene graph subpath can be *compiled* to optimize rendering. If the properties of this scene graph object have to be altered after their creation, capabilities have to be set to *allow explicitly* a modification. By default all capabilities are cleared for speed reasons. A programer should only set themwhen needed. (e.g. `ALLOW_MATERIAL_READ`).

Figure 5.4: The concept of a scene graph: The hierarchial structured composition of a monocycle.

5.5.1 Node Objects

There are two kinds of node objects, the group node and the leaf node. Group nodes are the inner elements of the scene graph, the specify how their children objects are organized. Leaf nodes are the elements that Java3D is actually rendering, like geometric objects, light, and sounds.

5.5.2 Shape3D Node

To create for example a cone in Java3D a Shape3D node is needed. This node is a leaf node and contains all the geometric data for the object. The Shape3D node holds a list of geometry objects and a single appearance object. In the cone example the programer needs only to define the size and the appearance of the cone but not how the triangles generate the cone internally. With the appearance object the programmer can specify the Shape3D attributes like color, material, and the texture.

5.5.3 Virtual Universe Object

The virtual universe is an object that consists of locale objects which contain the scene graph, as shown in Figure 5.9. A typical Java3D application has only one virtual universe. The virtual universe is defined as a three-dimensional space with associated objects.

For very large virtual universes Java3D has developed the concept of *locales.* One or more locale can be in a virtual universe, each defining a co-ordinate origin for its children.

5.5.4 Locale Object

A locale, with its associated high-resolution coordinates, serves as the next level of representation down from a virtual universe and is the root for a sub scene graph. High-resolution coordinates act as an upper-level translation-only transform node. All children of a locale object have their location relative to the locale's high-resolution coordinates. Locales have no parents they are implicit attached to a virtual universe during construction.

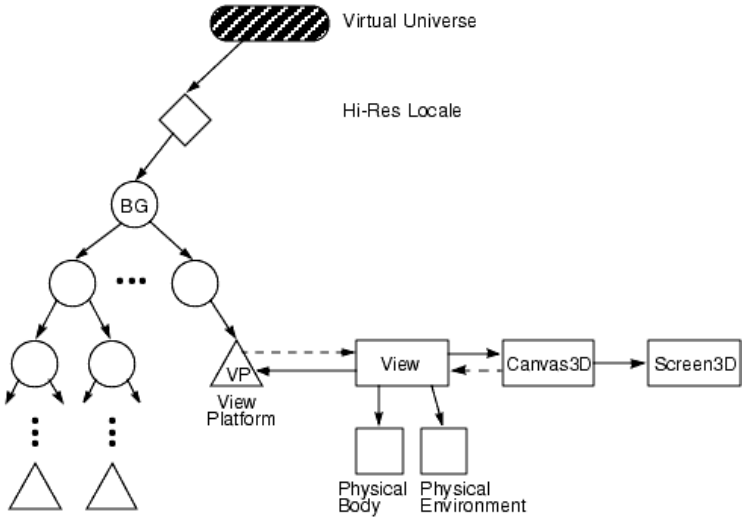


Figure 5.5: The J3D Scene View. (Source: [Java01])

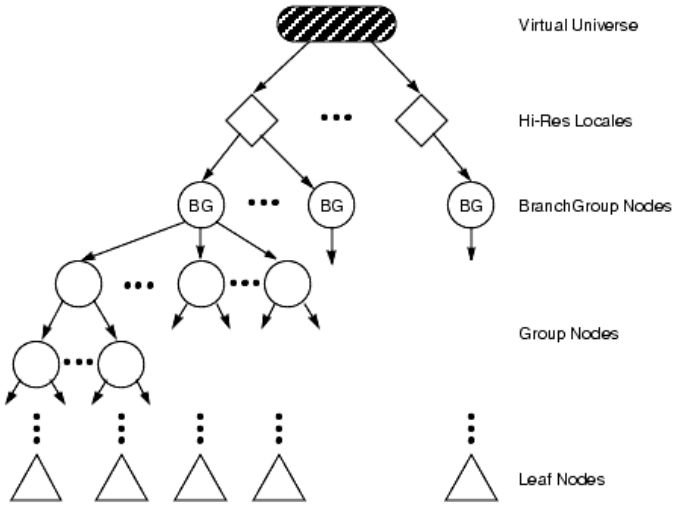


Figure 5.6: The J3D Scene Content. (Source: [Java01])

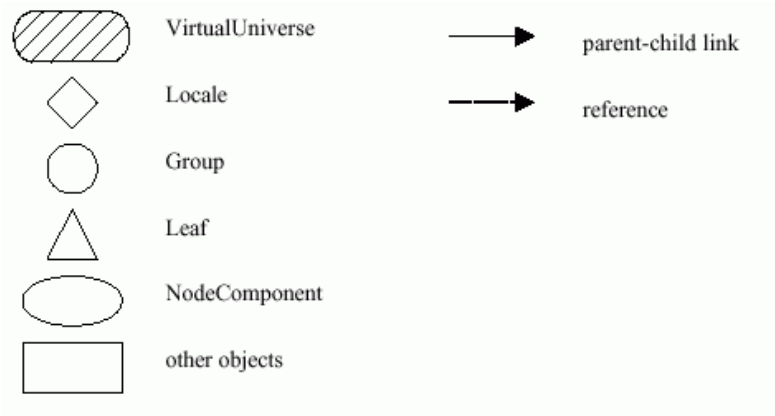


Figure 5.7: Symbols Representing Objects in Scene Graphs. (Source: [Java01])

```
while (true)
        process input
        if (request to exit) break
        perform behaviors
        traverse the scene graph and render visual objects
cleanup and exit
```

Figure 5.8: The Java3D rendering process.

5.5.5 Canvas3D Object

The output of the Java3D renderer is shown in the Canvas3D object. A view object can have multiple Canvas3D objects to see the scene from different positions. When a Canvas3D object is attached to a view object, Java3D renders the specific view onto the canvas on the output, which is normally a monitor.

In FAEJ3D a main Canvas3D shows a perspective view of the objects and a second smaller Canvas3D shows a parallel top view.

5.5.6 Screen3D Object

Java3D is capable of rendering the objects to different outputs such as a monitor, shutter glasses, virtual reality cubes, and so on. In the Screen3D object the physical parameters of the output like height and width in pixels are saved.

5.5.7 View Object

Figure 5.5 shows a View object attached to a simple scene graph for viewing the scene. With the view object the programer can define the camera model of the scene. This model has form and size of the viewing frustum, orientation, and placement in the virtual environment. The Java3D view object model allows a six-degrees-of-freedom tracking, and can follow the tracker to fit the users viewpoint exactly.

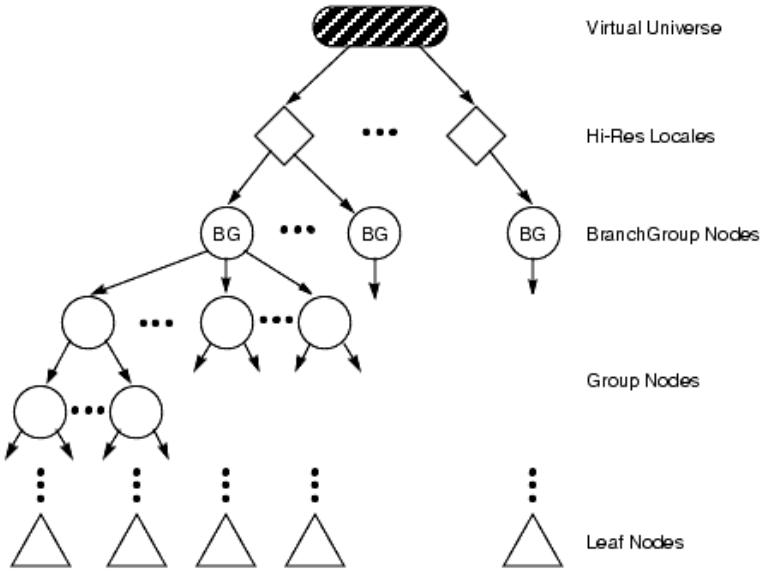


Figure 5.9: The Virtual Universe. (Source: [Java01])

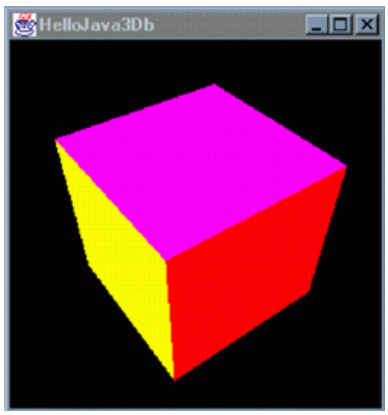


Figure 5.10: Image of the rotated color cube.

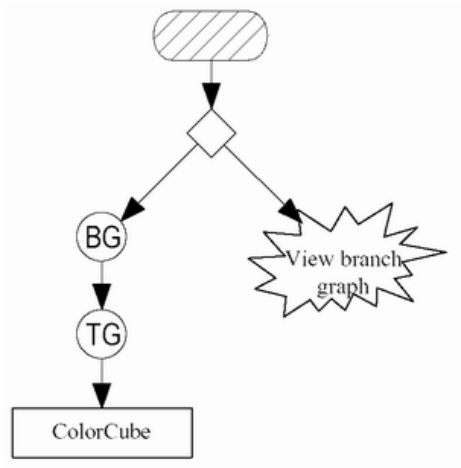


Figure 5.11: Scene Graph for Hello J3D World. (Source: [Java01])

```
1.  public BranchGroup createSceneGraph() {
2.  // Create the root of the branch graph
3.  BranchGroup objRoot = new BranchGroup();
4.
5.  // rotate object has composite transformation matrix
6.  Transform3D rotate = new Transform3D();
7.  Transform3D tempRotate = new Transform3D();
8.
9.  rotate.rotX(Math.PI/4.0d);
10.  tempRotate.rotY(Math.PI/5.0d);
11.  rotate.mul(tempRotate);
12.  TransformGroup objRotate = new TransformGroup(rotate);
13.
14.  objRotate.addChild(new ColorCube(0.4));
15.  objRoot.addChild(objRotate);
16.  return objRoot;
```

Figure 5.12: The code for the color cube example.

5.6 A Typical Java3D Program

To make a simple "Hello J3D World" example as shown in Figure 5.10 its only necessary to create a root node, some transformations in this case two transformations since the cube is rotated around the x and the y axis and the leaf object a ColorCube. The scene graph for this examples is shown in Figure 5.11.

The source code part for the content side of the scenegraph is shown in Figure 5.12. To see the ColorCube on the screen the root node has to be attached to a view. The quickest way to do this is to create an instance of the Java3D SimpleUniverse. Setting up a Canvas3D e.g. in an Applet and a small view transformation the moves the users back from the center creates the ViewBranch graph and results in the Hello J3D World program.

