

Geon Diagrams: a Perception Based Method for Visualizing Structured Information

By

Pourang Polad Irani

BCS, University of New Brunswick, 1997

A Thesis Submitted in Partial Fulfillment of
the Requirements for the Degree of

Doctor of Philosophy
In the

Graduate Academic Unit of Computer Science

Supervisors:	Colin Ware, Ph.D., Computer Science Jane Fritz, Ph.D., Computer Science
Examining Board:	Bernd Kurz, Ph.D., Computer Science, Chair Przemyslaw Pocheć, Ph.D., Computer Science Daniel Voyer, Ph.D., Psychology Lloyd Waugh, Ph.D., Civil Engineering
External Examiner:	John Dill, Ph.D., Simon Fraser University (CS)

This thesis is accepted.

Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

July 2002

© Pourang Polad Irani, 2002

Abstract

Abstract information is commonly structured, presented and communicated using diagrams. One particular type of diagram that has played a predominant role in the Information Sciences is the node-link diagram. Nodes that represent entities, objects, or units and links that represent relationships of various kinds between the nodes characterize this type of diagram. Visually these nodes are represented using outline forms such as boxes and circles and the links have been depicted with lines consisting of different characteristics.

Recent advances in perception present evidence that our visual system is endowed with the capability of recognizing objects from their structural composition. This structure is inherently 3D in nature. This thesis investigates methods for applying these theories to creating diagrams that are more expressive tools for communicating abstract information. A set of rules for creating diagrams is abstracted from vision research and collectively these define a new kind of diagramming convention called a *Geon Diagram*.

A series of experiments was carried out to evaluate the effectiveness of *Geon Diagrams*. The results indicate that applying theories of structural object recognition to the construction of node-link diagrams can facilitate identification, recall, and understanding of diagram structures. The benefits are significant in comparison to diagrams created with box-and-line notations, or diagrams with elements consisting of 2D solid outline shapes. The results show that a diagrammatic notation based on perceptual semantics, such as those available from structural object recognition theories, can facilitate intuitive comprehension of abstract problem descriptions by both novice and expert users in a given problem domain.

Acknowledgements

I am tremendously grateful to my supervisors Colin Ware and Jane Fritz for being advisors and mentors for the duration of my thesis work. I am grateful for the numerous hours Colin spent copiously revising my thesis, his patience with my errors, and his support of my ambitions. I am thankful for Colin's commitment to my research after his departure from the University of New Brunswick. Jane's encouragement and support of my research were far more than I expected. Her insightful suggestions were key in many facets of my thesis.

I could not have completed my thesis without the help of others. In particular, I am grateful to Maureen Tingley who generously spent time on the statistical analyses in my thesis. I am also grateful to Richard Spacek who carefully proofread my papers. I am thankful to the staff in the Computer Science Faculty for their help and cooperation, in particular Linda Sales.

I am greatly indebted to my parents Polad and Mani Irani who have guided me in every step of my education. I am undeserving of the many sacrifices they have forgone to ensure that I be adequately schooled. They both have showered me with care, encouragement, and affection. I am grateful to my brother-in-law Judah Bunin for his insightful comments on my research and for proof reading papers and various sections of my thesis. I am thankful to my sisters Parisa and Paricher who have blessed me with their care and consideration throughout my studies. I am also indebted to my little nephew Liam who has been a source of great happiness.

I have enjoyed the friendly and stimulating environment during the project. I am thankful to my fellow lab workers and graduate students Roland Arsenault, Oleg Goloubitsky, Dimitry Korkin, Nhu Le, Jordan Lutes, Irina Padioukova, Graeme Sweet, and Natalie Webber for their help and friendship. I am also grateful to all the subjects who, without hesitation, participated in my experiments.

I am thankful to my friends, near and far, who have constantly provided me their sincere support, encouragement and friendship at times when these were most needed. I hope to repay in kind if they allow me to do so.

I am grateful to NSERC for supporting my thesis work with NSERC PGS-A and PGS-B scholarships.

Contents

Abstract	ii
Acknowledgements	iii
List of Tables	xii
List of Charts	xiii
List of Figures	xiv

Chapter 1 - Introduction	1
1.1 Goals and Approach	3
1.2 Organization of the Thesis	3
 Chapter 2 - Visualizations of Structured Information	 6
2.1 What is Structured Information?	7
2.1.1 Lists	7
2.1.2 Hierarchies	8
2.1.3 Networks	8
2.1.4 Entity-Relationships	8
2.2 Requirements of Visualization Techniques	9
2.2.1 Adequate Representation	10
2.2.2 Minimize Distortion	10
2.2.3 Semantic Representation	11
2.2.4 Abstractions	11
2.2.5 Interaction	11
2.2.6 Navigation	11
2.2.7 Space	12
2.2.8 Scaling	12
2.2.9 Visual Quality	12
2.2.10 Integration	13
2.3 Display Techniques	13

2.3.1 Displaying List Information	13
2.3.1.1 Time lines	13
2.3.1.2 SeeSoft	14
2.3.1.3 Lifestreams	15
2.3.1.4 TimeTube	16
2.3.2 Displaying Hierarchies	17
2.3.2.1 Tree-Views	18
2.3.2.2 Cone Trees	19
2.3.2.3 Hyperbolic Browser	20
2.3.2.4 Cheops	21
2.3.2.5 Tree Maps	23
2.3.2.6 Information-Slices	24
2.3.3 Displaying Networks	25
2.3.3.1 NicheWorks	27
2.3.3.2 SeeNet	28
2.3.3.3 Harmony Local Map	29
2.3.3.4 SemNet	29
2.3.3.5 NV3D	31
2.3.3.6 Narcissus	32
2.3.4 Displaying Entity-Relationship Data	33
2.3.4.1 Flowcharts	35
2.3.4.2 Data-Flow Diagram	36
2.3.4.3 Entity-Relationship Diagram	37
2.3.4.4 UML	39
2.4 Discussion	40
 Chapter 3 - Perceptual and Cognitive Issues	 42
3.1 Cognitive Theories	43
3.1.1 Metaphors	47
3.2 Graphical Features	50
3.2.1 Color	50
3.2.2 Size	51
3.2.3 Orientation	51

3.2.4 Transparency	51
3.2.5 Texture	51
3.2.6 Positioning	52
3.2.7 Remarks	52
3.3 Graphical Patterns and Principles of Gestalt	53
3.3.1 Pragnanz	54
3.3.2 Similarity	54
3.3.3 Continuity	55
3.3.4 Connectedness	55
3.3.5 Proximity	56
3.3.6 Symmetry	56
3.3.7 Closure	57
3.4 Perceptual Syntax of Node-Link Diagrams	57
3.5 Objects and Object Recognition	58
3.5.1 Image Matching	59
3.5.2 Structured Object Perception	61
3.5.3 Shape-From-Shading	65
3.6 Perceptual Studies of 3D Displays	66
3.7 Discussion	67

Chapter 4 - The Geon Diagram: Definition and Construction 69

4.1 The Geon Diagram	70
4.2 High-level Architectural Description of the Toolkit	71
4.3 Functional Description of the Toolkit	73
4.3.1 Creating Geons using Parametric Descriptions	73
4.3.2 Resizing Geons	86
4.3.3 Positioning Geons to Create Diagram Structures	86
4.3.4 Mapping Color, Texture, Transparency, and Labeling onto Geons ...	87
4.3.5 Saving and Opening Geon Diagrams	88
4.4 Geon Toolkit User Interface	88
4.4.1 Modifying Geon Sizes	88
4.4.2 Positioning Geons	89
4.4.3 Mapping Surface Properties	91

4.5 Discussion	92
----------------------	----

Chapter 5 - Comparison of Geon and UML Diagrams for Sub-structure

Identification and Recall 94

5.1 Parsing Visual Structures	95
5.2 Role of Structural Description for Recognition	97
5.3 Empirical Studies	100
5.4 Experiment 1: Sub-Structure Identification with Geon versus UML Diagrams	101
5.4.1 Task	102
5.4.2 Hypothesis	102
5.4.3 Experiment Control Software	103
5.4.4 Equipment	103
5.4.3 Method for Experiment 1	103
5.4.4 Results of Experiment 1	106
5.4.5 Discussion	108
5.5 Experiment 2: Recall of Geon versus UML Diagrams	109
5.5.1 Hypothesis	110
5.5.2 Method for Experiment 2	110
5.5.3 Results of Experiment 2	112
5.5.4 Discussion	113
5.6 Experiment 3: Recall of Geon versus UML Diagrams (without surface attributes)	113
5.6.1 Hypothesis	114
5.6.2 Method for Experiment 3	114
5.6.3 Results of Experiment 3	115
5.6.4 Discussion of Experiment 3	115
5.7 General Discussion	116

Chapter 6 - Importance of Shading 118

6.1 Shape-From-Shading	120
6.1.1 Structure-From-Shading	122
6.1.2 Shading models and parameters used for the experiments	123

6.2 The importance of shading in Geon diagrams	127
6.3 Experiment 4: Sub-Structure Identification with Geon versus 2D-Silhouette Diagrams	128
6.3.1 Hypothesis	128
6.3.2 Equipment for experiment 4	128
6.3.3 Method for Experiment 4	129
6.3.4 Results of Experiment 4	131
6.3.5 Discussion of Experiment 4	134
6.4 Experiment 5: Recall of Geon versus 2D-Silhouette	134
6.4.1 Hypothesis	134
6.4.2 Equipment for Experiment 5	134
6.4.3 Method for Experiment 5	135
6.4.4 Results of Experiment 5	137
6.4.5 Discussion of Experiment 5	137
6.5 Discussion	138

Chapter 7 - Evaluating Perceptual Semantics 139

7.1 Perceptual Semantics	140
7.2 Experiment 6: Deriving Structures using Perceptual Semantics	142
7.2.1 Generalization	145
7.2.1.1 Hypotheses for Generalization	146
7.2.1.2 Method for Generalization	146
7.2.1.3 Results for Generalization	148
7.2.1.4 Discussion of Generalization	149
7.2.2 Dependency	150
7.2.2.1 Hypotheses for Dependency	151
7.2.2.2 Method for Dependency	151
7.2.2.3 Results for Dependency	152
7.2.2.4 Discussions of Dependency	153
7.2.3 Relationship Strength	154
7.2.3.1 Hypotheses for Relationship Strength	154
7.2.3.2 Method for Relationship Strength	155
7.2.3.3 Results for Relationship Strength	156

7.2.3.4 Discussion of Relationship Strength	157
7.2.4 Multiplicity (or cardinality)	157
7.2.4.1 Hypotheses for Multiplicity	158
7.2.4.2 Method for Multiplicity	158
7.2.4.3 Results for Multiplicity	159
7.2.4.4 Discussion of Multiplicity	160
7.2.5 Aggregation	161
7.2.5.1 Hypotheses for Aggregation	161
7.2.5.2 Method for Aggregation	162
7.2.5.3 Results for Aggregation	163
7.2.5.4 Discussion of Aggregation	163
7.3 The Perceptual Syntax	164
7.4 Experiment 7: Validating the Perceptual Syntax	166
7.4.1 Hypothesis for Experiment 7	167
7.4.2 Equipment for Experiment 7	167
7.4.3 Method for Experiment 7	167
7.4.4 Results for Experiment 7	169
7.4.5 Discussion of Experiment 7	170
7.5 Experiment 8: Learnability of the Perceptual Syntax	170
7.5.1 Hypothesis for Experiment 8	171
7.5.2 Equipment for Experiment 8	171
7.5.3 Method for the Learning Phase	171
7.5.4 Method for the Matching Phase	172
7.5.5 Results for Experiment 8	174
7.5.7 Discussion of Experiment 8	175
7.6 Applying Theories of Perception to Drawing Diagrams	176
7.7 General Discussion	177

Chapter 8 - Conclusion 180

8.1 The Geon Diagram Model	181
8.2 Review of Experimental Findings	184
8.3 Major Contributions of the Thesis	186
8.4 Further Investigations	188

8.4.1 Usability and Practicality of the Geon Diagram Toolkit	188
8.4.2 Future Theoretical Investigations	190
Appendix I	194
References	201
Vita	

List of Tables

Table 5.1 - Summary of Results of Experiment 1	107
Table 5.2 - Reaction Times for 3 and 7 component diagrams	99
Table 5.3 - Summary of Results of Experiment 2	112
Table 5.4 - Summary of Results of Experiment 3	115
Table 6.1 - Summary of Results for Experiment 4	132
Table 6.2 - Reaction Times for 3 and 7 component diagrams	133
Table 6.3 - Error rates in recalling Geon diagram vs. 2D outlined-silhouettes ..	137
Table 7.1 - Comparison of error rates of Geon diagrams vs. UML diagrams	169
Table 7.2 - Error rates of matching Geon and UML diagrams to problems	174

List of Charts

Chart 5.1 - The time required for parsing a Geon or UML sub-structure.....	108
Chart 6.1 - The time required for parsing a Geon or 2D sub-structure.....	133
Chart 7.1 - Average rankings for dependency.	153
Chart 7.2 - Average rankings for relationship strength.....	156
Chart 7.3 - Average rankings for multiplicity.....	160
Chart 7.4 - Average rankings for aggregation	163

List of Figures

Figure 2.1 - Classification of the types of structured information	9
Figure 2.2 - Time line	14
Figure 2.3 - SeeSoft	15
Figure 2.4 - LifeStreams	16
Figure 2.5 - Time Tube	17
Figure 2.6 - A generic view of a tree or hierarchy	18
Figure 2.7 - Windows Explorer	19
Figure 2.8 - Cone Tree	20
Figure 2.9 - Hyperbolic Browser	21
Figure 2.10 - Cheops.....	22
Figure 2.11 - Selection in Cheops	23
Figure 2.12 - Tree map and Cushion Map	24
Figure 2.13 - Information Slices	25
Figure 2.14 - Network	26
Figure 2.15 - NicheWorks.....	27
Figure 2.16 - SeeNet	28
Figure 2.17 - Harmony Local Map	29
Figure 2.18 - SemNet	31
Figure 2.19 - NV3D	32
Figure 2.20 - Narcissus	33
Figure 2.21 - Classification of the general entity-relationship visualizations	35
Figure 2.22 - Flowchart.....	36
Figure 2.23 - Data Flow diagram	37
Figure 2.24 - ER-Diagram	38
Figure 2.25 - UML diagram.....	40
Figure 3.1 - Gestalt principle of Pragnanz	54
Figure 3.2 - Gestalt principle of Similarity	54
Figure 3.3 - Gestalt principle of Continuity	55

Figure 3.4 - Gestalt principle of Connectedness	55
Figure 3.5 - Gestalt principle of Proximity	56
Figure 3.6 - Gestalt principle of Symmetry	56
Figure 3.7 - Gestalt principle of Closure	57
Figure 3.8 - Grammar of a node-link diagram based on gestalt principles	58
Figure 3.9 - Structural object recognition vs. template recognition	59
Figure 3.10. Series of processing stages in structural object recognition	62
Figure 3.11 - Perceptual system makes assumptions that contours are connected	63
Figure 3.12 - Silhouettes are critical in defining a structural skeleton	63
Figure 3.13 - Geons and objects created with them	64
Figure 3.14 - Difference between shaded and non-shaded objects	65
Figure 4.1 - Architectural overview of the Geon toolkit	72
Figure 4.2 - Attributes of Geon cross section and axis	74
Figure 4.3 - Sample textures used in the toolkit	87
Figure 4.4 - Effect of selecting a Geon	89
Figure 4.5 - Geon properties editor	89
Figure 4.6 - Translating a Geon	90
Figure 4.7 - Rotating a Geon	91
Figure 5.1 - Additional Geons do not affect structural object recognition	96
Figure 5.2 - Impossible objects	98
Figure 5.3 - Mappings between Geon and UML	101
Figure 5.4 - Geon and UML sub-structures with 2 nodes or 3 components.....	102
Figure 5.5 - Geon and UML sub-structures with 4 nodes or 7 components	105
Figure 5.6 - An example of a Geon diagram and its UML equivalent	105
Figure 5.7 - A Geon and its equivalent UML diagram used in Experiment 2	111
Figure 5.8 - Mapping used for Experiment 3	114
Figure 5.9 - A Geon and its equivalent UML diagram used in Experiment 3	115
Figure 6.1 - 3D shaded objects vs. 2D outline shapes vs 2D outlined-silhouette	119
Figure 6.2 - Targets used in Enns and Rensink's experiments	120
Figure 6.3 - Sample targets used in the experiment by Sun and Perona	121

Figure 6.4 - Assumptions about the directions of light source	122
Figure 6.5 - Treemap vs. cushionmap structural presentations	123
Figure 6.6 - Diffuse and specular shading	124
Figure 6.7 - Specular reflection from an object surface	125
Figure 6.8 - Gouraud shading	126
Figure 6.9 - Geons without smooth shading	126
Figure 6.10 - Geons with smooth shading	127
Figure 6.11 - One-to-one mapping between Geon primitives and 2D silhouettes	130
Figure 6.12 - Example of diagrams used for Experiment 4.....	130
Figure 6.13 - Sub-structures to identify in Experiment 4	131
Figure 6.14 - Sample diagrams used in experiment 5.....	135
Figure 7.1 - Geon Structural Description (GSD) critical for identification	142
Figure 7.2 - UML representation for Inheritance	146
Figure 7.3 - Sample set for Inheritance (one component)	147
Figure 7.4 - Sample set for Inheritance (two components)	148
Figure 7.5 - Sample set for Dependency	152
Figure 7.6 - UML representation of Dependency	152
Figure 7.7 - Sample set for relationship strength	155
Figure 7.8 - UML representation for Multiplicity	158
Figure 7.9 - Sample set for Multiplicity	159
Figure 7.10 - UML representation of Aggregation	161
Figure 7.11 - Sample set for Aggregation.....	162
Figure 7.12 - Perceptual notation.....	165
Figure 7.13 - Related entities in a system describing a Space Center	166
Figure 7.14 - Geon representation for entities in a conference.....	166
Figure 7.15 - Geon diagram representation used in validating the syntax.....	168
Figure 7.16 - UML equivalent to the Geon representation of figure 7.14	169
Figure 7.17 - UML and Geon diagram for entities of a neighborhood.....	173

Chapter 1 - Introduction

Diagrams are essential in documenting large information systems. They capture, communicate and leverage knowledge indispensable for solving problems. A diagram provides a mapping from the problem domain to a visual representation by supporting cognitive processes that involve perceptual pattern finding and cognitive symbolic operations. However not all mappings are equivalent, and to be effective a diagram's representation needs to be constructed with characteristics such that meaningful patterns can be easily perceived. It is often stated that what we communicate depends on the language we use for communicating. This may be especially true for visual information display where diagrams can alleviate ambiguities that prevail in textual descriptions. Therefore good diagramming techniques are essential in providing clear visual communication.

Throughout the evolution of software engineering techniques and methods, diagrams have played a significant role in the building of systems. They serve as a medium of discourse for members on a team, such as end users, architects, and programmers. A major problem with heterogeneous teams is developing a common model of the problem domain and a language to describe it. Diagrams can help with this.

Taxonomies for classifying diagrammatic representations have organized diagrams based on their representational aspects, their structure, and their intended meaning. A general category that emerges from these classification schemes is the class of node-link diagrams. Networks of nodes and links are used to represent structured information. Node-link diagrams make up a large component of the set of visual tools used in software engineering disciplines.

Today, a myriad of graphical syntax for node-link diagrams exists, in different forms and for a variety of purposes. While each of them holds merit in their own respect, they have usually been developed as arbitrary conventions without taking into account perceptual issues. However, there is abundant evidence that perceptual factors influence how easily diagrams are understood. If these are not applied carefully, diagrams can be misleading and misinterpreted without rigorous training in their syntax and grammar.

It would seem valuable to create diagramming conventions that can be easily used and understood by both novices and experts. Unfortunately this is unrealistic for many

applications, considering the legacy of often cryptic notations that work well for a large population of experts. Nevertheless, by employing a perceptual foundation for graphical representations we can hope to come closer to a common basis for at least new problem domains where conventions have not yet been established.

1.1) Goals and Approach

The primary goal of this thesis is to explore the applicability of theories of structured object perception to creating more effective diagrams. Subgoals that follow from this are:

- to analyze the perception of structure and its relevance to diagram comprehension,
- to create a set of diagramming conventions based on the theory of structured object perception,
- to develop a computer graphics program to draw diagrams that incorporate the conventions,
- to evaluate the effectiveness of the resulting diagrams.

1.2) Organization of the Thesis

Chapter 2 begins with a taxonomy of the types of structured data. It reviews the field of visualization with respect to the display of structured information. It outlines the main display requirements that have guided researchers in the field of visualization. The mainstream techniques for visualizing lists, hierarchies, networks, and entity-relationship structures are presented according to the taxonomy.

Chapter 3 summarizes the perceptual and cognitive issues relevant to the problem of displaying structured information. It begins with an overview of the models that support the role of diagrams in cognitive activities. The main part of the chapter is devoted to how perception theories are used for representing diagrams. It starts by introducing low-level perceptual features, then continues with Gestalt principles, and concludes with the mainstream theories of object perception with an emphasis on theories of structural object recognition. The chapter ends with a presentation of studies into the perception of 3D displays.

Chapter 4 introduces a new set of diagramming conventions that collectively define the Geon diagram. The rules for drawing Geon diagrams are introduced. A major part of the chapter is reserved for describing the toolkit used for creating Geon diagrams. The toolkit's features are highlighted with an emphasis on the derivation of the parametric equations used for drawing a set of Geons. The chapter then concludes with a description of the necessary features implemented to make the toolkit a usable tool for evaluating Geon diagrams.

Chapter 5 describes a set of experiments that compare the Geon diagram to the mainstream UML diagram. The experiments specifically investigate the use of perceptual primitives for identifying and recalling structures within node-link diagrams. The results show that using perceptual primitives such as Geons has a positive impact on the tasks of visual parsing and recall.

Chapter 6 gives a description of two experiments used for investigating the importance of shading on the nodes and links in the Geon diagram. The results show that shading diagram elements plays a significant part in the identification and recall of the structures.

Chapter 7 highlights the use of perceptual semantics to construct meaningful representations. Theories of perception explain the way in which we construct images of objects using connectivity rules with Geons. A mapping of these rules for creating diagram structures can reveal semantic information intuitively. The investigation leads to a perceptual syntax with no restrictions on the type of audience using the visualization. The chapter ends with a complete description of what we call the Geon diagram.

Chapter 8 gives a summary of the contributions of the thesis. The most significant results and their implications are discussed. Potential applications are also reviewed.

Chapter 2 - Visualizations of Structured Information

We are surrounded with structured information and rely upon it on a daily basis. Structure helps us organize, search, categorize, and navigate information spaces [DDMR90]. With the abundance of information available, only a small fraction is being utilized and comprehended. Visualizing structured information facilitates tasks such as searching, pattern matching, and decision making. Visualization helps us grasp the underlying structures of the information, which constitutes the first step in the analysis of a problem. This section presents a survey of some of the more prominent and mainstream visualization techniques developed for the different types of structured information.

2.1 What is Structured Information?

Structured information is any information that can be analyzed into parts (or components) and relationships between the parts. As such, different analyses can create many structures from the same information. Thus structure is dependent on the bases or tools used for the analyses and extraction of the information. Regardless of the tools or methods used for structuring information, a structure can be separated into its form and its semantic content. The form is represented by the set of relationships between all the components. The content is the data that is encapsulated within the components and their relationships.

This thesis does not treat the subject of visualizing inherently spatial data such as maps and plans; instead its primary focus is on representation of non-spatial information. Information structures can exist in the following forms: lists, hierarchies, networks, and entity-relationship structures [Ware99]. Some information can be structured into a combination of any of these, for example, hierarchies within networks. In the following sections we describe the characteristics of these structures and later use them as a basis for categorizing and discussing visualization techniques.

2.1.1) Lists

List information is information that is structured based on a sequential ordering of discrete components. Text documents, file directories, program source code, alphabetical lists, chronologically ordered items, are examples of linearly organized data into lists.

2.1.2) Hierarchies

If we divide information into categories then we subdivide these categories into subcategories, and apply this division process onto these newly created subcategories, then we impose a hierarchical structure on the data. Hierarchies naturally describe structures of an organization, file systems, and family genealogies. Information structured as hierarchies has an implicit ordering organized into what is commonly referred to as levels. The hierarchy begins with a component at the top most level, and all other components are related to this entity. The hierarchy is further decomposed into sub-hierarchies with similar characteristics. There is usually very little correlation between components in sub-hierarchies at any given level.

2.1.3) Networks

Information structured into networks consists of information parts in which relationships are defined as connections between the parts. They are different than hierarchies in that they do not contain an implicit nesting of the data and every component can be related to another component in a variety of ways, i.e. every node can have more than one unique path to any other node in the network.

2.1.4) Entity-Relationships

A more general way of describing networked data is the entity-relationship model. The term 'entity-relationship' does not refer to the specific Entity-Relationship model as used in modeling database systems as introduced by Chen [Chen76]. In the thesis, it is used in a generic sense to classify all structures that can be described as having the components of the information referred to as entities, and the structure between components referred

to as relationships. In this type of structural organization, strong emphasis is placed on the type of relationships between the entities of the information. The relationships, as well as the entities, are qualified by attributes that give them a specific role in the classification of the information.

A classification of the structured information types discussed above can be depicted as shown in Figure 2.1. The entity-relationship model (ER) is the most general structure. Networks are a subset of ER structures in that they do not have as elaborate syntax of attributes on the entities and relationships. A hierarchy is a special case of a network and lists are subsets of hierarchies.

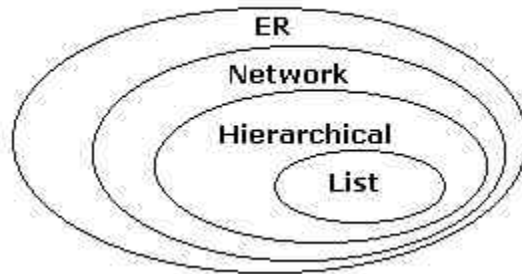


Figure 2.1 - Classification of the types of structured information.

2.2 Requirements of Visualization Techniques

The choice of a visualization technique depends on the analytic tasks and queries that will be applied to the information. Some common tasks applied onto hierarchical, network and entity-relationship data are:

- ❑ Discovering substructures from the larger structure,
- ❑ Discovering properties of sub-structures or the entire structure,
- ❑ Discovering relationships between different components of the data.

Unaided by visualization techniques, some of these tasks can be insurmountable with limited time and resources. Properly visualizing these structures has proven to considerably reduce the time allotted to these tasks [GE95] and in several cases they have revealed information unavailable with the use of more traditional aids [Spence00,GE97]. This section presents the requirements necessary for creating effective visualization systems.

2.2.1) Adequate Representation

An appropriate structure for the data should be chosen to adequately represent the information and it must not hide any details of the data that are relevant to the user. Visualizations are not always easily adaptable to the variety of types of information. For example, numerical and non-numerical displays may be considerably different and the same visualization techniques cannot be applied [Cleveland85]. To provide general solutions, visualizations should provide a combination of the basic types of techniques that mold to the particular problem [MacKinlay86].

2.2.2) Minimize Distortion

Displays should avoid presenting misleading visual artifacts. At the very least, the visualization technique should not distort the reality of the information being analyzed [Tuft90]. Incorrect mappings of visual properties onto information structures can lead to

confusing and distorted displays [RT96,Huff54]. For example, mapping a rainbow-hue color map to a qualitative scale can result in several deceptive artifacts [RL98,LH92,RLK92].

2.2.3) Semantic Representation

Semantic content is embedded within information structures. Structures should be arranged in ways such that their semantic information is unambiguous and immediate to the user of the visual display.

2.2.4) Abstractions

Visualizations need to define effective abstractions to give generality to the model. Without abstractions, the visualization is limited to a design specific solution and is not easily adaptable to other problems.

2.2.5) Interaction

Users need to interact with displays seamlessly and smoothly [BCS96]. Research into direct manipulation techniques is taking great strides in integrating all the sensory faculties to interact with a display to simulate real-life experiences [Hibbard99].

2.2.6) Navigation

Visualizations techniques need to incorporate navigation capabilities. When perceiving the world we use local detail and global context, referred to as focus+context. Even though we see detail for a small region, our peripheral vision provides us with a global context. This is important since we rely on the global context or peripheral vision to orient ourselves and to make sense of the local detail before us [Furnas86]. Several

focus+context techniques have been and continue being developed [Furnas86,SA83,SB94,CCF95]. For a more elaborate discussion on navigation in information displays see [HMM00].

2.2.7) Space

Limited by the size of display areas, visualization systems should be compact and dense but should not overcrowd information, which can result in poorly communicating the structure. Several solutions include presenting the information in multiple views [BWK00], using efficient layout techniques [CBTTB92], and presenting only what is necessary.

2.2.8) Scaling

Visualizations should handle both large and small data sets. Certain visualizations work well for a limited size structure, and expanding the amount of information makes the visualization impractical. This is a common problem faced in visualizing hierarchies, which by nature grow exponentially.

2.2.9) Visual Quality

Visual displays should have sufficient quality for the data to be seen. Advances in hardware and software technologies have enhanced display and geometry resolutions. However, visual displays do not provide users with the same quality of resolution we are used to when interacting in the world [Tuft83,Hibbard99].

2.2.10) Integration

To be useful, visualization techniques need to be integrated by coupling them with different systems such as networks and software systems to facilitate visual tasks.

2.3 Display Techniques

Here we review some of the techniques that have been developed for visualizing the types of structured data described above. We begin with lists, then consider trees, networks and entity-relationship models in turn. The analysis is based in terms of the requirements presented in section 2.2 above.

2.3.1) Displaying List Information

Data organized as lists impose a sequential ordering on the elements. Often the data is textual in nature catalogued as documents (components) that are loosely related. Current display techniques for list structures visualize either the content of a single document or visualize the structure between a collection of interrelated documents. Various visualization techniques have been developed for presenting list information.

2.3.1.1) Time lines

Data structures can encode temporal information. The arrangement is a list ordered with time (Figure 2.2). Such a mapping of data onto space allows comparisons between two points in time. A time slice is the amount of time that has elapsed between two adjacent points.

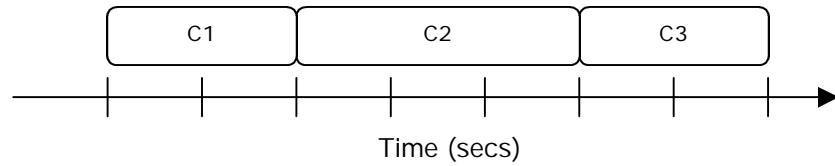


Figure 2.2 - Video clips C1,C2 and C3 laid out on a time line.

2.3.1.2) SeeSoft

SeeSoft is a program developed to visualize textual information of large and complex software systems [ESS92]. The information about each file in the system is arranged into vertical columns. The elements of the structure are ordered lists based on the ordering of the lines of codes in the software module (Figure 2.3). SeeSoft presents information on software complexity metrics, types of software modifications, number and types of bugs. This information is then shared among software architects, programmers and managers, who can develop an understanding from the patterns revealed. The main feature of this tool is that it compresses lines of linear code into rows of pixels. It also has the capability of switching between multiple views, the more common ones being the Line View and the Pixel View. In the Line View representation, SeeSoft displays each line of source code as a single colored line. Rectangles representing each file are presented with the grouping of all the lines. Thus, the user can immediately scan for larger files by looking for the longer rectangles. Line length can show the length of the line and indentation in the source code, and therefore can display a rough outline of the control structures in the software (for, while, and if statements). Color is mapped to a variety of attributes that could represent the date of origination, date of change, id of the person who changed the

line, which lines are bug fixes, nesting complexity, and the number of times the line was executed during testing.

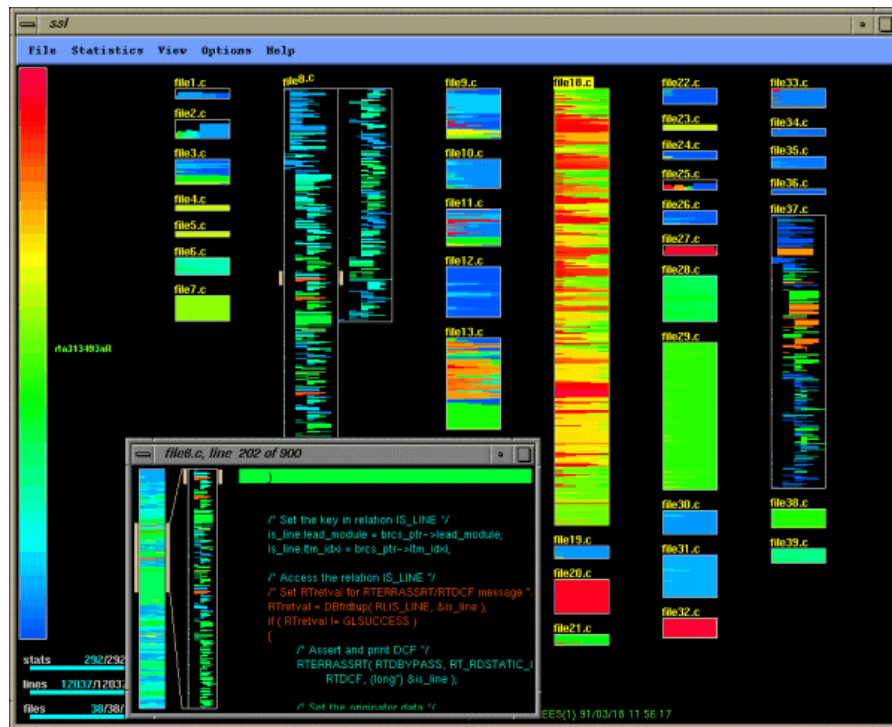


Figure 2.3 - SeeSoft view of fifty-two files containing 15, 255 lines of code. The pixel and line representations are being used for the files. The pixel representation (inner window) provides a more detailed line representation. Color-coding is used for representing the age of the code where the newer lines are shown in red and older lines in blue. (Reproduced with permission)

2.3.1.3) Lifestreams

Lifestreams organizes "streams" of documents into chronological order [FF95]. This mapping of documents onto time inherently creates a linear structure (Figure 2.4). The z-axis is used for representing a series of events. The x- and y-axes are used for providing space for seeing details of a specific event. If a document is of interest, by clicking the

document, it expands and becomes the center of focus on the screen. It is claimed to be an effective technique for keeping track of multiple versions of a document.