5.7 Java3D Software

Like all the Java API's Java3D is provided both as a Software Developer Kit (SDK) and as a Run Time (RT). Since Java3D makes heavy use of the graphic hardware there are three releases of the SDK and RT for the main graphics hardware. The first is for Microsoft with the DirectX package the second is for OpenGL, a standard defined from Silicon Graphics [Open97], and the third is for the Sparc/Solaris hardware:

- Windows (DirectX Version) RT for the JRE
- Windows (DirectX Version) SDK for the JDK
- Windows (OpenGL Version) RT for the JRE
- Windows (OpenGL Version) SDK for the JDK
- Solaris/SPARC RT for the JRE
- Solaris/SPARC SDK for the JDK

5.8 Conclusion

Java3D is defined as a platform-independent 3D API which is meant to run on the current generation of computer system characterized outwardly by a 2D display, a mouse, a keyboard, and maybe sound output. In terms of performance, today this would mean say., a Pentium III with 500MHz and 128MB of RAM, like the PCs used mostly in offices and homes.

It has to be clearly stated that Java3D is *not* a substitute for programming directly in OpenGL or DirectX, since the Java Virtual Machine guaranties platform-independence but not speed.

A beginners tutorial to Java3D is available, but since this tutorial uses only the SimpleUniverse, it is only suitable for learning the scene-graph concept and basic features like basic transformations, light, texture, animation, and interaction.

To go deeper there is the book: "The Java3D API Specification" [Sowi97], for a detailed description of all classes. Another recommend book is: "3D User Interfaces with Java3D" [Barr01]. Here the reader can not only find many samples for Java3D, but also a complete discussion about user interface aspects in general. For a complete introduction to computer graphics there is the famous Foley et al[Fole90] which has comprehensive coverage of general computer graphics concepts including representation of points, lines, surfaces, and transformations.

Chapter 6

The File Attribute Explorer in Java3D (FAEJ3D)

The File Attribute Explorer in Java3D or FAEJ3D is an attempt to use known aspects from information visualization theory within a program that uses Java3D as the visualization tool.

The goal of developing the Java3D File Attribute Explorer was to map as many dimensions as possible to extrinsic and intrinsic spatial dimensions. Platform independence and a visualization for standard computer hardware was the goal. Since Java runs on a virtual machine and there are implementations for all modern operating systems, Java was selected as programing language.

Java and Java3D promised to be a good solution to achieve the desired goal. Due to limitations when a large number of 3D-shapes are displayed, some of the ideas had to be shelved to maintain bearable display and interaction speeds. The main cut was the decision to use a single glyph for one data object, which rapidly reduced the number of mappings and so possibilities for having many extrinsic and intrinsic spatial dimensions.

6.1 Design Principles for the Java3D File Attribute Explorer

The main objective was to separate the visualization from the data generation and the mapping. With this approach in mind a J3DPanel was made, which can be used like the standard jPanel from the java.swing classes.

The J3DPanel can be used like a "Java-Bean". The placement of the J3DPanel is the same as any other swing container (e.g. `contentPanel.add( J3DPanel, ''east'')`). Many efforts in the J3D side of the program make it possible to generate a 3D Visualization of data without knowing a single Java3D command. The simple setup for the J3DPanel is described in the next chapter.

From the theory of information visualization was taken mainly the Gestalt-laws and the Benediktine Triple-space approach. Extrinsic dimensions specify location in space (e.g. x, y, and z coordinates). Intrinsic dimensions determine characteristics of the resulting point in space such as glyph shape, glyph size, and glyph color.

For speed reasons there are only two shapes; a cone which is the standard object and a cube. A cone as standard object was chosen because of its uniqueness in shape from all view directions. A single cone also remains identifiable in a pile whereas a cubes might appear to be as a single glyph. The extrinsic and intrinsic values are set outside the J3DPanel so the programers have full control over this part but they are released form Java3D internals like transformation, light, appearance, and so on.

The advantage of the Triple-space is that many laws from the information visualization theory

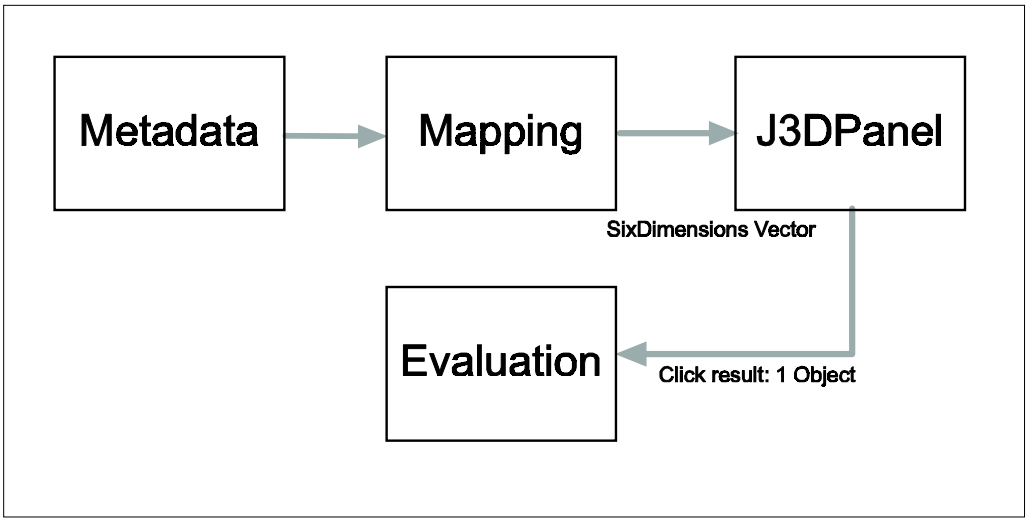


Figure 6.1: The programflow of FAEJ3D.

are implicitly used. The law of proximity for example lets the users perceive data as groups and this lets the users find data closely related with the given extrinsic settings immediately and preattentive processing helps the observers to group glyphs with the same color or shape.

Figure 6.1 shows the program flow of a program using the J3DPanel. The whole visualization is one object which sends back the data from the selected glyph via a callback set up by the application. The main design principles used in FAEJ3D and J3DPanel are:

- Operating system independence.
- Reusable 3D visualization.
- A 3D representation of metadata mapped onto the interval [0..100] for six dimensions.
- Programers have full control over the data and the mapping the want to bring the metadata to the interval [0..100].
- The users can select different mappings to view the data; with the mouse users can zoom in and by clicking a glyph they get details on demand.
- Users can set up a file filter.
- Common 3D navigation with a 3 or 2 button mouse (rotation is mapped to the left, zoom to the middle, and translation to the right mouse button).
- Secondary navigation with control buttons.
- Output can be input for a new visualization.
- A parallel view as top view to see the data always from the same location.

6.2 The FAEJ3D Classes

The main class is the `application.class`, this class creates an instance of the `mainframe.class` and centers it on the screen. The `mainframe.class` handles all the necessary algorithms to load the files into a vector and to create depending of the user-defined mapping a six dimensional

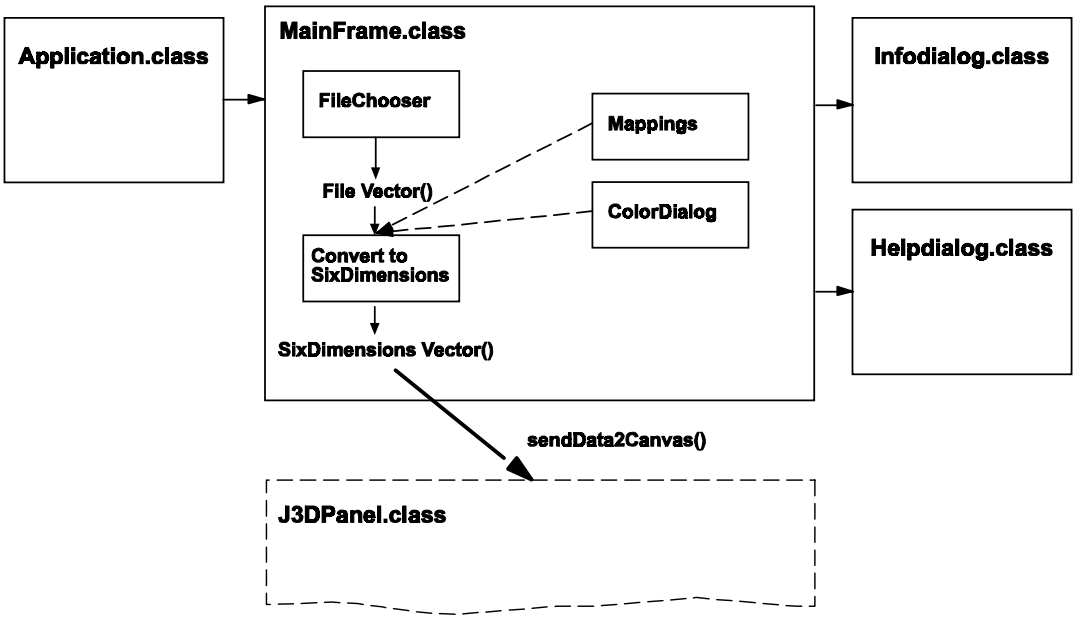


Figure 6.2: The FAEJ3D Mainframe Class.

vector which will then be passed to the `J3DPanel`. Figure 6.2 shows a scematic representation of the major functions of the `mainframe.class.` As can be seen in Figure 6.1 there is a clear separation between the generation of the `sixDimensions` vector and the `J3DPanel`.

The Java3D visualization is programed in a class derived from JPanel. This has the virtue that a programmer can use the `J3DPanel.class` like a normal JPanel from the `java.swing` library. How easy it is to set up the J3DPanel for a new application will be shown later in this section. Figure 6.3 shows in detail how the `J3DPanel.class` works and how one object is returned from the whole displayed data.

6.3 Using the J3DPanel Class for a new Application

To use the J3DPanel in a new program, the programer has to observe the following steps:

- Load the metadata to be visualized.
- Convert the data to a SixDimensions Vector.
- Add the J3DPanel at the appropriate place in your application.
- Set up the mouse callback and interface.
- Send the SixDimensions vector to the J3DPanel via `setVector(data).`
- Evaluate the returned object data.

To set up the mouse callback, some simple steps are needed as shown in Figure 6.4. Now the programer has in `returnedObject` the object data from the selected glyph and can start the evaluation of the data.