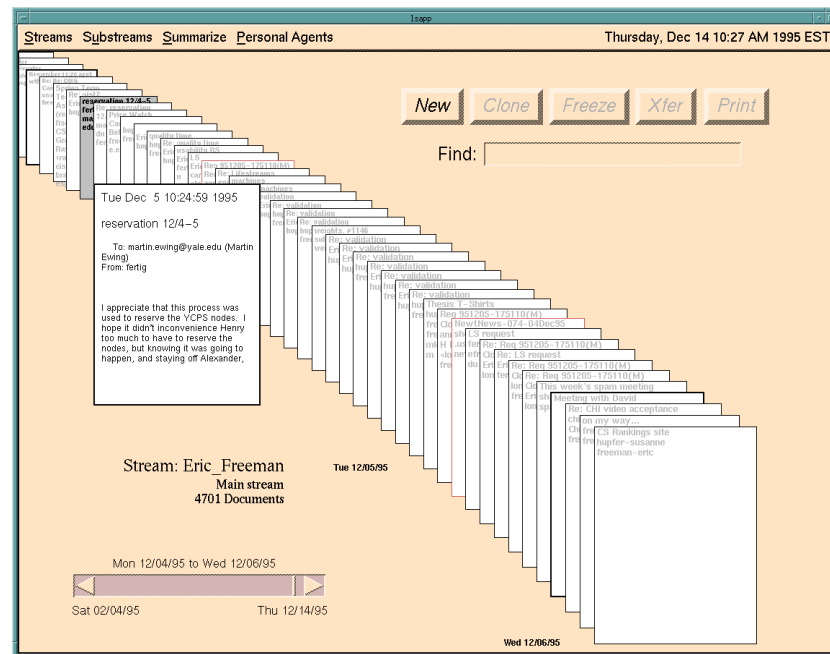


Figure 2.4 - Documents visualized using Lifestreams. (Reproduced with permission)

2.3.1.4) TimeTube

In certain instances, the development of a document needs to be represented. The Time Tube visualization represents the evolution of content on the Web over long periods of time [CPMPGC98]. The mapping onto time once again imposes a list structure shown as a left to right ordering. At each time slice, a Disk Tree is used to show the structure of the Web and the links that have been more active [CPMPGC98] (Figure 2.5). Overall, the Time Tube created with a series of Disk Trees acts as an aid to authors and webmasters

with the production and organization of content, it assists web surfers in making sense of information, and helps researchers understand the Web.

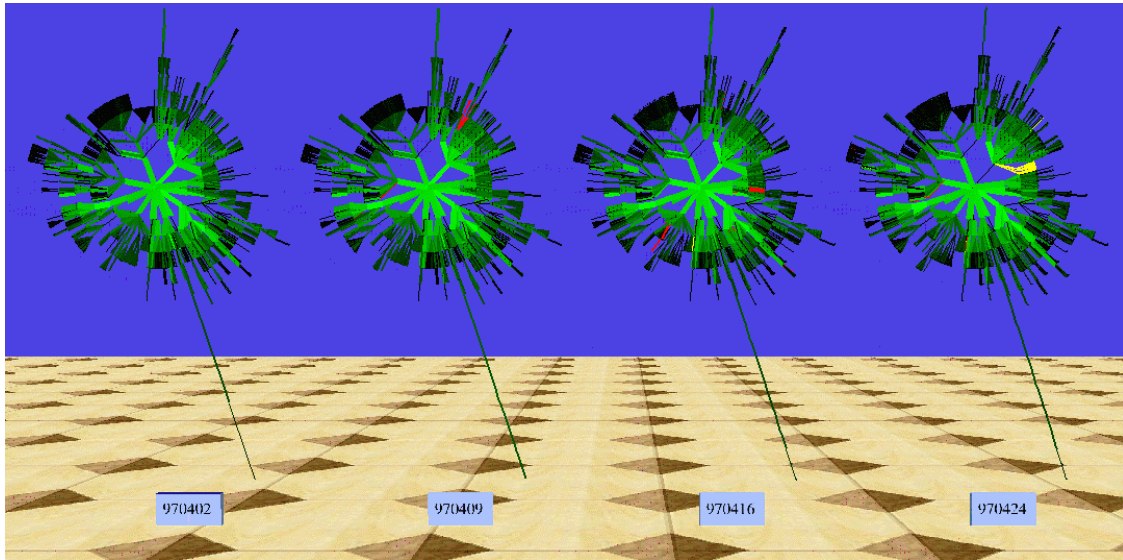


Figure 2.5 - A Time Tube formed by the evolution of a web site in four weekly increments [CPMPGC98]. (Reproduced with permission)

2.3.2) Displaying Hierarchies

Hierarchies are structures in which each component has one parent and there exists a top-most parent (or root). Each component can have one or many children (Figure 2.6). They are also referred to as trees, which have the property of placing components at levels denoting some importance or role in the hierarchy. Hierarchies cannot have cycles, i.e. there is only one path from one node to another. Several challenges exist in displaying hierarchical structures:

- The number of nodes can be an exponential function with the number of levels therefore it can be difficult to display large deep trees,

- ❑ For large structures some method for navigating the hierarchies to maintain focus and context is needed,
- ❑ Depending on the application, it may be necessary to show the size of subtrees, their paths and the entire structure. Hierarchies can be unbalanced, with some branches that have deeply nested structures and others that are 'bushy' requiring a wide arrangement of nodes on each level.

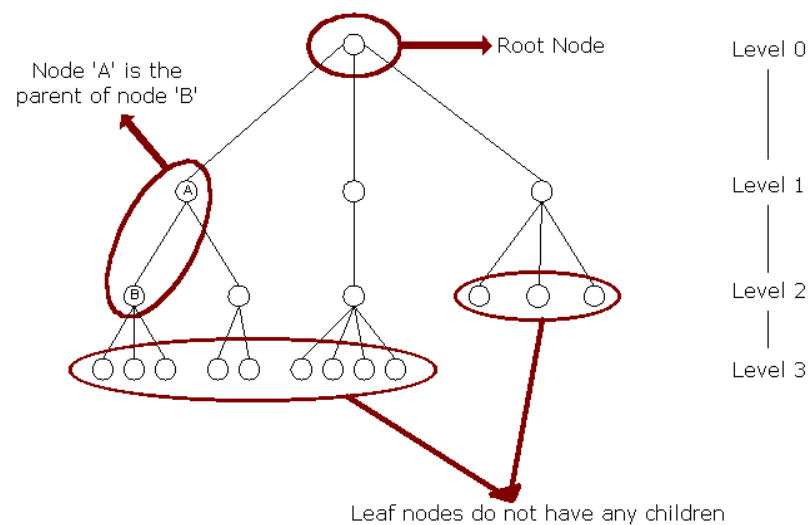


Figure 2.6 - A generic view of a tree or hierarchy.

2.3.2.1) Tree-Views

The most common method for illustrating a tree is shown in Figure 2.6. However Windows and other Operating Systems have made an alternative interactive view popular referred to as Tree-views. They are used to display the hierarchy of files on a disk. The tree can display several levels of the hierarchy in a single view and at each level the user can expand or contract the hierarchy (Figure 2.7). Its limitations include the incapability

of seeing the entire structure of the hierarchy, the possible loss of context at a given level in the hierarchy, and limited navigation through the view. Its strengths include excellent display of text labels and easy interactivity in showing and hiding sub-trees.

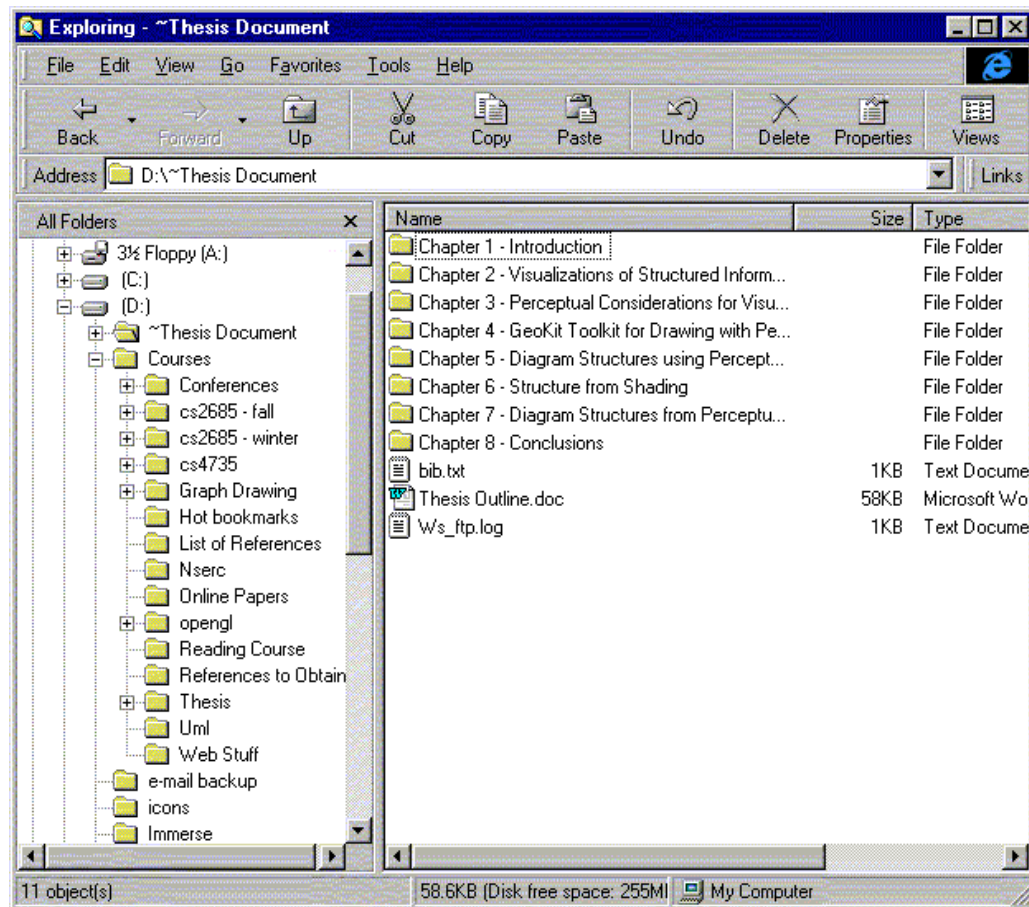


Figure 2.7 - View of my hard drive using Windows Explorer.

2.3.2.2) Cone Trees

A cone tree is a 3D representation of hierarchical information [RMC91a]. The root of the tree becomes the apex of a cone, with its children evenly spaced around the circumference of the base (Figure 2.8). This process is iterated for the entire hierarchy

such that each layer has the same height. Furthermore, the aspect ratio of the tree is fixed to ensure the tree fits within the viewing area. Shading can be applied to the body of the tree so the cone is easily perceived [RMC91b]. Cast-shadows can also be shown below and to the side which give alternative views of the conic structure. To navigate, the user clicks on the node of interest and rotates the tree around to bring the node and its path to the front. Robertson et al. claim that as many as a thousand nodes can be effectively displayed using cone trees [RMC91a].

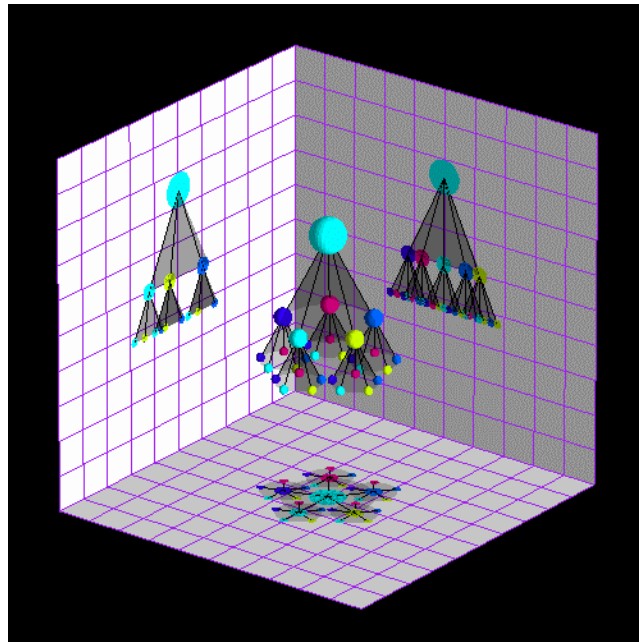


Figure 2.8 - Structure of a cone tree.

2.3.2.3) Hyperbolic Browser

The hyperbolic browser lays the information tree out onto a hyperbolic plane and then maps it onto a circular display region [LR94] (Figure 2.9). In a hyperbolic plane parallel lines will diverge from each other and the circumference of a circle grows exponentially

with its radius. Thus as the distance from the center increases, space becomes available exponentially. A hierarchy can be laid out in a hyperbolic space in a uniform way, such that the distance between the parents, children and siblings is approximately the same everywhere in the hierarchy. To navigate the hierarchy, any visible point is dragged into another position to translate the view over the hyperbolic surface. Munzer [Munzer97] has implemented a variation of the hyperbolic browser in 3D space.

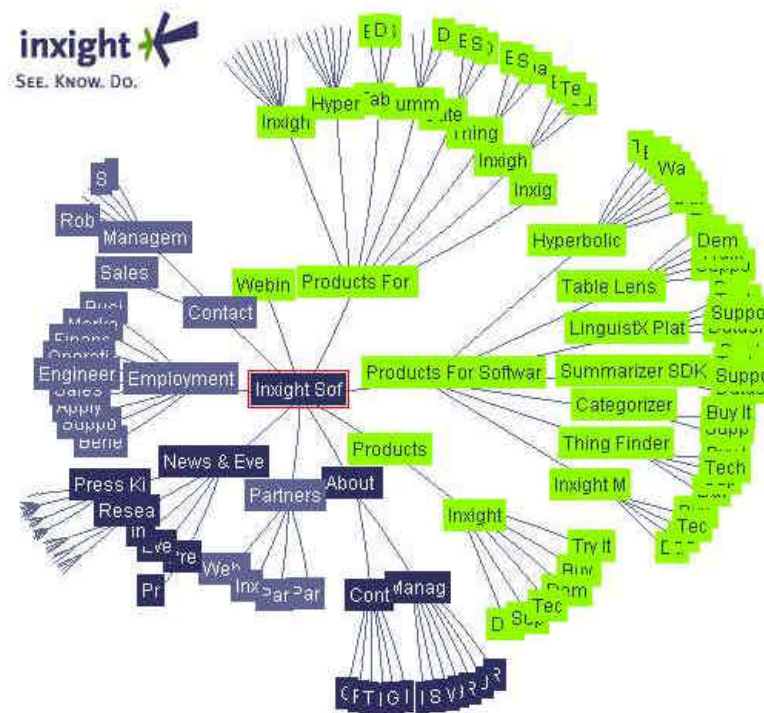


Figure 2.9 - Hyperbolic Browser displaying the hierarchy of the www.inxight.com.

2.3.2.4) Cheops

In the Cheops approach [BPV96], a hierarchy is compressed by overlaying children nodes onto parents (Figure 2.10). The compression reduces the total number of nodes to a

set of Cheops visual components, which are essentially shared nodes at any given level. The node is shown as belonging to a subtree when the user clicks on the root of that subtree. In the example illustrated below, the figure to the right shows the Cheops version of the tree to the left.

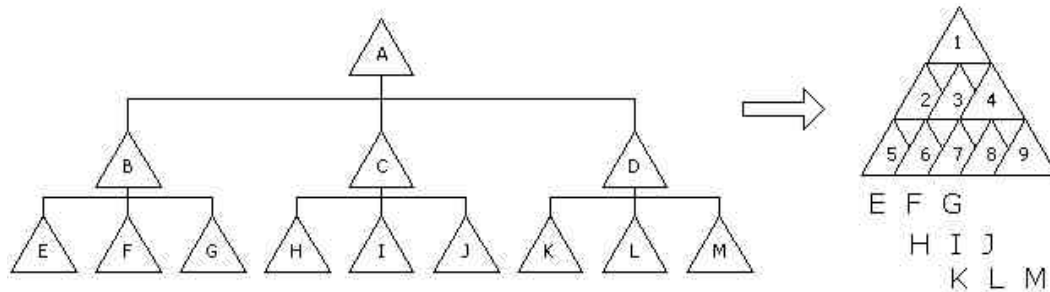


Figure 2.10 - A 3x3 deep hierarchy and the resulting Cheops representation. The Cheops version reduces the 13 logical node hierarchy into a 9 visual node pyramid and requires 1/3 the drawing space.