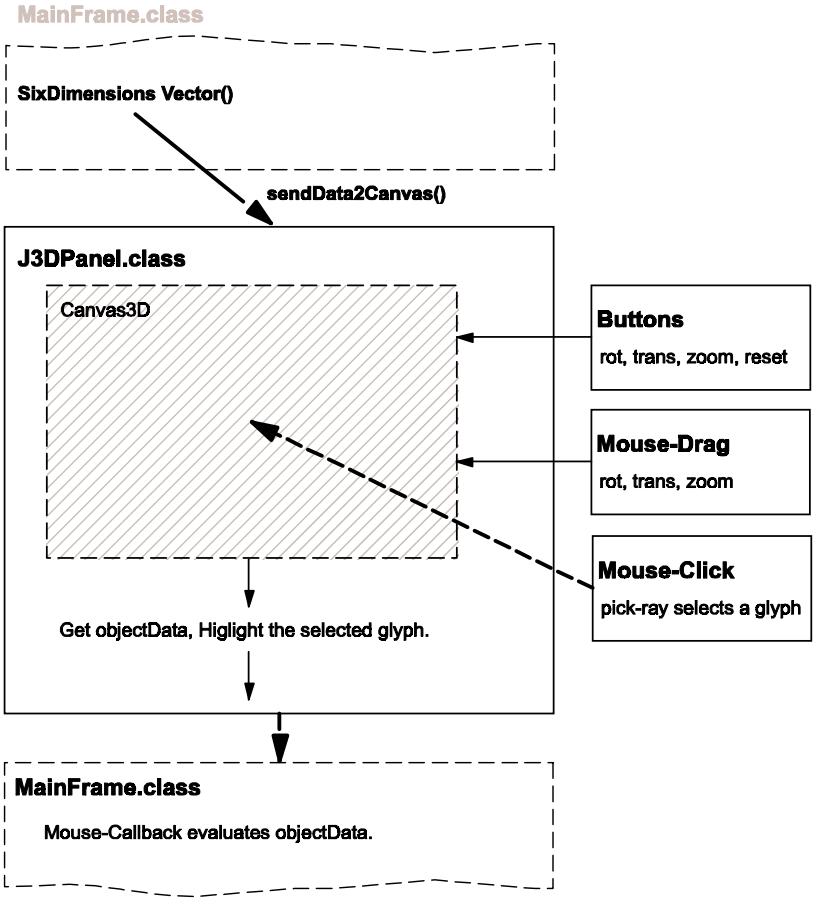


Figure 6.3: The J3DPanel Class.

First the programer needs the Mouselistener interface
```
public class MyProgram implements MouseListener{....}
```
....
add somewhere the J3DPanel to the root panel at the geometrically location you want
add the mouslistener
```
/**
 * addthe mouselistener
 */
 J3DPanel.getCanvasComponent().addMouseListener(this);
```
...
implement the mouselistener interface
```
/**
 * mouslistener
 */
 public void mousePressed(MouseEvent e) {
 }
 public void mouseReleased(MouseEvent e) {
 }
 public void mouseEntered(MouseEvent e) {
 }
 public void mouseExited(MouseEvent e) {
 J3DPanel.position(); // needed !
 }
 public void mouseClicked(MouseEvent e) {
// get the object from the j3d mousepick
 returnedObject = J3DPanel.getObject();
```

Figure 6.4: Setting up the mouse callback.

6.4 A Sample Application Made with J3DPanel, MP3ViZ

To illustrate the ease in which a new visualization can be made an application to visualize the ID3 metadata associated with MP3 audio files was built. MP3 files are audio files coded in the ISO-MPEG Audio Layer-III. The ID3 tag version 2.3.0 informal standard can be found at [ID301].

Since the attempt here is to show the simplicity of the visualization implementation and not the mapping there were made only six different predefined mappings. The application extracts from MP3 file the following metadata:

- Song title
- Artist name
- Album name
- Song length
- Track number
- Year
- Genre

In this short example "Genre" which is one of 125 predefined genres is always mapped to color. The user can load starting from a root directory (recursively) MP3 files and display them depending on the selected mapping in the J3DPanel. When the user clicks on one of the glyphs

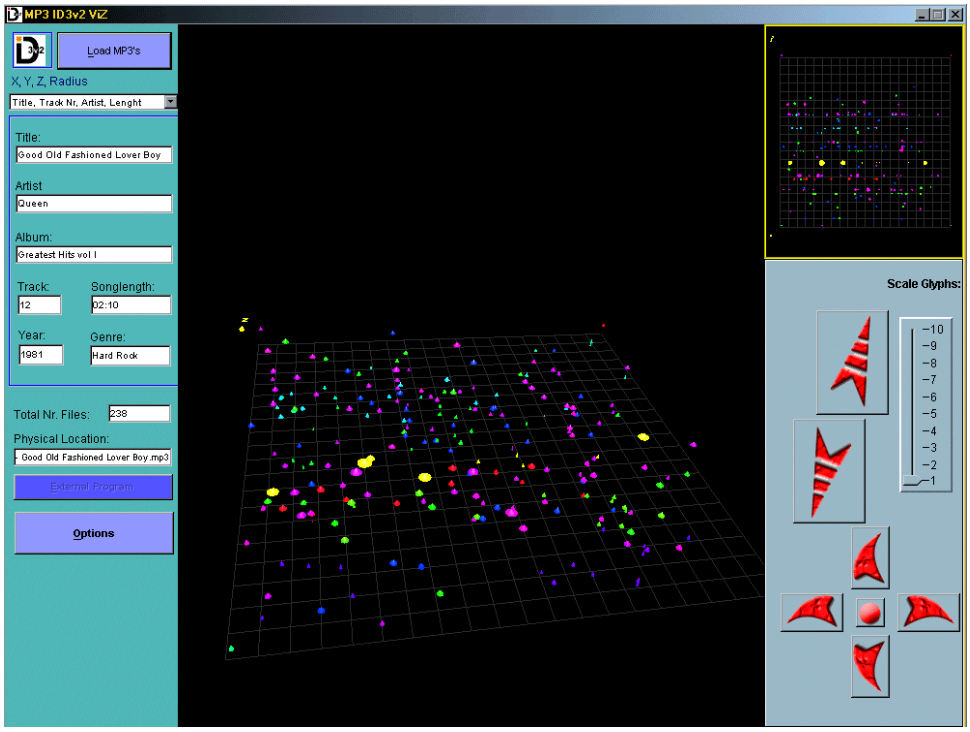


Figure 6.5: MP3ID3v2 Viz Screenshot with the start-up settings.

the corresponding metadata is displayed in the evaluation panel on the left. If the user has an external program such as WINAMP or the MS-MediaPlayer, then the user can listen to the selected MP3 file by clicking on "External Program".

Figure 6.5 shows 238 MP3 files where the mapping was set to get an overview over all files depending on their `''Title''` and `''Artist''`. This would be the appropriate setting to find, say, a song from `''Adam Doe''` with the title `''zzz makes the bee''`.

If users want to see the files that are on the same CD closer together they have to change the mapping to `''Album''` on the x -axis and something that songs from one CD in common like `''Year''` on the z -axis. Figure 6.6 shows now piles of cones corresponding to the songs on one CD.

In front is seen a cyan pile and Gestalt proximity law lets the user perceive the cones as one pile, even if there are some cones missing. When the users zoom (see Figure 6.7) they can see by clicking the cones that this is the Album `''Greatest Hits Vol 1''` by `''Queen''` and its easy to find out which songs are not yet converted to MP3 from this CD.

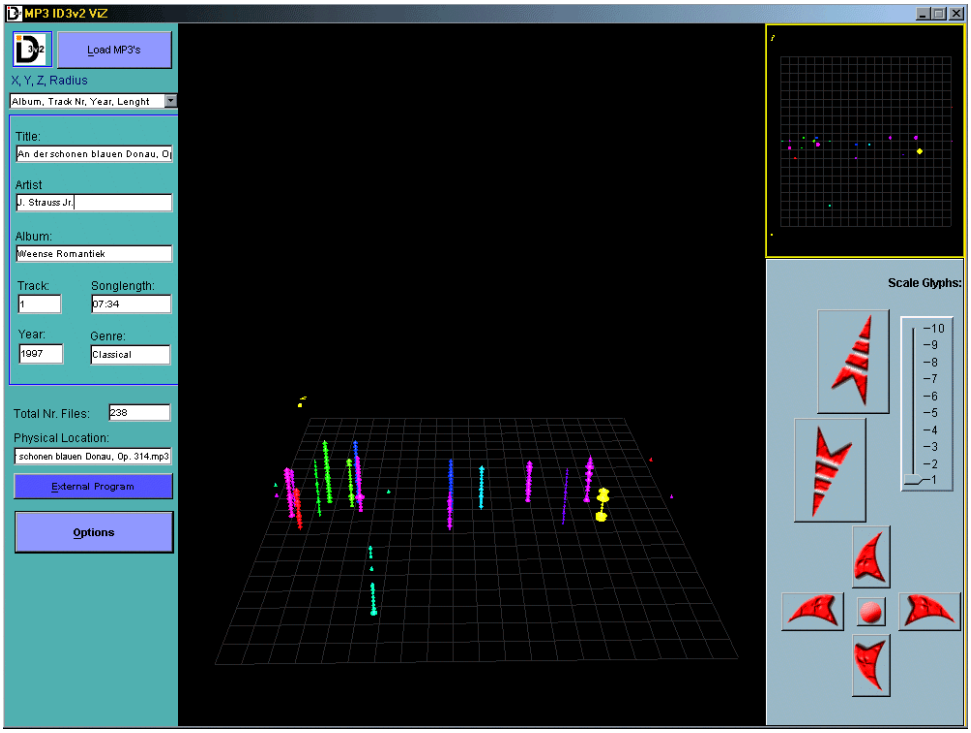


Figure 6.6: MP3ID3v2 Viz Screenshot with x set to album, y track number, and z to year.