Triangles 6, 7 are shared between nodes F, G and nodes H, I. If triangle 2 is selected then triangles 6 and 7 become nodes F and G. If triangle 3 is selected, then triangles 6 and 7 will become nodes H and I (Figure 2.11). Therefore, the representation of each logical branch of a hierarchy is a result of a specific action at a previous levels.

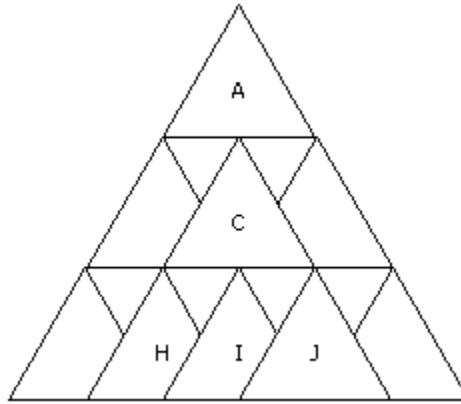
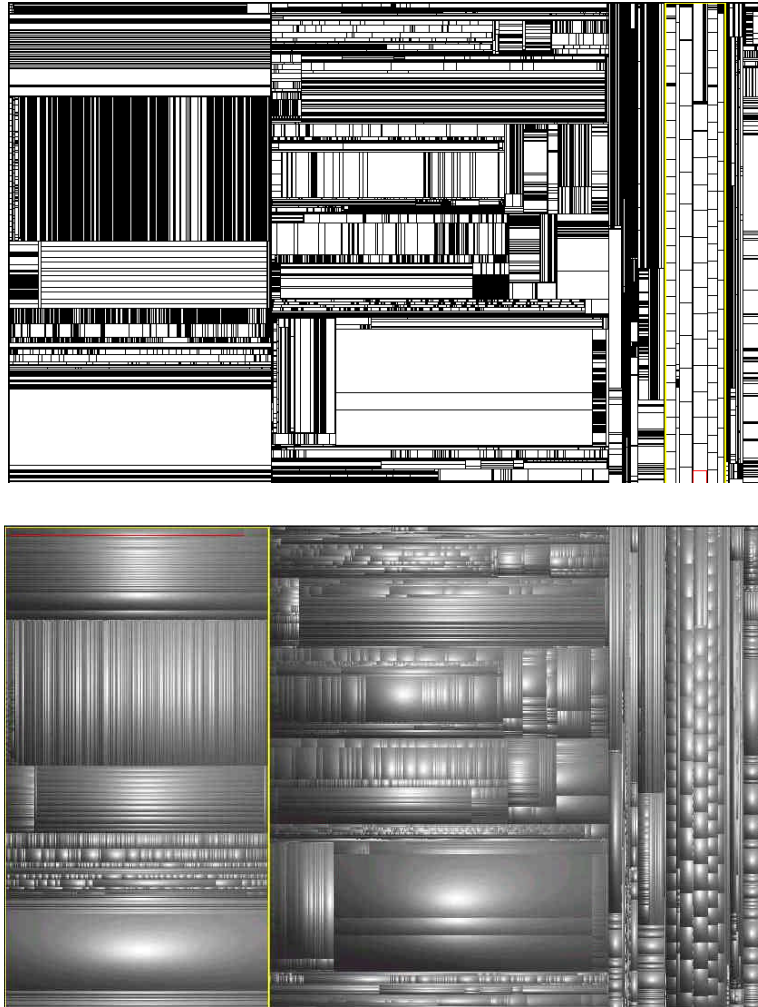


Figure 2.11 - Selection of node C gives details and brings to the front nodes H, I, and J.

2.3.2.5) Tree Maps

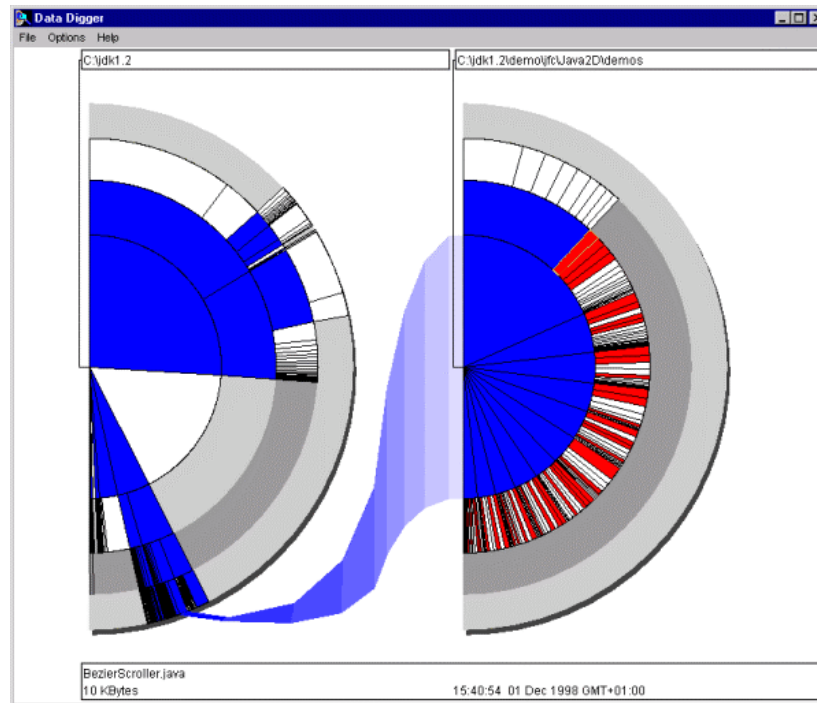
Tree-maps [JS91] is a space-filling technique that divides the display area into regions to map an entire hierarchy of nodes and their children based on their properties. The space is filled based on the weight of the node and its children. As such, the relative size in the display area for each node is determined (Figure 2.12). TreeMaps reveal patterns in the data such as a large area of a specific type of information. However, they tend to obscure the hierarchical structure. They have been applied to visualizing stock market information (marketmaps) [SmartMoney] and displaying competition trees in a tennis match (TennisViewer) [JB96]. A 2-1/2D shaded version of the treemap (cushion map) has been implemented to reveal the structures better [WW99].



**Figure 2.12 - Tree Map and Cushion Map displays of disk drive.
Large files can be immediately perceived.**

2.3.2.6) Information-Slices

Information-Slices uses a semi-radial display to fan out hierarchies along semi-circular discs [AH98]. The area displayed on each disc is proportional to the size of the sub-hierarchy. A sub-tree can be displayed on additional slices by clicking on the hierarchy (Figure 2.13). Information-Slices can to some extent show both, the size of nodes and the tree structure.



**Figure 2.13 - Information-Slice view of files on a hard drive.
(Reproduced with permission)**

2.3.3) Displaying Networks

Certain types of information consist of multiple interrelated parts with arbitrary links between them (Figure 2.14). Examples include hypermedia pages and links such as those found on a web site, physical and logical organization of computers and networks in a corporation, or semantic databases of elements and relationships between them. Networks are traditionally displayed with components represented by nodes, and relationships represented by links between the nodes thus sometimes referred to as node-link displays. The most abstract form is the graph, which emphasizes the pure topology of the structure.

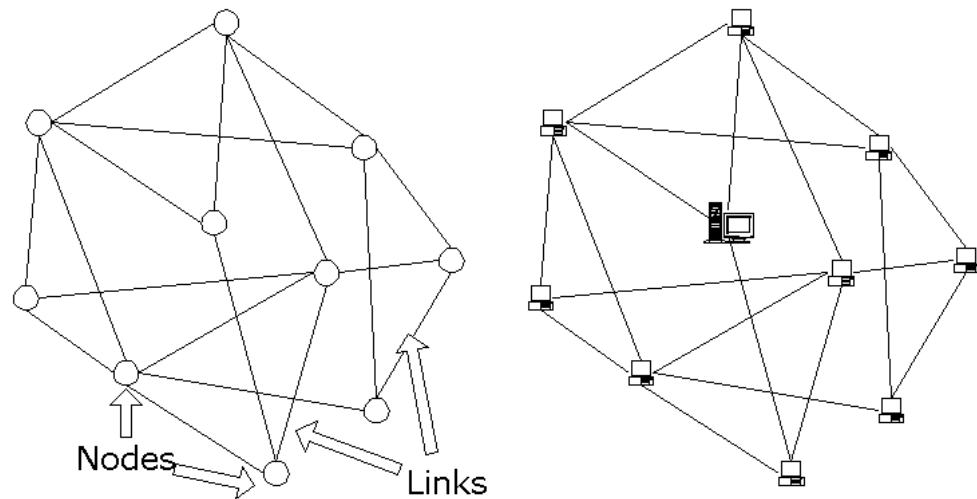


Figure 2.14 - The most abstract form of a network is a graph depicted in the left image with nodes and links. The image to the right displays the logical structure of a computer network with the server in the center.

In many instances techniques from the field of graph drawing [BETT94] have been utilized to visualize networks. Card et al [CMS99] outline some of the challenges in visualizing large networks:

- ❑ Positioning nodes to make the display 'legible'
- ❑ Node and edge layout so that crossings are minimized
- ❑ Scaling from small to large set of nodes
- ❑ Interacting and navigating to reveal different sub-structures

Some of the more elaborate techniques that have functional solutions for these challenges are NicheWorks, SeeNet, Harmony Map, SemNet, NV3D, and Narcissus. The first three systems lay out the information in 2D space, while the last three create a 3D environment for visualizing the graph structures.

2.3.3.1) NicheWorks

NicheWorks is a visualization tool created for exploring statistical properties of the node and link data in a graph structure [Wills97]. It has the capability of displaying networks with up 1,000,000 nodes and gives the user the freedom to explore parts of the graph, query for information on demand, or visit multiple views of the same data. To present all the required information, NicheWorks has a built-in layout manager, which uses a variation on radial and tree layouts (Figure 2.15) [BETT99]. It has applications in visualizing relationships between functions and files in a large software development effort, in providing web site analysis and in performing correlation analyses in large databases.

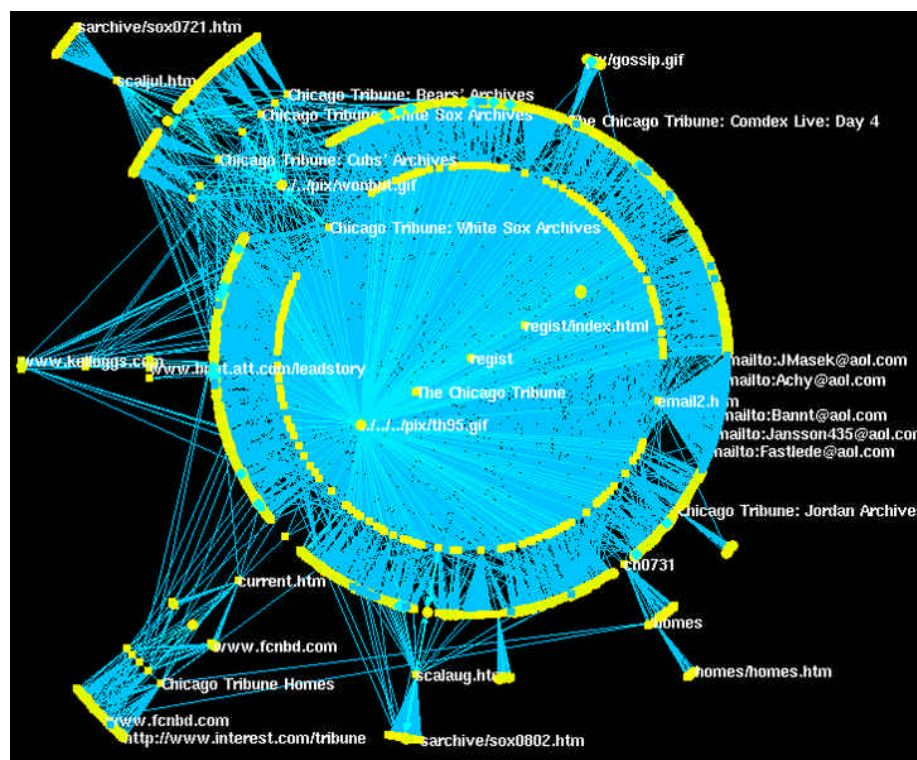


Figure 2.15 - Visualization of the Chicago Tribune Web Page. The nested structure of the web site is immediately apparent. (Reproduced with permission)

2.3.3.2) SeeNet

The focus of SeeNet is to visualize data on a physical network and not the structure of the network [BEW95]. It was designed to answer such questions as where overloads occur, on which dates, during which periods, and where network damages occur. The entities in SeeNet are mapped to the physical location of cities and thus use spatial variables for positioning the nodes (Figure 2.16). As a toolkit, it allows the user to parameterize multiple views for better comprehension of the data.

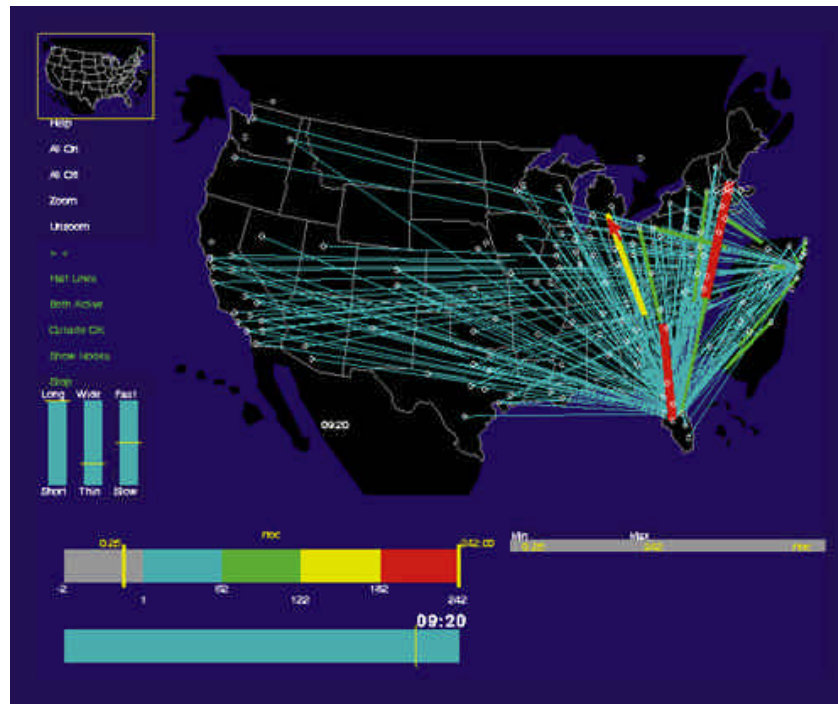


Figure 2.16 - Call connections on the AT&T network on December 24th. A concentration of incoming and outgoing calls is noticeable around Florida, probably due to a higher concentration of retirees or winter vacationers in this region. (Reproduced with permission)

2.3.3.3) Harmony Local Map

The Harmony Local Map was developed to display a hypertext node-link network [Andrews96]. It visualizes documents on a web site as nodes and the links in the documents as connections between the nodes. The nodes are assigned levels based on their topological ordering (Figure 2.17). The visualization then uses a variation of Eades and Sugiyama's [ES90] graph layout algorithm to reduce the number of edge crossings.