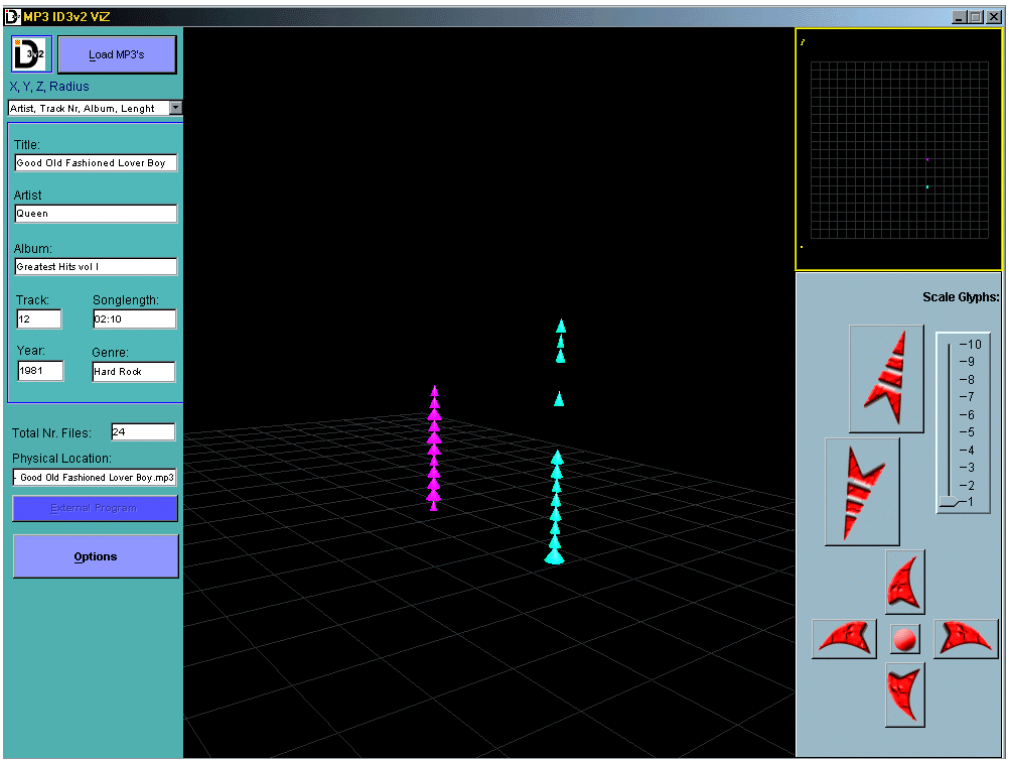


Figure 6.7: MP3ID3v2 Viz Screenshot with z set to album to see Mp3's from one album.

Chapter 7

Selected Details of the Implementation

First the implementation of the J3DPanel and its internal structure will be described and how the Java3D scene graph is built up. Then will be the description of the FAEJ3D its usage, capabilities, limitations, and the way the mappings are generated.

7.1 User Interface of the J3DPanel

The J3DPanel consists of three parts. The main canvas which is a perspective view of the scene, the top view, and the control panel.

7.1.1 The Main Canvas

Here the data is shown in a perspective view of the glyphs, placed depending on their extrinsic values in the 3D space. For better orientation a grid and the x, y, and z axis are shown to indicate the origin of the 3D world. Clicking onto the canvas activates the mouse behaviors and the scene can be rotated, translated, and zoomed. Also the pick behavior is then available and with this behavior a glyph can be selected. A selected item changes its appearance to give visual confirmation of successful picking. The behavior sends the object data from the selected glyph to the callback if it was setup by the programer.

7.1.2 Top View

The top view is a simultaneous view of the scene which allows the user to see the main mappings immediately and always from the same position. This view can be translated to the left, right, up, and down with the arrow keys of the keyboard. Looking at the top view gives a feeling about the distribution of the data depending on the extrinsic mappings. Together with the glyph's color the users perceive this view more like an image of the whole, whereas the main canvas is to see individual glyphs.

7.1.3 Control Panel

The control panel has seven buttons and one slider. With the slider the users can scale the glyphs if they need a larger or smaller representation. A programer can setup the initial size factor by using the `J3DPanel.setSizeFactor (int value)` method.

The buttons are to control the main canvas just like the mouse, but each button is limited to one transformation. The buttons have two functions: clicking once will set the selected transformation and holding the button down will repeatedly apply the transformation until the button

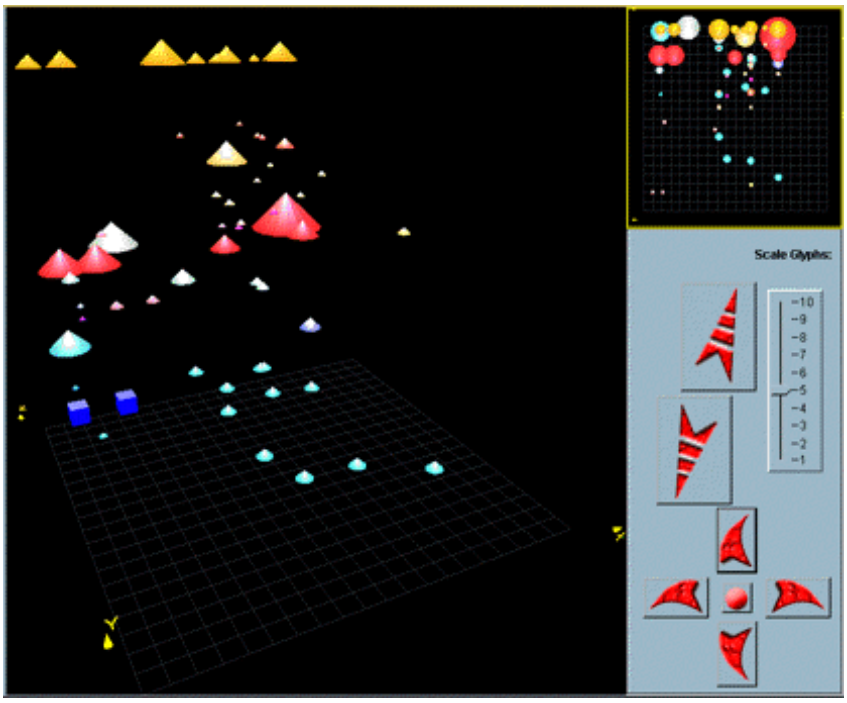


Figure 7.1: The entire J3DPanel.

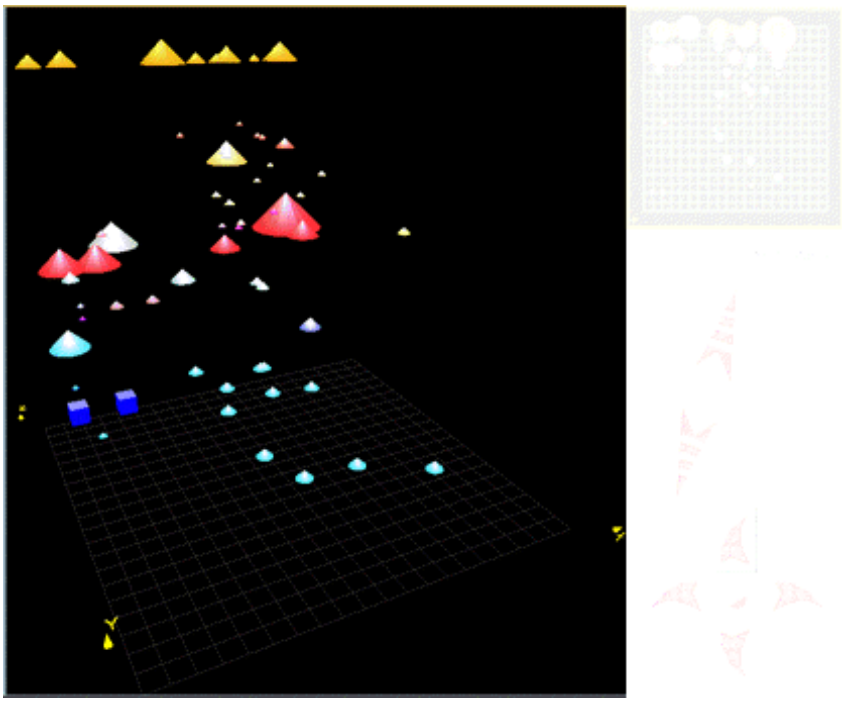


Figure 7.2: The main canvas.

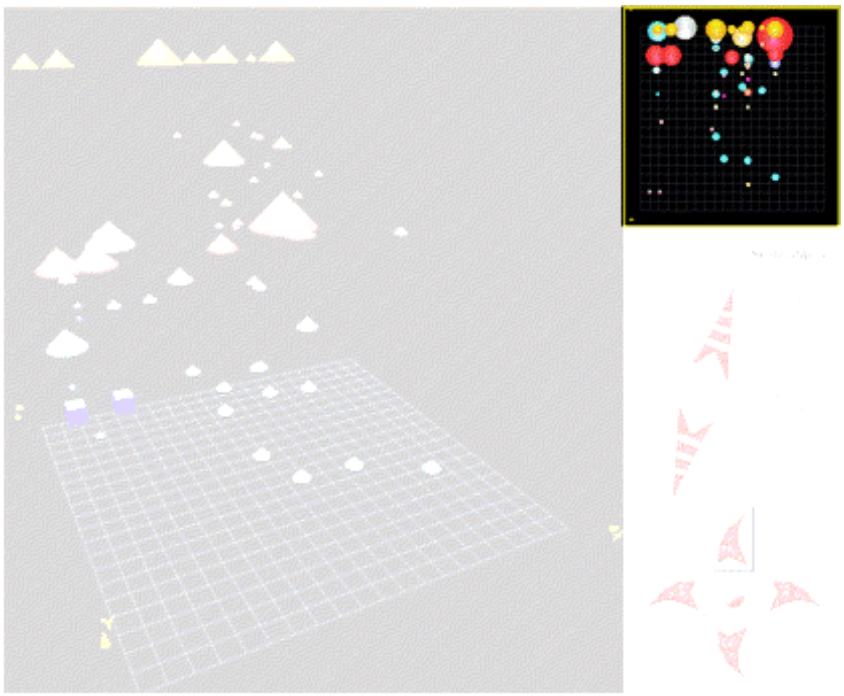


Figure 7.3: The top view.

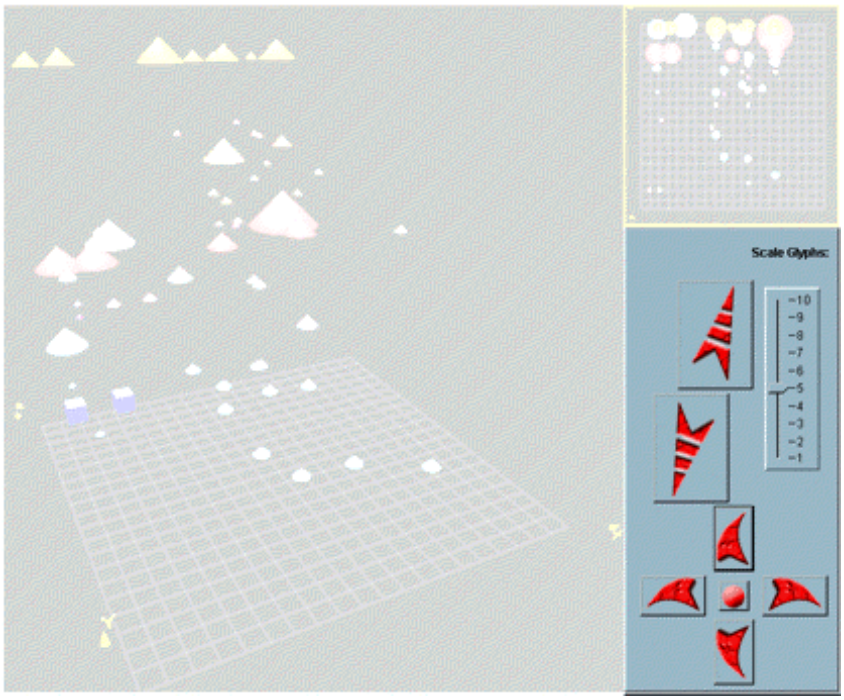


Figure 7.4: The control panel.

is released. Rollover images are implemented so the user can see which button is pressed or held. The upper two buttons are for zooming in and out the four curved arrows below are to turn the scene around the x- or y-axis. The ball in the center is to reset the view to the origin of the 3D-world.

7.2 Java3D Scene Graph

As the scene graph (see Figure 7.5) shows there is a part for controls and views and a detach-attach node where the glyphs and their transformations are added. In the this version of Java3D is it necessary to add a "virtual root node" between the scene and the branchgroup node where the scene will be detached-attached.

In the current implementation of Java3D the detach method is not very stable when the `ScreneGraph` is built up in a JPanel. For future versions of Java3D the following lines of code should be verified and possibly changed.

The normal way to detach and attach a Node is:

```
bgDetach.addChild(objRoot2);
objRoot2.addChild( drawIt());
```

The way it has to be done in the moment is:

```
objRoot2.detach();
objRoot2=null;
objRoot2 = drawIt();
bgDetach.addChild(objRoot2);
```

7.2.1 The SixDimensions Vector

```
public SixDimensions
   (Object inObject, String inGeometry, float[] inData, Color3f inColor3f )
```

This is the SixDimensions object from which the J3DPanel obtains the values to show the glyphs. The first parameter is a `java.Object.` Here the programers can attach any object to a glyph, this will be the object the mouse callback sends back when the glyph is picked. The string `inGeometry` allows to select which object will be drawn. In the moment only `CONE` and `BOX` are implemented. Following is a float array where the programer sets the six dimensions and finally a color can be defined externally for the glyphs.

The six dimensions are:

- Dimension 1: X-Axis (extrinsic)
- Dimension 2: Y-Axis (extrinsic)
- Dimension 3: Z-Axis (extrinsic)
- Dimension 4: Glyph width (intrinsic)
- Dimension 5: Glyph height (intrinsic)
- Dimension 6: Glyph color (intrinsic)

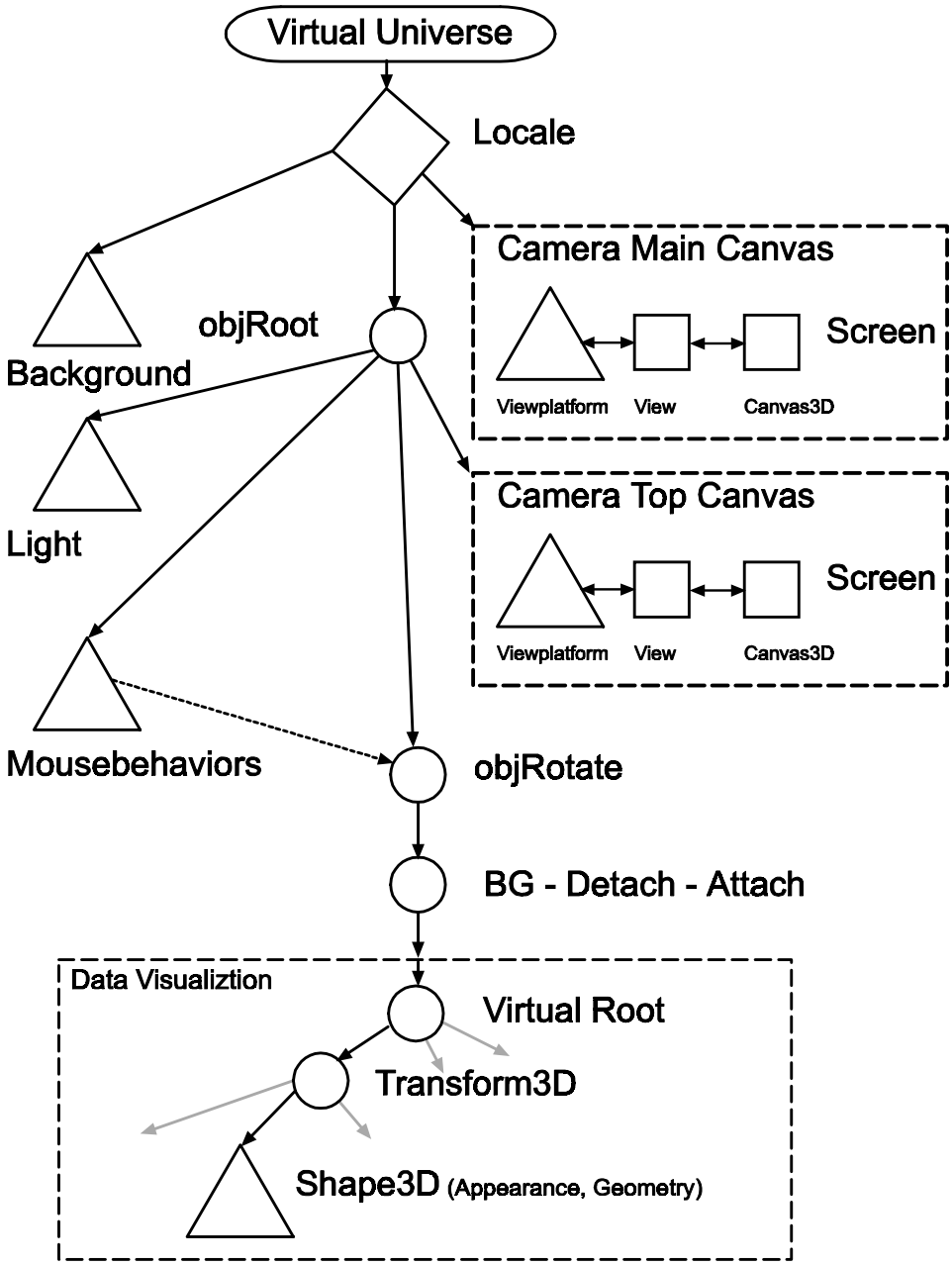


Figure 7.5: Java3D scene graph of the J3DPanel.

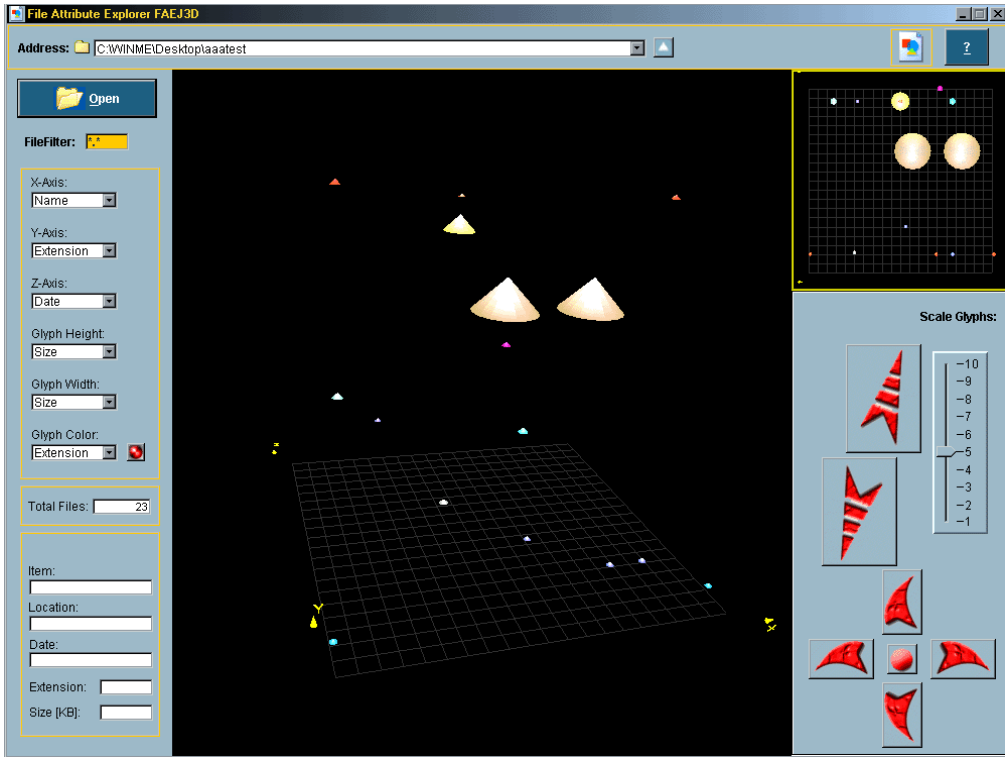


Figure 7.6: The FAEJ3D User Interface.

7.3 Limitations

In the current implementation of Java3D and Java Swing the Java3D community at j3d.org advises against using Java3D behaviors with the update of Swing elements because of thread problems. To avoid this and keep thread security an `INVOKELATER` method is used but this can sometimes cause picking not to be passed to the evaluation fields, if the user clicks too fast.

For the moment, users must click slowly so Swing passes the mouse pressed and mouse released events correctly and avoids loosing returned object data.

7.4 User Interface of the FAEJ3D

The FAEJ3D is an application that uses the J3DPanel as its visualization tool. FAEJ3D recursively loads files and directories from a selected starting directory. To select the starting point a standard jFilechooser is used.

The user has to click on the Open Button (see Figure 7.7) to bring up the Filechooser. If the search should be limited to a special type of files a file filter can be set. The file filter uses the convention that extensions of files are separated by a dot from the filename.

There are four ways to start a new visualization. The most common is the one described above. Another is to double click on a displayed directory which leads into a new scene starting now from this directory. The third way is to use the address combo box where previously visited directories are saved. (see Figure 7.8). Finally, an existing visualization can be changed if the file filter is changed.

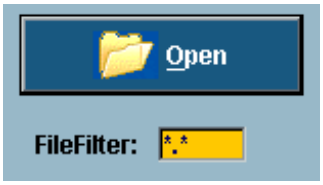


Figure 7.7: FAEJ3D Open button and Filefilter field.

Figure 7.8: The address bar.

7.4.1 Mappings

Figure 7.9 shows where the users can set which metadata they want to map onto extrinsic and intrinsic dimensions. The first three combo boxes allow the users to map: "None", "Extension", "Name", "Size", or "Date" onto the coordinates of the 3D world. Glyph dimensions are limited to "Size" or "Date". Color again can be mapped to everything apart from "None".

Mappings are generated in a way to fit the demands of the J3DPanel of closed intervals of [0..100] as follows:

- Extensions are mapped in a natural alphabetic order so Name A will be mapped to 0 whereas Name ZZZ will be mapped to 100.
- Names are treated the same way and are mapped in a natural alphabetic order.
- Size looks for the smallest and largest files in the loaded files and stretches the mapping so that the smallest is mapped to 0 and the biggest to 100.
- Date works the same way only that the algorithm looks for the youngest and oldest file.