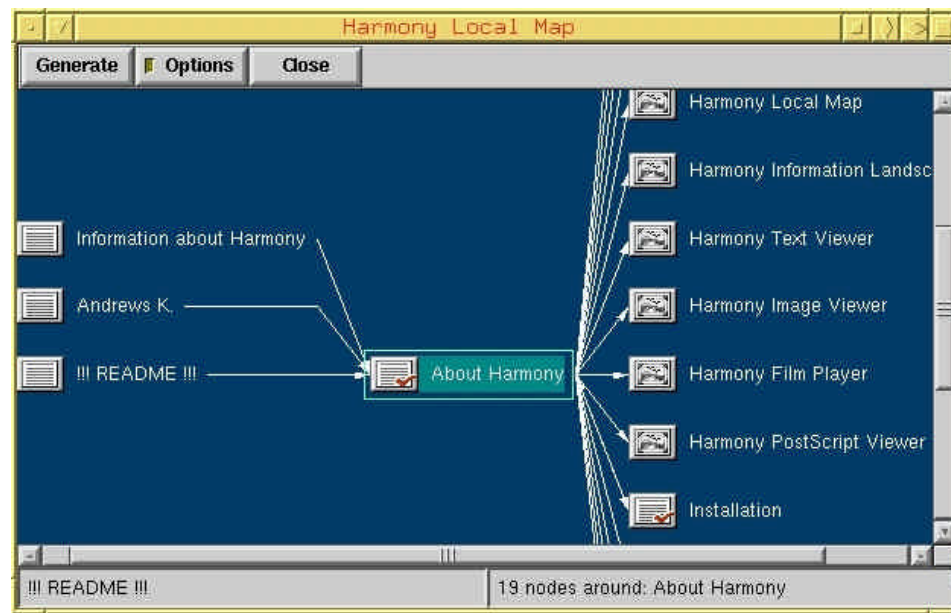


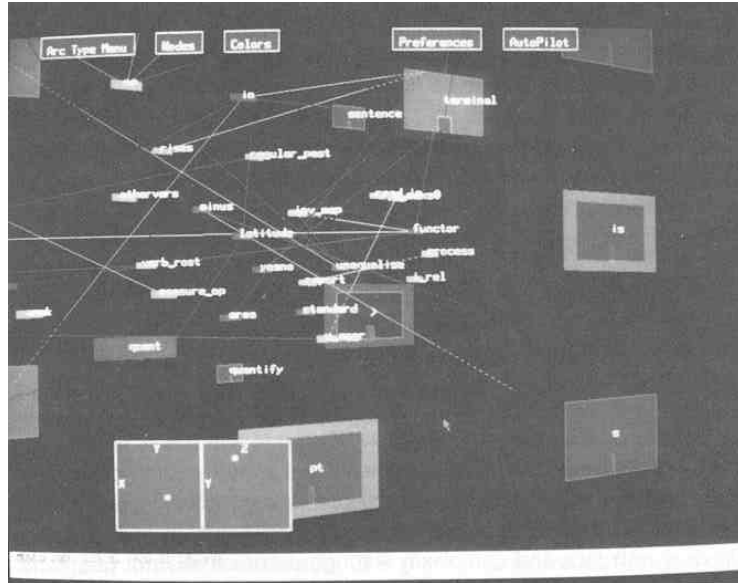
Figure 2.17 - Harmony Local Map display of the Harmony web site. (Reproduced with permission)

2.3.3.4) SemNet

SemNet (Figure 2.18) was designed as a research prototype to advance understanding of the complex relationships and structures of large, arbitrary knowledge bases [FPF88]. To avoid the large number of arc intersections that can exist in networks, SemNet used a 3D

layout of the elements in the knowledge base. It positioned the knowledge elements or nodes using three sources. Mapping functions take some inherent attributes of the elements and map them onto the space. Proximity-based functions were used to automatically position related elements in proximity to one another. As a result, SemNet reduced screen clutter by grouping spatially proximate elements into single cluster objects. Finally, users were given control for positioning elements for which they had additional information.

One of the goals of SemNet was to explore the application of 3D visualization techniques and to investigate the problems that could thereby result. A main problem addressed by SemNet was to present large knowledge bases for improved and more efficient comprehension. It did this by displaying information of the details while maintaining a global representation of the rest of the knowledge base. Fairchild and Poltrock hypothesized that for comprehension of a knowledge base, a user must recognize: 1) the identity and meaning of individual elements; 2) the relative position of elements within a hierarchical context; and 3) the explicit relationships between elements [FPF88].



2.3.3.5) NV3D

The aim of NV3D was to utilize 3D visualizations to provide a more readable and comprehensible graph visualization [WF94]. The project was aimed at displaying predominantly directed graphs, though the focus of the development had been to visualize software structures. The structures in this case were component relationships within object oriented C++ programs. Some examples included software module dependencies, software usage, and inheritance relations. NV3D drew on a number of ideas from SemNet and hinged upon the same concepts. However NV3D improved upon SemNet in a number of ways. One area of divergence was a shift from the predominantly automatic node placement of SemNet to a hybrid of a manual and automatic layout of nodes in NV3D. A major emphasis in NV3D was the implementation of interactive queries that allowed subsets of the data to be revealed or hidden.

Similar to SemNet, NV3D also included the notion of composite (or nested) nodes and arcs [Harel88]. This reduced the visual complexity of a graph structure by representing similar nodes and arcs as a single composite node or arc. These composite nodes could then be collapsed or expanded (opened or closed) as needed, depending on the focus of the viewer (Figure 2.19). By nesting nodes a natural hierarchy can be derived from the structure.

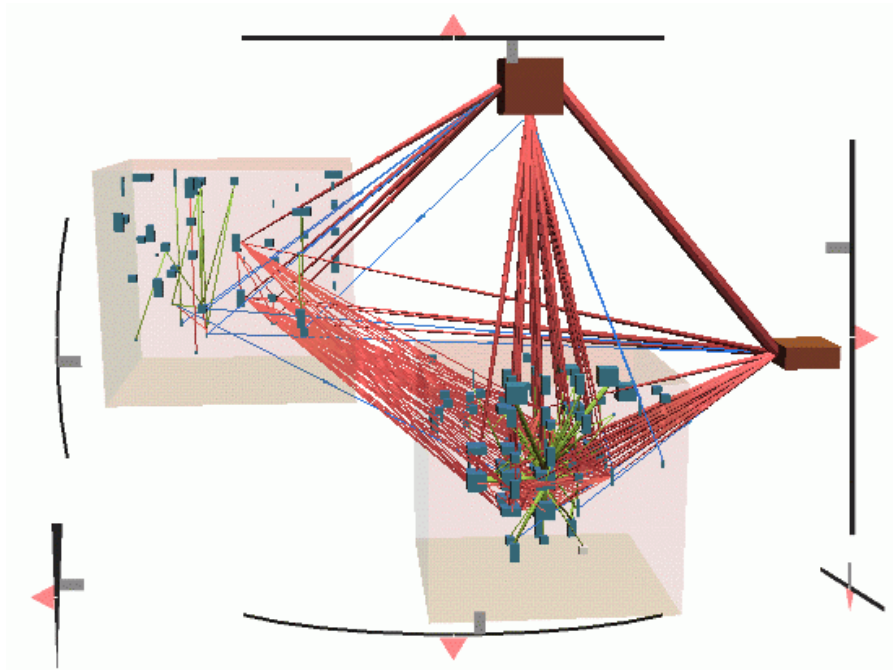
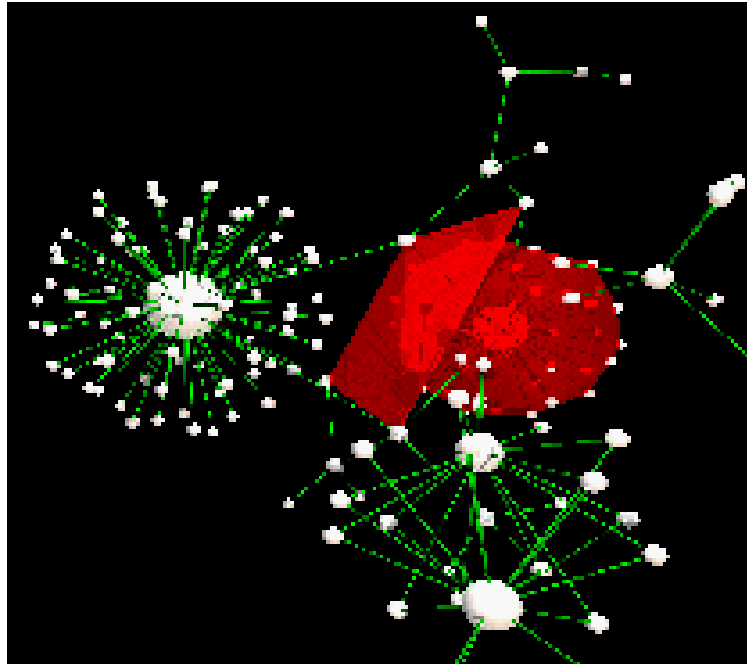


Figure 2.19 - This graph represents almost 6 million lines of code. The graph contains approximately 33 thousand nodes and 34 thousand relations. Nesting leads to hierarchies within the network. (Reproduced with permission)

2.3.3.6) Narcissus

Narcissus is a self-organizing 3D-layout visualization model through which users can navigate and manipulate objects [HDWB95]. Each component of the structured

information is given a behavior that determines its movement and position in 3D space. The rules that define the position of the nodes are based on forces. All nodes exert a repulsive force on all other objects and active relationships between nodes lead to attractive forces. As a result, after reaching a steady state, the visualization creates clusters of nodes that are closely related (Figure 2.20). In applying this visualization to the display of software systems, the user can easily find patterns that help identify relationships of inheritance and dependencies.



**Figure 2.20 - Web site visualization with clustering in Narcissus.
(Reproduced with permission)**

2.3.4) Displaying Entity-Relationship Data

In our classification scheme (Figure 2.1) the entity-relationship model is the superclass of all the other types of structured data. In this type of structured information we are not only interested with the organization and layout of the structure but also pay attention to

the types of components and the types of relationships in the structure. In addition to having nodes and links, this kind of data is comprised of attributes that describe and qualify each of the components and their relationships. Some of these attributes can be multidimensional. As stated earlier, it should be clear that the term 'entity-relationship data' is used here as a generic term to classify all models that can be categorized using the description above and does not refer to the specific ER model as defined by Chen [Chen76].

Visualizations of entity-relationship structures primarily consist of static diagrams with nodes and links equipped with attributes. These diagrams usually contain fewer nodes than a network diagram, as the presentation of the attributes is an important element of such visualizations. Diagrams for modeling entity-relationship data have been integrated into packages such as CASE (Computer-Aided Software Engineering) tools. It is also interesting to note that most entity-relationship models have been developed for the creation of software systems (database systems, object oriented systems, etc.). Only recently they are being introduced to areas such as enterprise modeling [Marshall99].

What follows is a description of some of the most common entity-relationship systems. They can all be thought of as subclasses of the general entity-relationship model as shown in Figure 2.21.

General Entity-Relationship Model

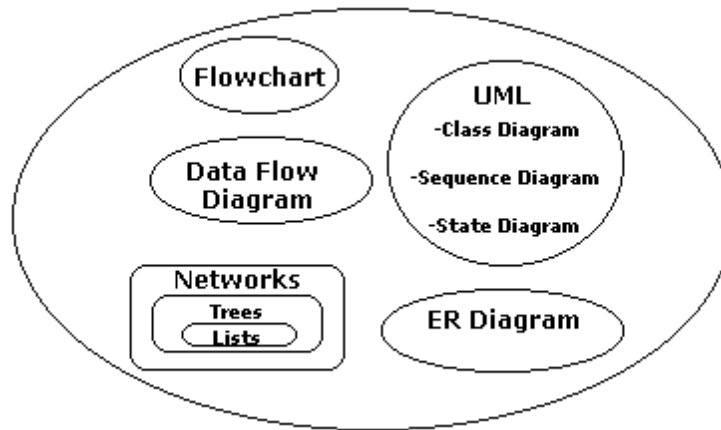


Figure 2.21 - Classification of the general Entity-Relationship Visualizations. Networks, trees and lists can be thought of as sub-categories of the ER model.

2.3.4.1) Flowcharts

Flowcharts played an important role in the developmental stages of the computing era and consist of some of the earliest visual aids in software modeling [Goldstine72]. They are static diagrams, depicting the control flow of a program by using interconnected nodes with different shapes to reveal programming structures such as decision branches, loops, or simple statements (Figure 2.22). The programmer can trace the accuracy of the flow of logic before proceeding to implementation. The entities in this diagramming convention represent different kinds of decision points. The relationships in this system are simply flow of control from one area to another in the program. Several studies have investigated the practical utility of flowcharts with respect to programming tasks and have reported mixed results [BD80a,BD80b,SMMH77,Scanlan89]. In general, research indicates that flowcharts have no significant effect on program composition but facilitate

understanding of logic structures. With a shift from procedural programming to object oriented systems, these diagrams have become less common and have been superseded with the class of sequence diagrams developed in the Unified Modeling Language (section 3.4.4).

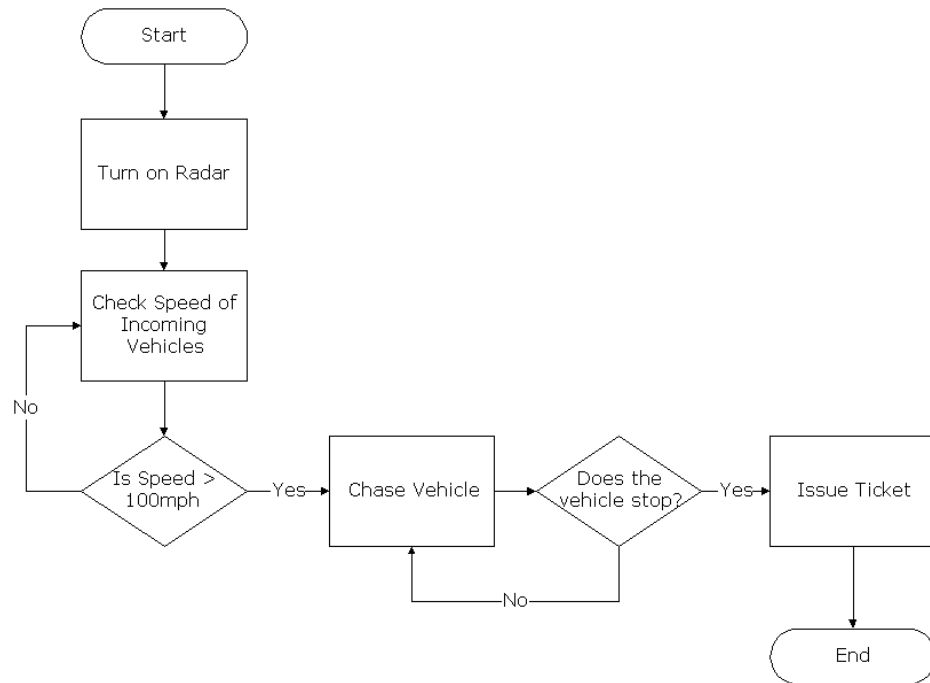


Figure 2.22 - Flowchart depicting the processes and decisions of a computer system monitoring the speed of motorists.

2.3.4.2) Data Flow Diagram

Data Flow diagrams are traditionally used during the software analysis stage of a project for stimulating understanding and communicating the requirements of the system to be created [GS86]. In data flow diagrams, constituent entities represent processes, data stores, and users interacting with a system. Entities are represented as nodes having

differing shapes depending on their role in the diagram. The links or relationships between the entities, are represented as labeled arrows indicating the flow of data from one component to another (Figure 2.23).

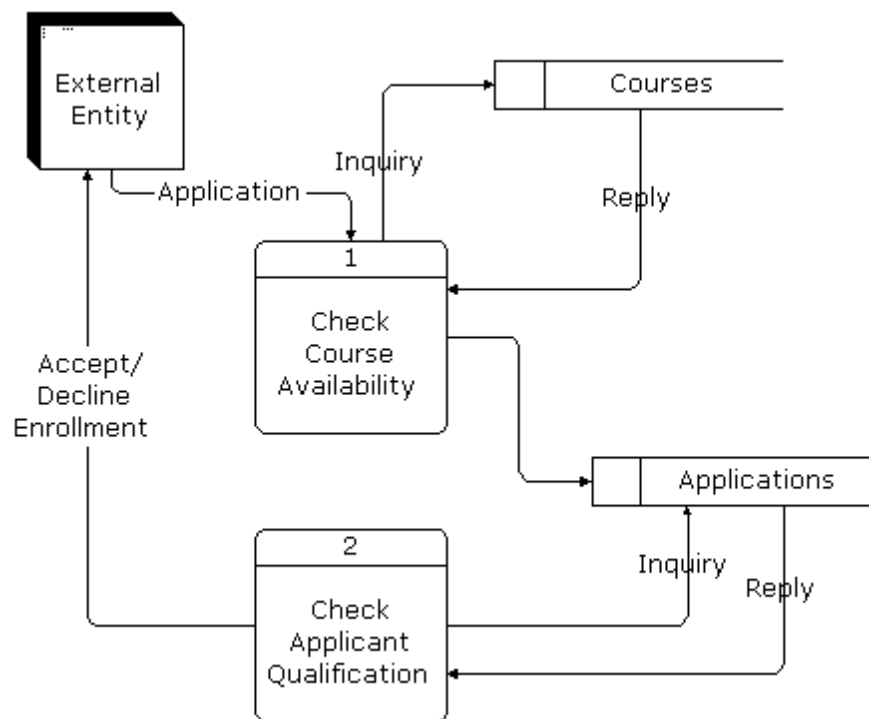


Figure 2.23 - Sample Data Flow diagram depicting the flow of data between processes, external entities and data stores in an automated course registration system.

2.3.4.3) Entity-Relationship Diagram

The Entity-Relationship diagram (not to be confused with the generic class of entity-relationship information as described in the sections above) was one of the earlier models enabling system architects to capture semantic information about the real world [Chen76]. It was developed to address some of the weaknesses of the network model [Bachman69], of the relational model [Codd70], and the entity set model [SAAF73].

Since its inception it has been widely applied to the development of database systems, by capturing relevant information concerning the entities and relationships of the system being modeled.