To the right of the color mapping combo box is a button which leads the users to the color option dialog. Since extensions are discrete, the color mapping is also discrete and is generated during file loading.

Colors are generated in a way that only the first mapping (usually <no extension>) has a "strong" perceived color, whereas the remaining have pastel colors so that when users set a new color by clicking on the predefined color they can see it immediately in the 3D Scene. This is because normally basic colors like red, blue or yellow are selected for the wanted extensions. Selection of the colors is done with the `JColorChooser` which has a swatches, a HSB, and RGB representation of colors (see Figure 7.11).

If the users select for example "Date" as the metadata for color then they have the possibility to choose between several gradients as shown in Figure 7.12. The mapping works in the way that (e.g. "Date" as metadata) the youngest file is given the color at the left side of the gradient and the oldest the right most color. Hue, saturation, and value are the known variables from the HSV color model. The others are color scales which are often used for data visualization [Levk92]. All of these color scales are perceptually linearized:

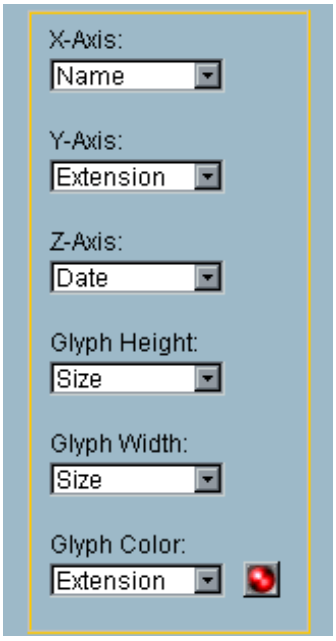


Figure 7.9: The mapping panel.

Figure 7.10: Dialog for discrete color mapping.

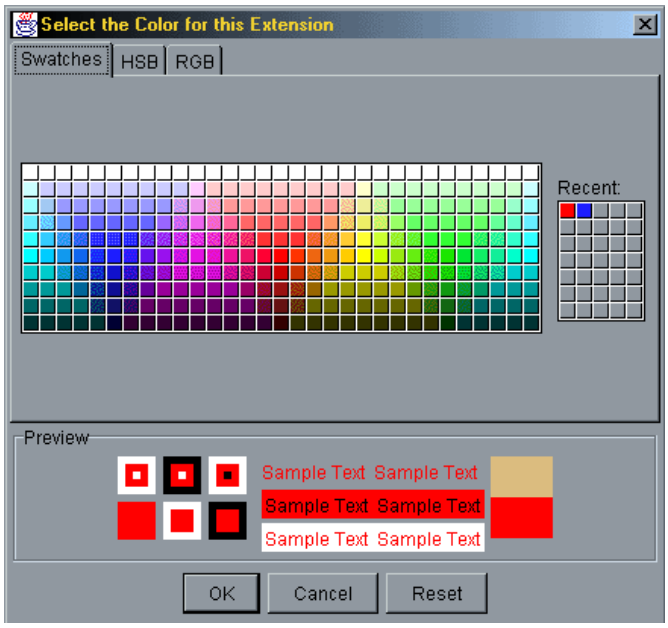


Figure 7.11: The JColorChooser.

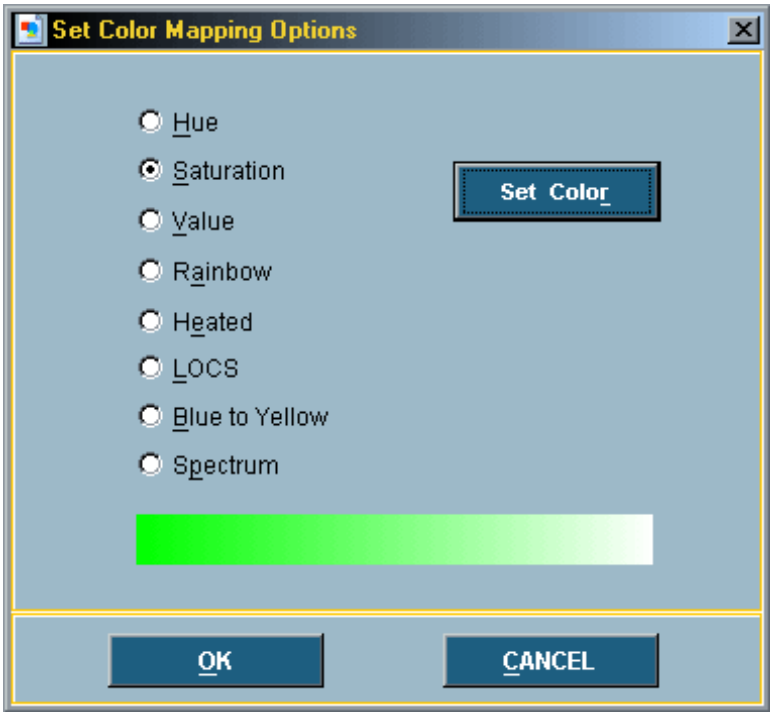


Figure 7.12: The color options dialog for gradients.

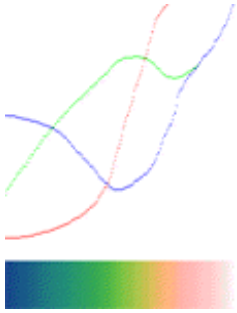


Figure 7.13: Rainbow colors.

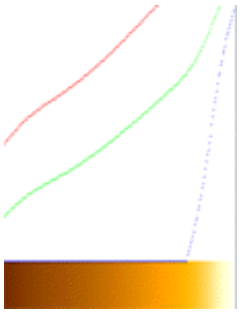


Figure 7.14: Heated colors.

- The hue interval [0..100] is mapped to 0° to 360°.
- Saturation metadata is mapped onto the saturation of a pre-selected color (`Set Color Button`). This results in a gradient from the selected color to white.
- Value as above, only that the resulting gradient goes to black.
- Rainbow maps the color to a spectrum like scale and uses our knowledge of real rainbows.
- Heated-objects also uses knowledge about heated metal which glows depending on the temperature.
- Linear optimal color scale LOCS. Every color is perceived lighter than the one before.
- Blue to yellow was implemented for color blind users to have one gradient completely without red and green.
- Spectrum is a approximation of the physical color spectrum.

7.4.2 Item Count

Figure 7.18 shows detail from the FAEJ3D application. After loading data this field displays how many files are loaded. For Java programs directories are also files and so are counted here as well.

7.4.3 Pick Results

The pick result panel (see Figure 7.19) is the place where file details will be displaced and also for some predefined file types, an icon or a standard icon. After picking one glyph the object data is received and displayed here (see Figure 7.20)

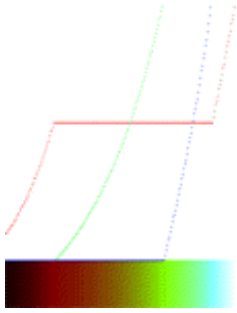


Figure 7.15: Linear optimal colors.

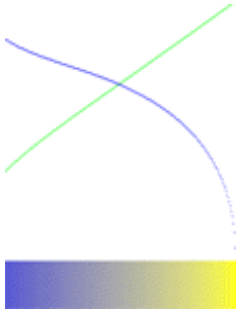


Figure 7.16: Blue to yellow colors.

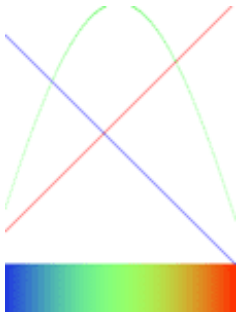


Figure 7.17: Spectrum colors.

Figure 7.18: Total files.

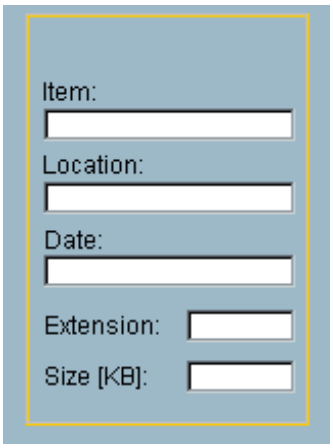


Figure 7.19: Pick results before selecting a glyph.

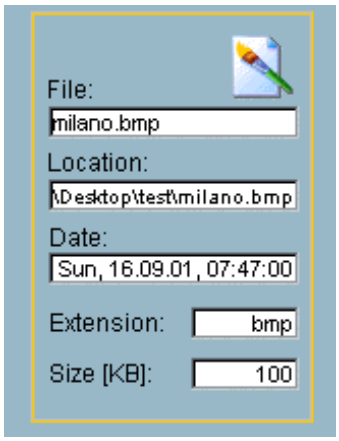


Figure 7.20: Pick results after selecting a glyph.

7.4.4 Address Bar

Figure 7.8 shows the address bar and the up button. Whenever a directory is loaded it will be added to the combo box. Selecting a directory then reloads this directory. Clicking the up button goes to the previously loaded directory.

7.4.5 Info and Helpdialog

To right of the address bar are two buttons (see Figure 7.21) one is the info button and the other one the help button which is indicated by the "?" character. Clicking them opens two different dialogs. The first of these two dialogs show some information about the program like copyright and the second a short help text (see Figures 7.22 and 7.23).

Figure 7.21: Help and info buttons.

Figure 7.22: The info dialog.

Figure 7.23: The help dialog.

Chapter 8

Outlook and Future Work

At the end of 2001 there will be a new version of Java (V1.4) and also a new version of Java3D (V1.3). If these versions have more thread security between Swing and Java3D as discussed in the beta version the "lazy" object picking could be replaced and the user would not be forced to do "slow" clicking.

If source code for all of Java3D and in particular the rendering thread were released, it would be possible to set up a progress monitor during rendering, which is essential for good human computer interaction to give the user the feeling that the program still works. At the moment only changing the mouse cursor is implemented.

Graphics hardware is becoming quicker and cheaper the same time as the rest of computer hardware and the programer can soon assume a good transform and lighting processor in every computer system. This means that instead of a single glyph for one item, more complicated glyphs could be used.

In the current implementation texture was not used to keep the number of loadable items high. With better graphics cards it would have been also possible to have texture to perceive additional dimensions of metadata.

Chapter 9

Concluding Remarks

This thesis began with an introduction to the field of human-computer interaction and a description of the eye and the visual pathway in Chapter 2. Chapter 3 discussed in detail the phenomena of visual perception and why images are perceived different from their "true" physical appearance. Chapter 4 is an overview about information and data visualization techniques. Goals were defined and it was argued how visualization systems could be categorized. For all off these categories there have been given examples from research and commercial visualization systems. Chapter 5 presented the programing language Java and the multimedia extension to it Java3D. The Java3D scene graph concept and the way Java3D interacts with the current computer hardware was described. The File Attribute Explorer in Java3D in Chapter 6 is the program which was made to implement the theory from the previous chapters. Chapter 6 gave a description about the design principles and the main classes that were used to make the FAEJ3D. Selected details of the implementation were then explained in Chapter 7. The user interface and all components were described.