The visualization of this type of entity-relationship has several variations, but the most common consists of nodes depicting the entities. The shapes of the node can be rectangular or diamond like and are equipped with labels for the name of the entity (Figure 2.24). The relationships can be typed as weak or strong relationships, or have properties denoting multiplicity. In some instances, a description is provided on the relationships and in this case the link is made up of an appendage node.

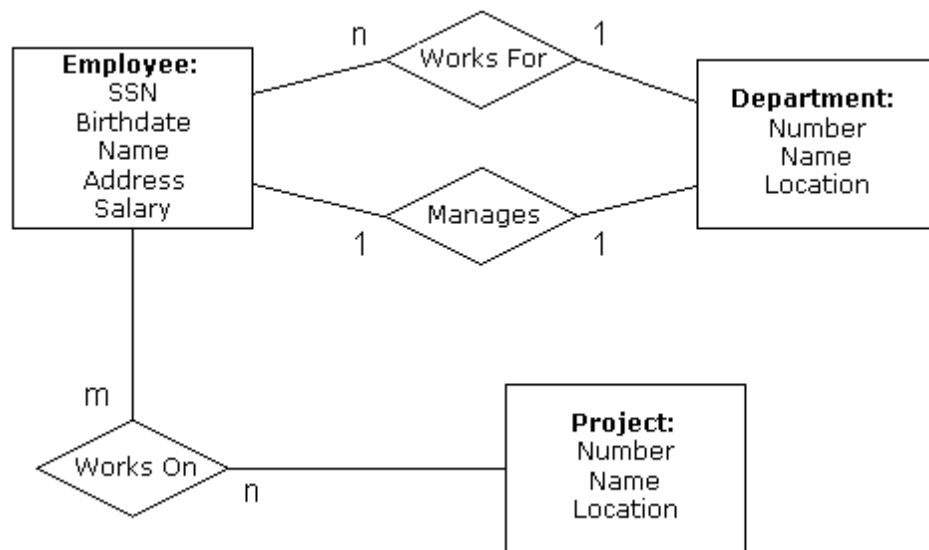


Figure 2.24 - Sample ER-Diagram depicting entities of an organization.

2.3.4.4) UML

A more elaborate and richer category of entity-relationship model is the UML (Unified Modeling Language) system. The notation has been derived from three main sources Booch [Booch94], Rumbaugh [Rumbaugh91] and OMT, through the efforts of the Object Management Group (OMG), to unify the standards and tools for diagramming software systems. It is comprised of a set of modeling tools, with different diagramming notations, each of which is used for achieving a given task during the software development cycle. Sequence diagrams, for example, depict the sequence of flow of data between different components of the system. The class diagram in UML presents a view of the entities in a system, including their interconnected relationships, attributes and operations [Fowler96]. It helps architects draft software objects and design relationships such as inheritance between objects, dependencies of entities, and aggregation or composition of objects within others. A sample UML class diagram is illustrated in Figure 2.25. UML is continuously evolving primarily in formulating new semantics for software constructs. In general UML diagrams show entities as rectangular boxes containing various text fields and the relationships as linking lines with different visual characteristics. For example, large arrowheads are used for representing inheritance, aggregation is denoted using a diamond shaped element on one end of the link, and asterisks (*) are used for denoting multiplicity (Figure 2.25).

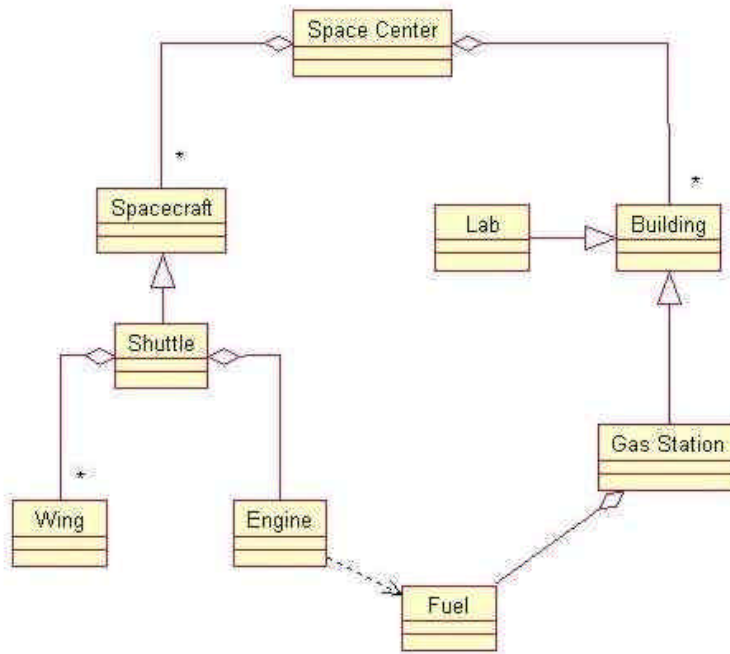


Figure 2.25 - Sample UML diagram depicting entities and relations of a space station.

2.4 Discussion

The taxonomy presented in this chapter classifies structured information under four main groups: lists, hierarchies, networks and entity-relationships. Lists are structures based on an ordering of discrete components. Hierarchies are formed from levels of structures within the data. Networks consist of parts and connections between them. Entity-relationships consist of components referred to as entities, and connections between the entities referred to as relationships. Entity-relationship information places a strong emphasis on the type of entities, relationships, and their attributes.

The investigation in this thesis is primarily focused at structured information of the most general type, entity-relationship structures. Visualizations for this group have the general characteristic of node-link diagrams where nodes represent the entities, and links, the relationships. Attributes of entities and relationships are expressed with coded symbols, a variety of line styles, and written labels. These diagrams are expressive and are embedded with semantics that enable them to capture models of real world systems. As such, they facilitate the communication process during the development of automated systems. However, their syntax and notations have been developed arbitrarily and primarily for experts in a given domain. This can impede and adversely affect the use of these diagramming convention by non-experts. In this thesis, the investigating is directed to the visualization of both, the structure of and the semantics of node-link diagrams.

In the next chapter, perceptual principles that can be applied to the syntax of visualization systems are introduced. The syntax of node-link diagrams is related to Gestalt theories and more recent theories of object recognition.

Chapter 3 - Perceptual and Cognitive Issues

The primary role of an information visualization designer is to analyze the requirements, translate them into information structures and later map these to visual structures. Usually data requirements will involve many variables and parameters. As a result, mapping this information onto visual structures can result in a number of alternative visualizations. The best display is the one that is capable of showing as many distinctions in the data, provides sufficient patterns to detect anomalies, and is faster to interpret [MacKinlay86].

From a perceptual standpoint, we can analyze a visual display in terms of three major levels of visual processing: graphical marks or features, patterns, and structured objects. At the lowest level are graphical marks, which consist of the one-to-one mapping of data elements onto graphical attributes. At an intermediate level are patterns, which consist of

groupings that are constructed from graphical features in the visual display. At a higher level, related structural objects assist in perceiving the organization of the data in the display.

After perceptually processing the visual input for extracting the needed information, cognitive mechanisms are involved in solving a problem or communicating the information. Cognitive theories explain the reasons for which diagrammatic displays assist in the problem solving process. In the section that follows, an introduction to theories of cognition is provided, explaining the role of diagrammatic representations in communicating and problem solving. Following this, we discuss more perceptual issues, beginning with low-level graphical marks, pattern perception as it relates to graph layout and finally object recognition and how it relates to identifying structures. Throughout the chapter, emphasis is placed on representing structured information.

3.1 Cognitive Theories

There has been a tremendous growth of research into the cognitive processes that take place when visual aids are used for problem solving. Diagrams in particular make up for a large portion of external aids. They nurture and enhance the development of theories and concepts; and without these one could argue that further progress of conceptualizing ideas would be limited. Diagrammatic representations have played a central role in the evolution of our science and technology. Some examples include Galileo's kinematics diagrams, Newton's dynamics diagrams, Babbage's mechanical notation. These

representations encode scientific principles and can be used as tools in building conceptual models [Cheng96].

Aside from enabling the construction and formalization of theories, diagrams facilitate the problem-solving process. During this process, we set up sub-goals that require visually obtainable information. For example, in restructuring a corporate entity, an organizational chart would be used intermittently in the process. The user would divide the tasks into sub-units such as classifying the types of employees, their current positions and their newly appointed positions. The organizational chart would then serve as a reference in performing the restructuring at the various levels of the hierarchy.

Researchers have recently introduced the term 'externalizations' to refer to any display that facilitates the intertwining of external and internal cognitive information for the purpose of solving a problem [Zhang97,SR96]. We make use of our environment and its properties to enhance our thought and reasoning. On a daily basis we refer to numerous aids in helping us carry out cognitive tasks and functions. In certain cases we would be at a loss without such aids - weather maps to make predictions, a city map or calendar as a reference tool, and architectural plans for analyzing alternate physical solutions. Some of these are designed to allow us to make rapid inferences from which we can make split second decisions. Others empower us with accuracy when performing a visual search for some kind of information. Regardless of what the purpose of the representation, they are used as tools that enhance cognitive processes externally as well as internally.

A study by Larkin and Simon [LS87], illustrated reasons as to why diagrams can be effective. In their study they set up a range of physics problems, and compared the efficiency of solving these problems with and without diagrams. They looked at the effort of search, recognition, and inference with diagrammatic representations versus sentential descriptions of the problem. They found that diagrams can be useful in three major ways: (a) searching for elements is reduced as diagrams group together related information; (b) matching symbolic labels is avoided as diagrams typically use location to group information about a single element; (c) perceptual inferences which are automatically supported by diagrams can be read directly. They concluded that the advantages of using diagrams are computational since the representations facilitate indexing of information that result in useful and efficient computational processes.

In their investigation on the use of diagrams for solving problems in physics, Priest et. al [PL92] found that subjects can solve problems more rapidly if there is a possible sketch to the problem (they refer to such problems as homogeneous). They also found that in the problem solving process, contradictory information is more likely to be detected and reported when the problems are homogeneous. Bauer and Laird [BL93] have reported results of deductive reasoning problems using diagrams. They compared the error rate and response times of subjects in answering questions based on a two-branch electrical circuit versus a logically equivalent verbal question. In their experiment subjects were more accurate and faster answering questions using diagrams. They attributed this advantage to the fact that diagrams immediately present all the alternative possibilities, especially in situations where the reasoning involves branching into several paths.

Spatial reasoning is another important cognitive task that is common in our daily decisions [GP92]. While visual representation allows the extraction of information such as size, shape, or color, spatial reasoning consists of making inferences from spatial properties of a representation, for example making inferences from the relative location of objects within an image. Forbus [FNF91] suggests that spatial representation consist of two parts: a metric diagram, which includes quantitative information which assists perceptual processing; and a graphic vocabulary which makes distinctions in shape and space relevant to the task at hand. He suggests that in order to make diagrams a good communication medium for different representations, it is useful to define the graphic vocabulary in terms of elements of the diagram.

Although diagrammatic representations can enhance cognitive tasks, they can also introduce problems that are not always present in verbal or symbolic representations. One such problem is that logical identity of diagrammatic presentations of a given problem does not imply that it is equally easy to infer the same relationships from different types of representations for the same information. Representations may be equivalent by the knowledge embedded in them without being equivalent in the power and speed of the inference processes they enable. They may be informationally equivalent without being computationally equivalent [LS87,Funt80].

In certain instances, a similar diagrammatic representation is used to aid in solving non-similar problems. The task of the designer of the visual representation is to carefully

select which representation is best suited for the task. This involves investigating the characteristics of the person who is using the diagram, the task they are to perform with the diagram, and the context of diagram use. Perhaps diagrams function not merely to extend problem-solving capacity, but rather present the important information in a particularly usable form. If they do, then understanding the processes behind the human visual system warrants special attention [Ware99,Lowe94]. As such, looking at different theories of perception can give us clues into adapting a form of representation fitted for diagramming. This is the main theme discussed in chapters 5, 6 and 7.

3.1.1) *Metaphors*

Metaphors can play a significant role in assisting the user of the visualization in building a cognitive model. We use metaphors whenever we speak or think about abstract ideas. They serve as natural tools that take concepts we are familiar with and use those to translate and give structure to the abstract [LJ80]. We are usually unaware of the extent to which we use metaphors, however they come up on a daily basis.

Metaphors in the literary realm refer to the use of words or phrases in a figurative manner. They are applied to a given context in which the words do not carry the meaning they are originally intended for. In information visualization, metaphors refer to the transfer of meaning from one context (source) to another (target). They help the user relate the abstract display to something more familiar. For example, in a primitive way, hierarchical structures as described in the previous chapter are defined with characteristics of trees we see in nature. For example, connection between levels in a

hierarchy are referred to as branches, the origin of the hierarchy is referred to as the root node, and leaf nodes make up the lowest level in the hierarchy from which no branches can stem. To display abstract hierarchical information, we resort to describing them as trees.

There are a number of displays that have been designed using metaphors derived from nature. Ploix [Ploix96] developed a software visualization tool mimicking the arrangement and movements of the planetary systems. The use of the planetary metaphor was also used by Spence [Spence00] to illustrate the idea of mapping notes of a musical instrument to planets. The notes could be carefully arranged along the elliptical orbit of the planets and when in movement would produce music for a given composition. Another metaphor is that of a data landscape. This has been used in Themescape [WTPLPS95], with the metaphors of valleys used for separating themes and high ranges used for showing the relevance of a theme in a given set of documents. ThemeRiver [HHN00] is also a visualization tool for mapping changes of themes in a document over a period of time. A study of ThemeRiver showed that subjects felt comfortable with the metaphor of a flowing river and performed better than they would have without it [HHN00]. Skupin [Skupin00] provides a perspective of a cartographer with respect to visual metaphors relating to geography and cartography. By applying a Sunflower metaphor, Rose [Rose99] was capable of facilitating users in searching for domain and task specific knowledge. The Magic Lens [BSPBR93] uses the metaphor of a magnifying lens to reveal detailed information in a document. Lifestreams [FF95] provides a

metaphor for organizing our library of electronic documents based on traditional methods of filing documents.

Metaphors are also useful in helping us detect patterns. If a familiar figure is represented with anomalies, then the user is quick to spot them. A technique developed by Chernoff, called Chernoff faces [Chernoff71], is an example in which multiple attributes of a data set are mapped to features of the face: nose, eyebrows, or faces. If, for example, temperature was mapped to the mouth, then a long mouth would indicate a raise in temperature. Facial features are familiar to a user, even at an early stage in our visual development [Ware99]. So, for example, if data were mapped to a face, in which the length of the mouth overruns the contour of the face the user can quickly make an inference on such an anomaly in the data.

Metaphors have their own advantages and disadvantages. Attempting to combine as many of their strengths while incorporating as few of their weaknesses as possible is likely to prove a challenging task. Erickson outlines the stages of evaluating metaphors once we come up with them [Erickson91]. In choosing the mapping from the domain of the source to the target, we need to ask: how much structure does the metaphor provide?, what is the applicability of the structure? is the metaphor easy to represent?, and, is the metaphor extensible? When metaphors are used, the visualization should make full use of the structure inherent in it. It enhances the utility of the system if it gives users realistic expectations about what inferences on the data can be made. This can help in strengthening areas of a display that require intuitive analysis of the information [GW96].

Having considered the high-level cognitive context for the use of diagrams we now turn our attention to the more purely perceptual issues that are the primary focus of this thesis.

3.2 Graphical Features

In an information display, the visual marks (consisting of points, lines, or areas) can be given a set of visual properties based on the data attributes [Bertin83]. These include color, elementary shape, size, orientation, transparency, texture and position. Allowing data to be mapped to these graphical properties can greatly enhance the data density of a display. From research into human perception [GL95,Treisman86,TG88,LH88], observations [Cleveland85,Wilkinson99], and developments in the fields of visual arts and information design [Albers87,Gombrich65], we now better understand the role and functioning of these visual properties in human subjects.

3.2.1) Color

This graphical feature is probably the most widely coding technique used in visualization systems. Until recently, it has not been common to encode entity-relationship diagrams with color. This was partially due to the scarcity and high costs of color paper printouts. Being a flexible property to map onto displays, mapping color also presents some drawbacks, such as crossing cultural gaps, contradicting conventional mappings, and bias found in personal preferences.

3.2.2) Size

Size can be defined in terms of length, area or volume. Size is mapped to quantitative amounts, strength, and degree of relevance. In ThemeScapes [WTPLPS95], for example,

the stronger the relevance of a theme in a set of documents, the higher will be the peaks in the landscape. In the Treemaps visualization system [JS91], the larger nodes represent subtrees with more content.

3.2.3) Orientation

A mark or node can assume an infinite number of orientations. However we are sensitive to a limited set of variations [Bertin83] and rotations of about 30° or more create clear distinctions [Kosslyn93]. In node-link diagrams, orientation does not play a major role. In several visualization systems, one dimension of the data is mapped to the orientation of the graphical marks [KA00].