The thesis concluded with an analysis of current trends, an outline of work currently in progress, and some ideas for future research.

Appendix A

FAEJ3D User Guide

To run FAEJ3D it is necessary to have Java and Java3D installed. FAEJ3D was programed using the JAVA2 JDK in the Version 1.3.1 and Java3D 1.2.1. these and newer versions of the runtimes can run the program using the following command line:

java -Xmx128m -jar faej3d.jar

This command reserves 128 MB of virtual memory for the FAEJ3D to avoid potential memory overflow problems already with less then 4000 objects loaded in the J3DPanel.

To make a new visualization the users has to load data into the FAEJ3D application. This is done be pressing the Open button (see Figure A.1). A file filter can be set before or after loading the files. In the file chooser dialog the users can select the start directory from which they want to load files.

The next step is to set the mappings (see Figure A.2) as desired depending on the goal the users have. Color gradients can be set by clicking the red ball close to the Glyph Color combo box. Depending on the selected mapping "Extension" or "Name", "Date" or "Size" a dialog pops up and lets the users set the color or the color gradient the user wants to visualize.

Now the user sees the glyphs in the J3DPanel placed in 3D space depending on the mappings and colored as defined. Clicking a cone sends back the object's metadata which will be displayed on the left side of the application the pick result panel (see Figure 7.20). Clicking on a box sends also back the object data, but with a doubleclick the user will enter this directory and get a new visualization with the same mappings and transformations like before but now with this directory as starting point.

To demonstrate the functionality of the FAEJ3D some screen shots were made. In these screen shots different mappings can be seen to get an overview of the data and to find a single object.

The first screen shows the FAEJ3D project directory itself (see Figure A.3). The mapping is set to get an overview over the whole data and for test reasons the color mapping and the x-axis are set to "Name" so the color gradient is seen immediately. If the users want to look for the different filetypes and they do not care much for the filename the settings can be changed as in

Figure A.1: The open button.

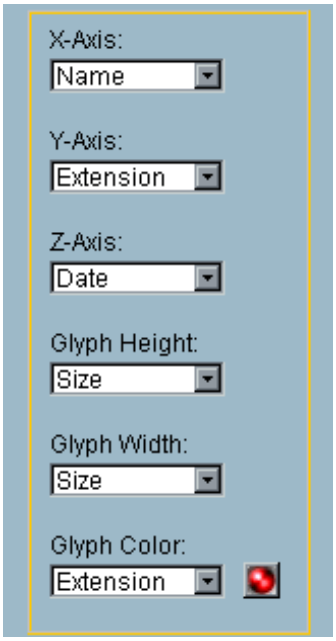


Figure A.2: The mapping panel.

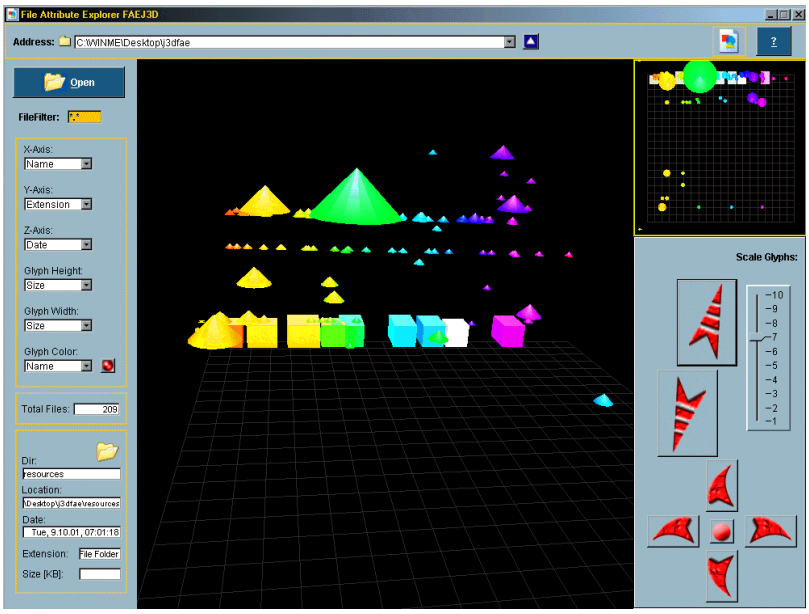


Figure A.3: FAEJ3D screen shot shows the start-up for the FAEJ3D distribution directory.

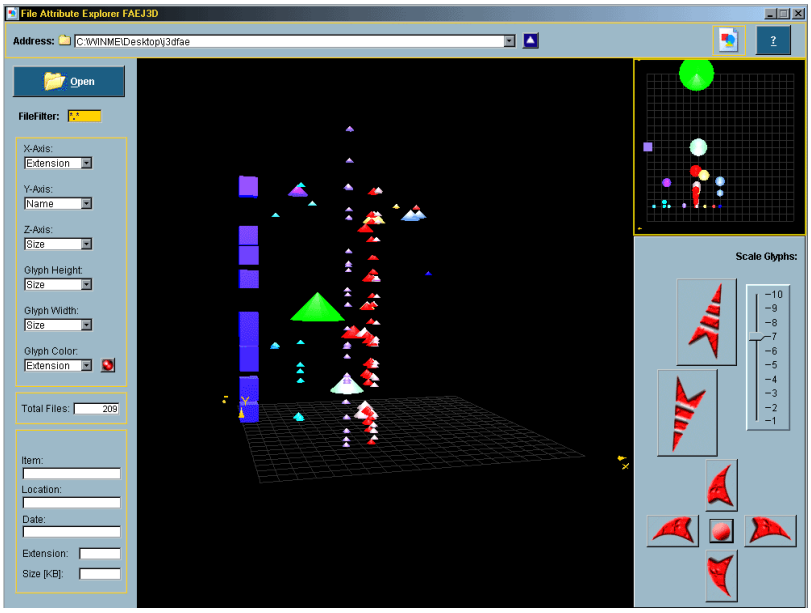


Figure A.4: Mappings are now set to see the different file types.

Figure A.4. Setting the x-axis to "Extension" and the color mapping also to color, then we set the color of the "JAR" extensions to a light green and now the users can see that they have five big piles which means five different types of files (here: java, class, html, gif, and directories)

Figure A.5 shows 610 files with the color mapped to name. Since they have almost all the same extension and the only one with a different extension pops out visually since it is placed much higher in the 3D world when the y-axis mapping is set to extension. With the same data loaded Figure A.6 helps to find the largest files. It can also be seen in both figures that the files have, similar creation dates (apart from one) when the user looks at the top view.

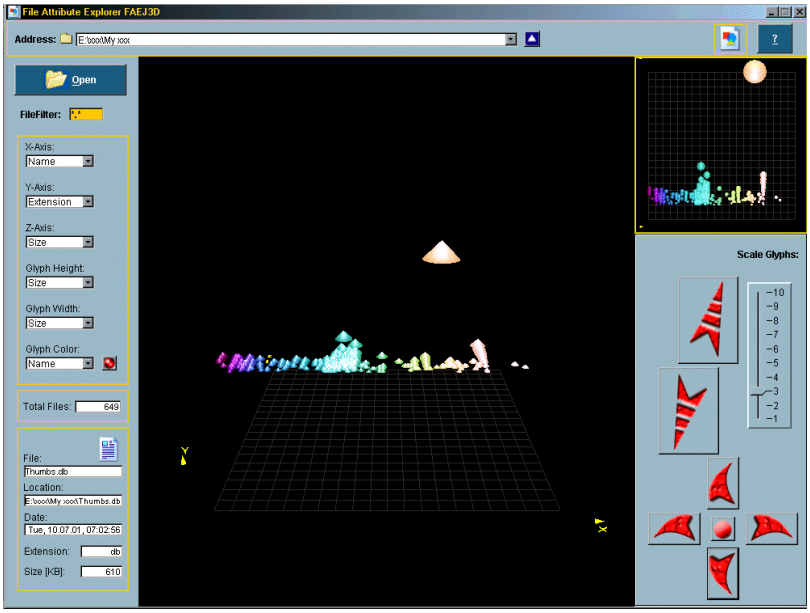


Figure A.5: Color mapping on extension.

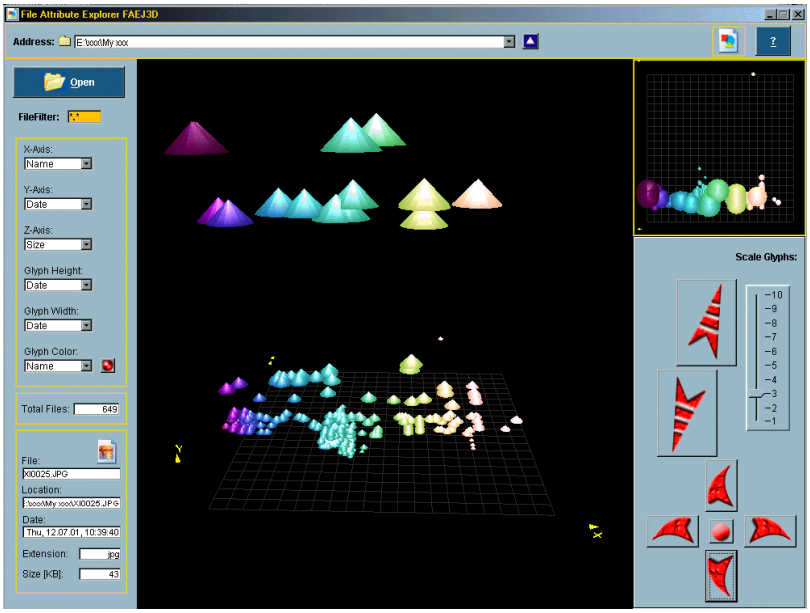


Figure A.6: Mappings set to find the largest files.

Bibliography

[ACM92] ACM SIGCHI (1992), Curricula for Human-Computer Interaction, ACM Press, ISBN 0-89791-474-0, `http://sigchi.org/cdg/cdg2.html`

[ANSI88] American National Standard for Human Factors Engineering of Visual Display Terminal Workstations. ANSI/HFS 100-1988.

[Andr95] Andrews (1995), Visualising Cyberspace: Information Visualisation in the Harmony Internet Browser; Proc. of IEEE Symposium on Information Visualization (InfoVis 95), Atlanta, Georgia.

[Barr01] Barrilleaux(2001), 3D User Interfaces with Java3D, Manning Publications Co., ISBN 1-8847-7790-2.

[Bert81] Bertin(1981), Graphics and Graphic Information Processing, de Gruyter Berlin, ISBN 3-1100-6900-8.