3.2.4) Transparency

Transparency is used for displaying objects contained within others or for showing overlapping layers in displays. In the case of containment, transparency immediately creates hierarchical representations of the data and is more common in 3D displays. NV3D uses transparency to display nodes contained within the expanded nodes [WF94]. This visual property needs to be mapped carefully such that the scene's background does not interfere with effects of transparency.

3.2.5) Texture

We think of textured objects in terms of the roughness of the surface, or visually in terms of the type of pattern mapped on to the surface. Two constituents of texture are its granularity and its pattern. Textures are recently being introduced into visualization systems and tremendous research potential exists in methods for mapping attributes of multi-dimensional data onto texture [WK92,WK95,HE98].

3.2.6) Positioning

Apart from using low-level perceptual properties to code attributes of nodes and links, their spatial arrangements play an important role. Some authors consider that positioning elements, and making use of the space in which the elements reside, is the most important component in visualizations [Cleveland85,MacEachren95]. Several techniques use proximity to cluster nodes of related information for example in a scatter-plot. Proximity is also important in visualizing hierarchical structures. A third way in which proximity is used is to place information or nodes with higher importance in the center of the screen.

3.2.7) Remarks

If a goal of visual analysis is to provide a compact ‘stimulus space’ by representing a large number of variables in a scene, then we need to understand how these properties interact [BF93]. As such, effective displays can be created with 'proper' assignment of data to visual properties so that they do not clutter information but instead highlight patterns that are relevant to the information in question.

Wilkinson [Wilkinson99] classified low-level graphical features into two major groups. According to his scheme the attributes of position, size, shape, and orientation can be categorized as part of the *form* of the structure, and the elements of color, texture, transparency can be categorized as *surface properties* or *attributes* of the structure. In choosing how to combine these visual properties in the spatial domain, the display must also take into consideration the differences in the perception of visual properties among individuals. This can constitute a dominant factor in selecting the variables needed for mapping elements of information.

3.3 Graphical Patterns and Principles of Gestalt

The choice of using nodes and links for representing entity-relationship models has certain perceptually favorable qualities. The patterns that are created from the arrangement of elements in these displays can be justified by considering Gestalt theories of perception.

In the early 1900's a school of psychology was established to study the way we perceive patterns [Koffka35]. Max Wertheimer, who was later joined by Kurt Koffka and Wolfgang Kohler, founded the school. These theorists were intrigued about the idea of how our mind perceives wholes out of incomplete elements. The set of laws they developed is being used today in many areas such as the visual arts [Arnheim69, Behrens84], graphic design [Arnston98], musicology [Lippe95], and architecture, as they provide a clear explanation of some of our basic perceptual phenomena. These laws explain why certain representations for structured information facilitate intuitive reasoning.

3.3.1) *Pragnanz*

Pragnanz or "good figure" is seen as the central principle of Gestalt theory [Koffka35]. It explains that in order to get a sense of the image the entire object needs to be examined. As a result the structure that is perceived from the image is seen as simply as possible (Figure 3.1).

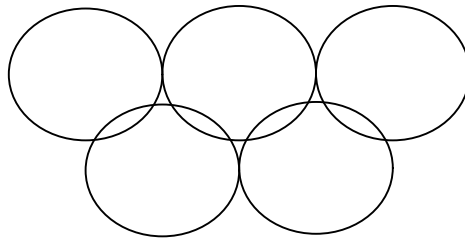


Figure 3.1 - We see 5 rings instead of 9 separate objects.

3.3.2) Similarity

The principle of similarity states that objects which share visual characteristics, such as shape, size, color, texture, value or orientation, will be seen as belonging together (Figure 3.2).

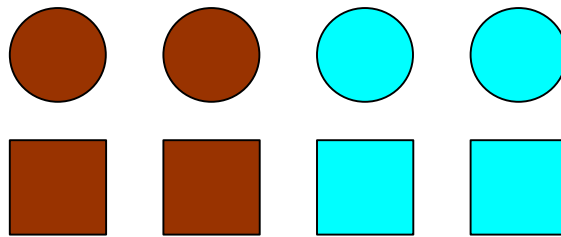


Figure 3.2 - We can group the object based on the similarity of their shapes, or based on the similarity of color.

3.3.3) Continuity

This principle states that we are likely to group simple and continuous curves rather ones that contain abrupt changes in direction (Figure 3.3).

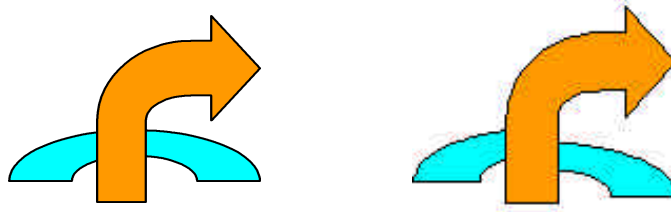


Figure 3.3 - To the left we see a curved arrow in front of a thick continuous segment. To the right we see three separate objects.

3.3.4) Connectedness

Although not originally included in the principles of gestalt, connectedness is considered to be a strong organizing principle. In certain cases, connected objects can portray a grouping better than any of the other principles. In Figure 3.4, the pair of connected circles can be seen as a unit. Experiments have shown it to be a stronger organizing principle than color, size, shape or proximity [Palmer94].

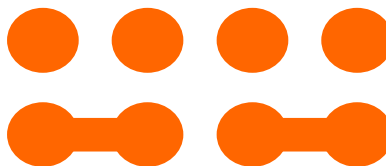


Figure 3.4 - The grouping of the units in the bottom appear stronger than that of the top units. In the bottom we tend to see only two units vs four on the top.

3.3.5) Proximity

The principle of proximity states that things which are closer together will be seen as belonging together. Looking at the picture below (Figure 3.5) although we could group

by shape, we tend to group by proximity and as a result see four groups of circle and shape pairs.

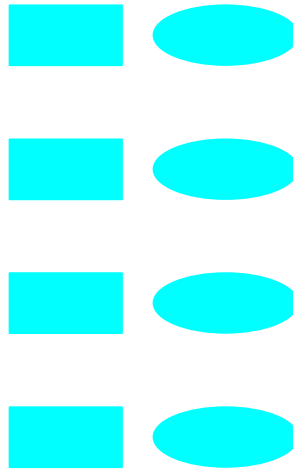


Figure 3.5 - Grouping tends to occur based on proximity of objects.

3.3.6) Symmetry

Despite the pressure of proximity to group the brackets in Figure 3.6 nearest each other together, symmetry overwhelms our perception and makes us see them as pairs of symmetrical brackets.



Figure 3.6 - Pairs of symmetrical brackets are perceived instead of asymmetrical single objects.

3.3.7) Closure

The principle of closure applies when we tend to see complete figures even when part of the information is missing. In Figure 3.7 we see three black circles covered by a white triangle, even though it could just as easily be three incomplete circles [Kanizsa79]. Our minds react to patterns that are familiar, even though we often receive incomplete information. Even though the circle to the right in Figure 3.7 is overlapped by a square, we still perceive a circle due to the principle of closure.

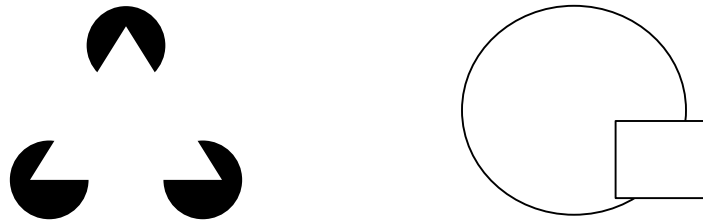


Figure 3.7 - In the image to the left, although a triangle is not explicitly drawn we see one overlapping the circles. To the right even when the boundary is not complete, we still perceive the circle as single completed unit.

3.4 Perceptual Syntax of Node-Link Diagrams

In light of gestalt principles we can justify certain features of node-link diagrams.

Contours and lines that are common in variations of node-link diagrams trigger certain perceptual characteristics that reliably convey certain types of information. Figure 3.8 below highlights several features of node-link diagrams that make use of gestalt principles:

- ❑ *Closed contours* delineate nodes
- ❑ *Connectedness* represents some form of relationship

- *Similarity* is used for separating classes of nodes (in a data flow model, boxes are external entities and circles are processes)
- In carefully laying out nodes, related entities can usually be placed in *proximity* to one another. This property reveals clusters or partial trends.

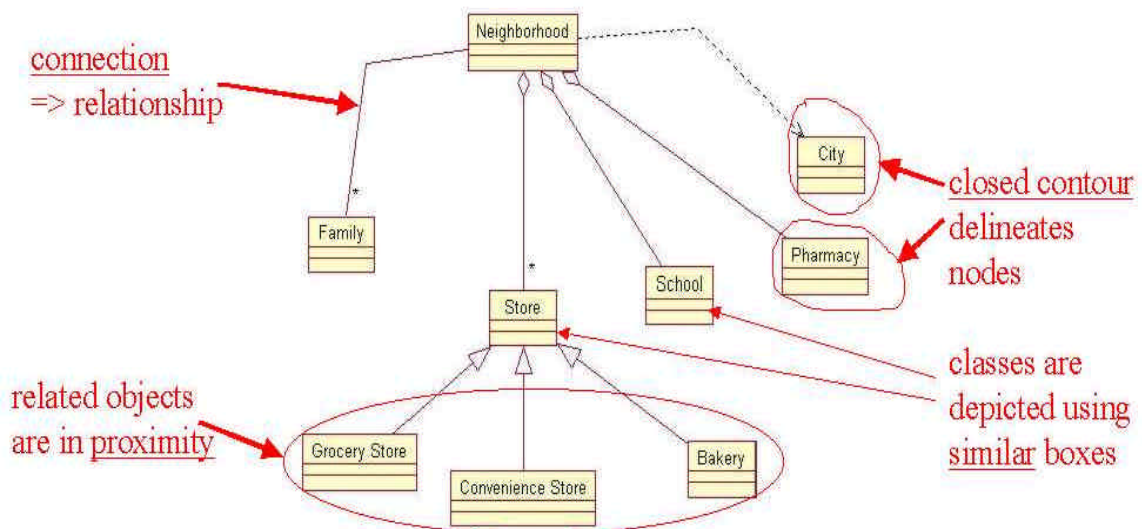


Figure 3.8 - Grammar of a node-link diagram based on gestalt principles.

3.5 Objects and Object Recognition

After preprocessing the Gestalts in a figure, a higher level of our visual system processes the forms and structures of an image. This complex phase is commonly referred to as the object recognition stage. Currently, there are two general classes of theories of object perception. One class emphasizes the properties of the visual image and suggests that we recognize objects based on the similarities of the image they present with the images of previously viewed objects Edelman [Edelman95]. Image-based theories account for the fact that upside-down faces are much harder to recognize than right-side up faces. The other class of theory emphasizes viewpoint independent analysis of object structure. To

illustrate, Figure 3.9 shows two images that are immediately seen as different views of the same object, even though the images are very different, whereas two very similar images are perceived as depicting different objects. To explain this and other similar effects researchers have proposed that the visual system builds structural 3D models. Current evidence suggests that the visual system supports both image based and structural object recognition – they need not be mutually exclusive. Thus aspects of both theories should be taken into account for constructing displays.

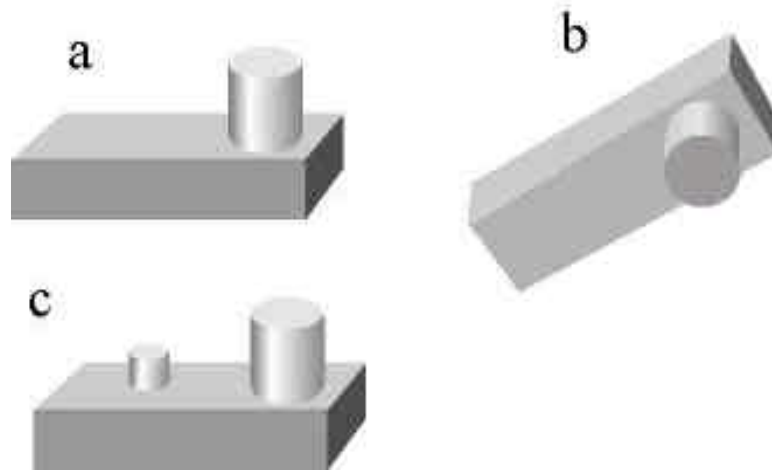


Figure 3.9 - Despite having very different shapes in their images, the objects a and b shown above are readily identifiable as being the same. According to image-based recognition theories a and c should be seen as most similar.

3.5.1) Image Matching

Image matching theories were developed in conjunction with models for solving the problem of character recognition using templates. For each character perceived, there is a similar pattern or template stored in long-term memory facilitating the recognition. Incoming patterns would be matched against the set of templates, and if there were

sufficient overlap between a novel pattern and a template then the pattern would be categorized as belonging to the class captured by that template [BGG97]. A possibly infinite series of templates needs to be stored, and thus a prohibitively large memory capacity is required in order for such matching to occur. Furthermore, template matching cannot easily cope with 3D objects [BH94], where object parts may be occluded from all views stored by the user.

However more sophisticated extensions to the template matching techniques have emerged. Ullman [Ullman89] proposes a scheme in which the recognition process is divided into two stages. The first stage takes the object and determines the necessary transformation in space (using simple rotations, translations, and scaling) to bring the viewed image into a standard form (alignment) and the second stage determines the closest match between the standardized image and one of the stored templates. The second stage of the process involves a search over all possible templates and not over all possible views, since the transformation had been determined in the alignment stage [Ullman89].

Whether using a direct template matching scheme or an alignment method in recognizing objects, surface properties play an important role. A study by Price and Humphreys revealed advantages for the classification as well as for the naming of objects when they were shown with surface properties such as color, texture, and shading [PH89].

3.5.2) Structured Object Perception

The main alternative to image based theories are theories of structured object perception. Figure 3.10 is intended to capture the broad outlines of structural object perception theory. Object recognition is accomplished in a series of stages. At the first stage, the visual image is everywhere analyzed into primitives of edge elements, color and texture. This information is then used to segment the image so that the boundaries of objects can be extracted. Elementary shape from shading information also is used in simple object components [Ramachandran88], or “blobs” [Biederman87]. At the same time a structural skeleton is identified, containing information about how the components are interconnected [Marr82,BG93]. Ultimately, all of the information is combined in object identification. In the following paragraphs we briefly elaborate slightly more on the role of silhouette shape, shading and surface features. A more extensive discussion is provided in the introduction of the chapters that explain the application of these principles in the experiments. These include chapters 5, 6, and 7.

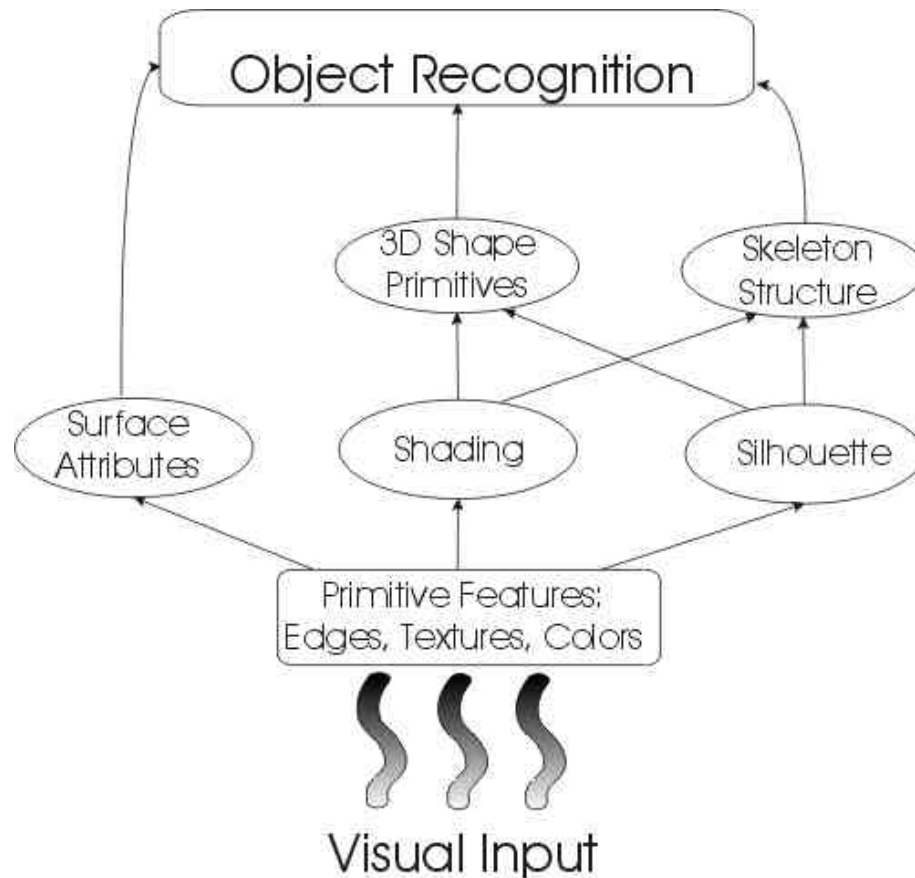


Figure 3.10. Theories of structural object perception propose a series of processing stages, culminating in object recognition.