[Bene91] Benedikt (1991) Cyberspace: Some Proposals, In Cyberspace: First Steps, MIT Press, pp. 273–302.

[Benf95] Benford, Snowdon , Greenhalgh et al. (1995), VR-VIBE: A Virtual Environment for Co-operative Information Retrieval, Eurographics'95, pp. 349–360.

[Benf94] Benford, Mariani (1994), Populated Information Terrains: Virtual Environments for Sharing Data, Research report CSCW/4/1994, Centre for Research in CSCW, Lancaster University.

[Bonf99] Bonfantini, Proni, Brunelli, Ferraresi(1999), Segni sui corpi e sugli oggetti, Quaderni di Ergonomia 1, Moretti e Vitali (in Italian), ISBN 8-8718-6124-8.

[Bray96] Bray (1996), Measuring the Web, Fifth International World Wide Web Conference Paris France, Open Text.

[Chal92] Chalmers, Chitson (1992), Bead: Explorations in Information Visualisation, Proceedings of SIGIR92, published as a special issue of SIGIR forum, ACM Press, pp. 330–337.

[Card91] Card, Mackinlay, Mackinlay (1991), Cone Trees: Animated 3D Visualizations of Hierarchical Information, Proceedings of the ACM SIGCHI 91 Conference on Human Factors in Computing Systems, pp. 189–194.

[Card99] Card, Mackinlay, Shneiderman (1999), Readings in Information Visualization - Using Vision To Think, Morgen Kaufmann Inc., ISBN 1-5586-0533-9.

[Chan01] Chandler (2001), Semiotics: The Basics, Routledge 2001, ISBN 0-4152-6594-0.

[Cher73] Chernoff, (1973).. The use of faces to represent points in k-dimensional space graphically. Journal of the American Statistical Association, Vol. 68, pp. 361–368.

[CLOW99] ClockWise3d, `http://www.clockwise3d.com/`

[Cone91] Robertson, Mackinlay, and Card. (1991). Cone trees: Animated 3D visualizations of hierarchical information. ACM Conference on Human Factors in Computing Systems (CHI '91), ACM: pp. 189–194.

[Crap00] Crapo,Waisel,Wallace, (2000) Proceedings of the sixth ACM SIGKDD international Conference on Knowledge Discovery and Data Mining, pp. 218–226.

[Chua95] Chuah and Roth, (1995) SDM: Selective Dynamic Manipulation of Visualizations. In Proc. of UIST '95, `http://citeseer.nj.nec.com/chuah95sdm.html`

[Fair93] Fairchild (1993), Information Management Using Virtual Reality-Based Visualizations, in "Virtual Reality: Applications and Explorations", Alan Wexelblat (ed.), Academic Press Professional, Cambridge, MA, ISBN 0-12-745045-9, pp. 45–74.

[Fein90] Feiner et al. (1990), Worlds within Worlds Metaphors for Exploring n-Dimensional Virtual Worlds, Proc. UIST 90 (ACM Symp. on User Interface Software and Technology), Snowbird, UT, October 3-5, 1990, pp. 76–83.

[Furn86] Furnas (1986), Generalized fisheye views, In Proceedings of the ACM SIGCHI 86 Conference on Human Factors in Computing Systems, pp. 16–23.

[Fior98] Fiorani (1998) Grammatica della comunicazione, Lupetti (in Italian), ISBN 8-8870-5853-9.

[Fole90] Foley, van Dam, Feiner (1990), Computer graphics principles and practice, Addison-Wesley, ISBN 0-201-84840-6.

[Gibs79] Gibson (1979), The Ecological Approach to Visual Perception, ISBN 0-8985-9959-8.

[Greg90] Gregory (1990) Eye and Brain the Psychology of Seeing, Oxford University Pres INC. New York 4th Edition, ISBN 0-19852-340-8.

[Hemm94] Hemmje, Kunkel and Willet (1994), LyberWorld - A Visualization User Interface Supporting Fulltext Retrieval, In: Proceedings of ACM SIGIR '94.

[Hend95] Hendley, Drew, et al. (1995), Narcissus: Visualising Information, University of Birmingham.

[Holz00a] Holzinger (2000) Basiswissen Multimedia Band 1 Technik, Vogel Fachbuch (in German) ISBN 3-8023-1856-0.

[Holz00b] Holzinger (2000) Basiswissen Multimedia Band 2 Lernen, Vogel Fachbuch (in German) ISBN 3-8023-1857-0.

[Holz00c] Holzinger (2000) Basiswissen Multimedia Band 3 Design, Vogel Fachbuch (in German) ISBN 3-8023-1858-0.

[HHMI01] Huges Medical Institue, `http://www.hhmi.org`

[Hump00] Humphrey,(2000) Creating Reusable Visualizations with the Relational Visualization Notation, Proceedings of the IEEE Symposium on Information Visualization 2000, ISBN 0-7695-0804-9.

[ID301] The ID3 Tag, `http://www.id3.org/`

[ISO19] ISO (1998). ISO 9241-5. Ergonomic requirements for office work with visual display terminals (VDTs) Part 5: Workstation layout and postural requirements.

[Java01] Java3D API Specification, Sun Microsystems, `http://java.sun.com`

[Keim00] Kaim, (2000), An Introduction to Information Visualization Techniques for Exploring Large Databases, Institute for Computer Science University of Halle, `http://www.informatik.uni-halle.de/~keim`

[Kroh96] Krohn (1996),VINETA: Navigation Through Virtual Information Spaces, in "AVI: Advanced Visual Interfaces", Ed. T. Cartaci, Gubbio, Italy, ACM.

[Levk92] Levkowitz Haim and Gabor, (1992), Color scales for image data. IEEE Computer Graphics and Applications.

[Luck98] Luck, Ford (1998), On the role of selective attention in visual perception, Proc. Natl. Acad. Sci. USA Vol. 95, pp. 825–830, The National Academy of Sciences 0027-8424/98/95825-6, ISSN 0027-8424.

[Marr82] Marr, (1982). Vision: A computational investigation into the human representation and processing of visual information. San Francisco: W.H. Freeman and Company., ISBN 0-7167-1567-8.

[Mack91] Mackinlay, Card and Robertson (1991), Perspective Wall: Detail and Context Smoothly Integrated, In Proceedings of the ACM SIGCHI 91 Conference on Human Factors in Computing Systems, pp. 173–179, New Orleans, LA, USA.

[MAPN01] Map.Net., `http://www.map.net/`

[Maur96] Maurer, Scherbakov (1996), Multimedia Authoring for Presentation and Education: The Official Guide to HM-Card, Addisson-Wesley Publ.Co. Bonn, ISBN 3-89319-928-4.

[Mina12] Minard, Carte figurative des pertes successives en hommes de l'armée française dans la campagne de Russie, 1812-1813. lith. (624 x 207, 624 x 245), 20 November 1869. ENPC: Fol 10975, 10974/C612; Tufte(1983, p. 176). [Annibal Napoléon Espagne Italie Russie armée flow-map comparison.].

[Munz96] Munzner, Hoffman, Claffy, Fenner (1996): Visualizing the Global Topology of the mbone, Proc. Symp. on Information Visualization.

[Narc95] Hendley, Drew, Wood, and Beale, (1995),. "Narcissus: Visualising Information", Proceedings of the IEEE Symposium on Information Visualization, IEEE CS Press, pp. 90–96.

[NEIN01] National Eye Institute, `http://www.nei.hih.gov`

[Norm99] Norman, (1999). Affordance, conventions, and design. Interactions 6, 3 (May. 1999), pp. 38–43.

[Open97] OpenGL ARB (1997), OpenGL Programming Guide, Addison-Wesley ISBN 0-2016-0458-2.

[Qpit94] Colebourne, Mariani, Rodden, (1994), Populated Information Terrains: supporting cooperative browsing of online information (CSCW/13/94), `http://www.comp.lancs.ac.uk/computing/users/jam/qpit_paper/home.html`

[Pers91] Mackinlay, Robertson, and Card. The perspective wall: Detail and context smoothly integrated. In Proceedings of CHI '91, pp. 173–9, New Orleans, LA, 1991.

[Raub99] Rauber (1999), The SOMLib Digital Library System Proceedings of the 3rd Europ. Conf. onResearch and Advanced Technology for Digital Libraries (ECDL'99), Paris, France,Lecture Notes in Computer Science (LNCS 1696), Springer.

[Rutg01] Ruttiger, Mayser, Serey (2001), The color constancy of the red-green color blind, Color-Research-&-Application. vol.26, suppl.issue; 2001; pp. 209–213.

[Shne96] Shneiderman, (1996), The eyes have it: A task by data-type taxonomy for information visualizations. In Proceedings of Visual Languages (Boulder, CO, Sept. 36). IEEE Computer Science Press, Los Alamitos, CA.

[Snel98] Snell (1998), Clinical Anatomy of the Eye, Blackwell Science Inc, ISBN: 0-6320-434-4.

[Sowi97] Sowizral, Rushforth, and Deering (1997), The Java3D API Specification, Addison-Wesley, Reading, Mass., 1997. ISBN 0-201-32576-4.

[Stro91] Stroustroup (1991), The C++ programming language, Addison Wesley, ISBN 0-2015-3992-6

[Thal95] Washington University St.Luis, `http://thalamus.wustl.edu/`

[Toot98] Tootell, Hadjikhani, Vanduffel, Liu,Mendola, Sereno, and Dale (1998), Functional analysis of primary visual cortex (V1) in humans. Proc. Natl. Acad. Sci. USA Vol. 95, pp. 811–817, February 1998, The National Academy of Sciences 0027-8424/98/95825-6, ISSN 0027-8424.

[Tuft82] Tufte (1982), The Visual Display of Quantitative Information, Graphics Press, ISBN: 0-9613-9214-2.

[UBCO99] University of British Columbia, `http://www.geog.ubc.ca`

[VISC01] Vischeck color vision simulator, `http://www.vischeck.com`

[Walk95] Walker (1995), Challenges in Information Visualisation, British Telecommunications Engineering Journal, Vol. 14, pp. 17–25.

[Wand95] Wandel (1995) Foundations of Vision, Sinauer Associates INC. MA, ISBN 0-87893-853-2.

[Ware00] Ware (2000), Information Visualization : Perception for Design, Morgan Kaufmann Publishers; ISBN: 1-5586-0511-8.

[Wilt99] Wiltgen (1999), Digitale Bildverarbeitung in der Medizin, Shaker Verlag GmbH; (German) ISBN: 3-8265-6212-7.

[YORK01] Universitè York, `http://www.yorku.ca/`

[Youn96] Young (1996), Report: Three Dimensional Information Visualisation, `http://vrg.dur.ac.uk/misc/PeterYoung/pages/work/Documents/lit-survey/IV-Survey`