Silhouettes are especially important in determining how we perceive objects. Children tend to draw objects based on their most salient silhouettes – for example profiles of animals are chosen in preference to other views [Halverston92]. Marr [Marr82] argued that “buried deep in our perceptual machinery” are processes that determine how silhouettes are interpreted. He argued that there are three rules embedded in this perceptual machinery:

- Each line of sight touching a silhouette grazes the surface only once. The set of such points is the contour generator.

- Adjacent points on the contour arise from adjacent points on the viewed object
- All of the points on the contour generator lie on a single plane. The idea of a contour generator is illustrated in Figure 3.11.

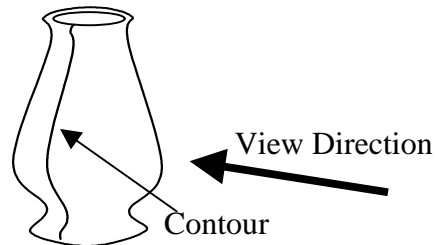


Figure 3.11 - According to Marr the perceptual system makes assumptions that occluding contours are smoothly connected and lie in the same plane at right angles to the line of sight. Adapted from [Marr82].

Marr and Nishihara [MN78] proposed that concave sections of the silhouette contour are critical in defining how different parts of an object are segmented. Figure 3.12 illustrates a crudely drawn animal that we nevertheless readily perceive as having distinct head, legs, torso, and tail parts. They also proposed a mechanism whereby the axes of the parts become cognitively connected to draw a structural skeleton.

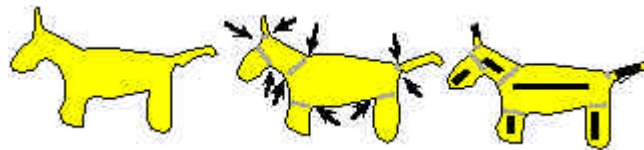


Figure 3.12 - According to Marr and Nishihara [NH78] concave sections of the silhouette define sub-parts of the object. These points are critical in defining a structural skeleton. Adapted from Marr and Nishihara [NH78].

According to Marr, the 3D primitives that are identified are a set of generalized cones. Biederman developed a more elaborate set of 36 3D primitives he called “Geons”

[Biederman87]. These are also defined by image properties on the silhouette according to a set of rules based on co-linearity, symmetry, parallelism, curvature and co-termination (the contours meet at a point, e.g. a cone) of parts of the silhouette. Some of Biederman's Geons are illustrated in Figure 3.13. The introduction to Chapter 4 gives a more detailed description of how Biederman derived his particular set of Geons.

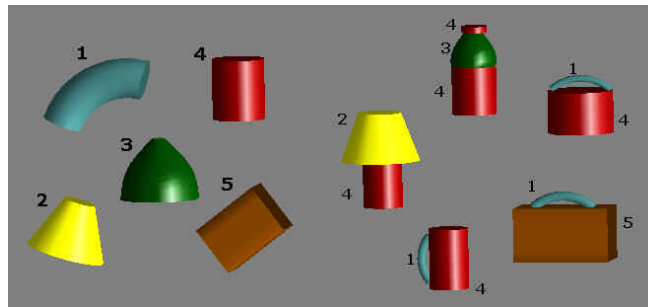


Figure 3.13 - (a) Geons are object primitives in Biederman's theory. (b) When connected in a particular structural relationship they can define an object. (c) Different connections of the same Geons can result in different objects as the figure shows.

Biederman's concept of a structural skeleton is more elaborate than that of Marr, although both point out that different combinations of Geons can result in different objects being recognized. Biederman identified that relationships between Geons are particularly significant. For example, information such as whether Geon A is "on-top-of" or "to-the-right-of" Geon B makes a difference whether a person or an animal is perceived if one is the head Geon and another is a torso Geon. The relevance of Geon relationships also known as perceptual semantics is further elaborated in Chapter 7.

3.5.3) Shape-From-Shading

Elementary shape-from-shading information is also important in the recognition of an object's structural components. Enns and Rensink [ER90b] showed that simple shading information can be processed pre-attentively suggesting that it is extracted very early in human visual processing. Ramarchandran [Ramachandran88] has shown that shading can combine to strongly influence the shape of an object. Figure 3.14 shows how shading information can cause two objects with the same silhouette to be interpreted quite differently. This shows that shape from shading processes feed both into the 3D shape primitives and into the definition of a structural skeleton. This discussion of the importance of shading information is discussed in Chapter 6.



Figure 3.14 - The same object silhouette, a rectangle, can result in two very differently structured objects as shown above- and below-left. The structural skeletons of the shaded version can be clearly seen.

There is evidence from a number of sources that surface color and surface texture are processed relatively independently from the object's structural description. According to theory, these surface attributes are considered to be secondary properties of objects [Triesman80, Biederman87]. Thus, in Figure 3.10, the basic information about color and texture by-passes the processes involved in forming 3D primitive and building structural description, only becoming associated with the object at the final stage.

3.6 Perceptual Studies of 3D Displays

Previous studies of the effectiveness of 3D diagrams have reported mixed results. Lee and MacLahan [LM96] have shown that reducing the number of diagrams by using the added dimension in a 3D presentation could result in efficient decision making. They compared the accuracy and speed of decision making using stereoscopic 3D displays of scattergrams and block diagrams. They found that 3D scattergrams increased accuracy and facilitated faster decision making, as more 2D scattergrams were required to present the equivalent information of a single 3D scattergram. Ware and Franck [WF96] used a path-tracing task to evaluate 3D versus 2D node-link diagrams. They found a 60% advantage with stereoscopic viewing and a 130% advantage with motion parallax depth cues.

On the other hand, studies of bar and pie charts suggest that 3D versions provide no advantage. For example 3D block diagrams did not have any effects on accuracy or speed of decision making compared to 2D tabular reports [LM86], and if the 3D features are overly decorative this may be detrimental [Carswell et al 1991]. To measure the

effectiveness of recalling information from 3D displays, Watson and Driver [WD83] presented 3D perspective graphs and 2D tabular displays to business students. They did not find any difference between recall for the 3D perspective graphs compared to the 2D tabular presentations.

However, there may be ways of mapping information to 3D solid components that provides a better match to human perceptual mechanisms particularly if we can take into account the perceptual mechanisms of perceiving 3D structured objects.

3.7 Discussion

The prevalent visualization methods available have indirectly made limited use of cognitive and perceptual principles. Gestalt principles have been applied to some extent to enrich node-link diagram information and more can still be applied. However, in many cases notations have evolved from one system to another without consideration for their perceptual effectiveness.

Recent findings of research in perception theories explain the capacity of the visual sensory system in identifying objects. A prevalent theory among theories of structural object recognition is the hypothesis of recognition-by-components. If we extract structural information by first decomposing a scene into components, then we might be able to develop visualizations from these structural primitives that have been engrained in our perceptual processes. The potential exists for applying these theories. Mapping Geon

primitives to nodes and links can potentially reveal structures more effectively. This is the central idea developed throughout the remainder of the thesis.

Chapter 4 - The Geon Diagram: Definition and Construction

Designing for perception means that visualizations should present information in such a way that important information is automatically extracted by the visual system. Thus if we have visual machinery to extract 3D primitives and a structural description from a scene, using these primitives might facilitate information extraction from a visualization of data. The Geons proposed by Biederman present a rich set of shapes that can be used to represent different classes of entities in a node-link diagram. Relationships can be represented by the way that Geons are attached directly to one another. Alternatively, elongated Geons can be used to show relationships. The secondary attributes of entities and relationships can be mapped to the secondary visual attributes of Geons, namely color and surface texture. In this chapter, the Geon diagram is defined by a set of rules for applying the theoretical principles of Biederman to the problem of drawing diagrams.

This chapter also describes the toolkit developed for creating Geon diagrams. Several 3D drawing toolkits were tested to determine whether they were suitable for the research, such as Simply 3D™ (Micrografx) or Alice™. However, certain capabilities such as transparency and drawing particular solid geometric shapes were not easily supported, which motivated the implementation of the Geon toolkit. All the features currently available in the Geon toolkit are primarily developed for the purposes of the research outlined in the thesis.

4.1 The Geon Diagram

The following rules define the “Geon diagram”. Biederman’s term “Geon” is used, as it nicely describes the idea of a 3D shape, although the particular set of 3D shape primitives in Biederman’s theory is not entirely endorsed. Five rules relating to the use of Geons as 3D primitives are defined, as well as three additional rules that relate to the layout of the Geon structure.

G1: Major entities of a system should be presented using simple 3D shape primitives (Geons).

G2: The links between entities can be represented by the connections between Geons. Thus the Geon structural skeleton represents the data structure.

G3: Minor sub-components are represented as Geon appendages – small Geon components attached to larger Geons.

G4: Secondary attributes of entities and relationships are represented by Geon color and texture and by symbols mapped onto the surfaces of Geons.

G5: Geons should be shaded to make their 3D shape clearly visible.

Although Geons are 3D shape primitives, theories of shape extraction rely heavily on a clear silhouette. For this reason a good 2D layout will also be important in determining how easily a Geon structural description can be identified. Thus the following layout rules are added.

L1: All Geons should be visible from the chosen viewpoint.

L2: The Geon diagram should be laid out predominantly in the plane orthogonal to the view direction.

L3: Junctions between Geons should be made clearly visible.

From the above it can be seen that Geon diagrams are a kind of hybrid of 3D and 2D rules. They incorporate 3D information in the shapes used to represent information, but they are meant to be laid out, as much as possible, in the 2D plane orthogonal to the line of sight. In the chapters that follow, additional rules are added to the list above as the research that supports them is described.

4.2 High-level Architectural Description of the Toolkit

A toolkit using the OpenGL 3D graphics standard was created to explore the effectiveness of Geons in node-link diagrams. The Geon toolkit was built with a simple and extendible architecture for the purposes of the research outlined in the thesis. It is

composed of two main components: the interface component and the Geon component. The interface component manages the user input. It also manages the presentation of the diagrams via the OpenGL drawing pipeline. The architecture of the OpenGL pipeline is described in [WNDS97]. The Geon component primarily manages the drawing routines for all the Geon primitives. Figure 4.1 below describes the interaction between the Geon component and the interface component. Geon primitives that are selected by the user are automatically rendered by the OpenGL pipeline once the Geon Draw Routine provides the adequate parametric description for the Geon.

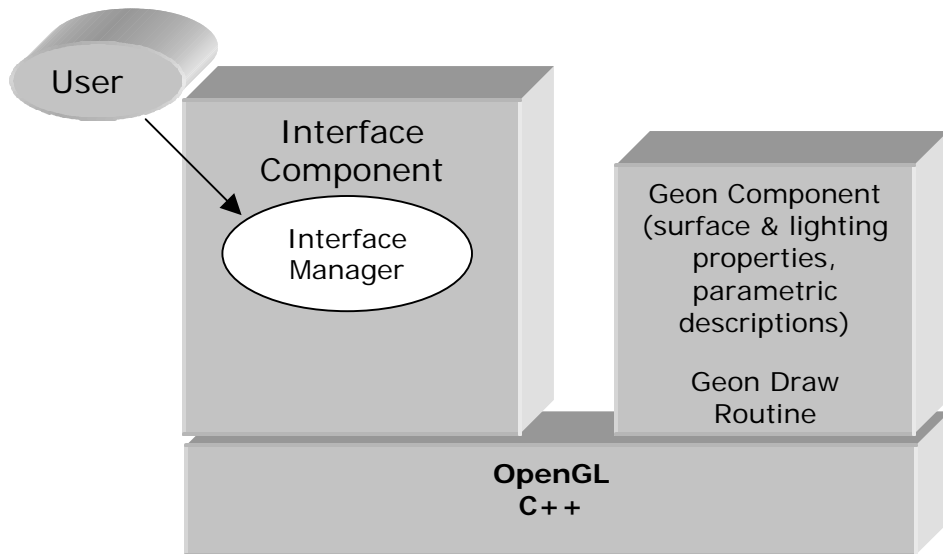


Figure 4.1 - Architectural overview of the Geon toolkit. User input is managed by the Interface Component, and Geon rendering is managed by the Geon Component and Draw Routine interacting with OpenGL.

4.3 Functional Description of the Toolkit

Using the toolkit we can create Geon diagrams according to the principles given previously. At present the Geon toolkit is designed to be a research tool rather than a diagramming utility and as such lacks many of the features essential in a general-purpose system. This section describes the functional aspects of the toolkit that facilitate the creation of Geon diagrams.

The toolkit has the following capabilities:

- Facilitates the use of a set of 24 Geons for building diagrams,
- Facilitates the modification of Geon length and radii sizes,
- Maps surface properties such as color, texture, shading, transparency, and labeling onto Geons,
- Facilitates positioning Geons to create diagram structures,
- Enables saving and opening Geon diagram files.

4.3.1) Creating Geons using Parametric Descriptions

In order to produce Geons, a requirement was to develop parametric forms for the list of 24 Geons. The parametric forms were developed for 24 Geons such that all the volumetric solids would be drawn from the same number of polygons. This approach reduces the complexity of the drawing routine and subsequent operations on the Geons such as resizing, texture mapping and calculating surface normals.

Geon primitives are derived from variations of two or three levels of what Biederman has called the non-accidental properties described in the 2D plane - co-linearity, symmetry, parallelism, curvature, and co-termination [Biederman87]. They are termed non-accidental since they are unlikely to be a consequence of an accident of a viewpoint or of an alignment of edge and eye. In general terms a Geon is the surface swept out by a cross section moving along an axis.

Figure 4.2 shows the generation of the set of 36 ($2 \times 2 \times 3 \times 3 = 36$) Geons from contrasts in the nonaccidental relations of four attributes of generalized cylinders. Three of the attributes specify characteristics of the cross section (its curvature, size, and symmetry) and one for the axis (its curvature).

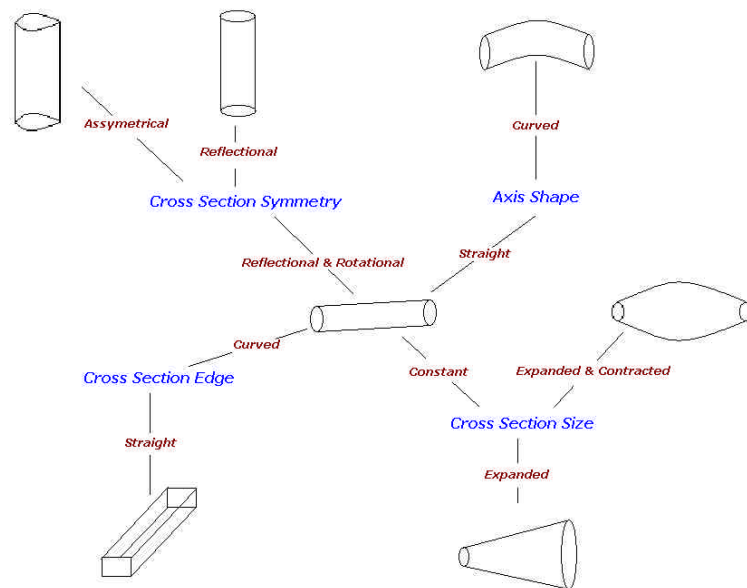


Figure 4.2 - Attributes of Geon cross section and axis. Adapted from [Biederman87].

The figure illustrates how variations in three attributes of a cross section and one attribute of the shape of the axis can generate a set of generalized cones differing in non-accidental relations. An axis (referred to as the central axis) can be straight or curved and can thus generate tubes or bent tubes. The cross section can be round (curved) or rectangular (straight). Cross section size can be constant, expanded, or expanded and contracted along the central axis. Constant sized cross sections have parallel sides; expanded or expanded and contracted cross sections do not have parallel sides. Cross section can have reflectional symmetry, reflectional and rotational symmetry, or be asymmetrical. Rotational symmetry does not change the shape of the object under a 90° rotation. For example, a circle has rotational symmetry while an ellipse has reflectional symmetry. By these properties Biederman defines the entire set to be made of 36 volumetric primitives. The toolkit only includes 24 Geons from this subset by not including those Geons with asymmetrical cross sections.

Twenty-four Geons in the toolkit were drawn using a parametric representation with additional inputs to allow for variations. For the most part the parametric representations are given in terms of the cross section (or radius) and the axis (length) of the Geon. The following sections describe the parametric representation used for the 24 Geons in the toolkit. The first 12 represent Geons with curved cross sections. They are ordered in triplets by the size of the cross section (constant, expanded, expanded and contracted). Each triplet contains the Geons with same cross section symmetry.

The parameters used for each equations are:

l - axis length

θ - sweep angle for the radius

r - cross section radius

r_1 - cross section radius one

r_2 - cross section radius two

a_y - amplitude of the Cosine function for Geons with expanding and contracting cross sections

c - constant for reflectional symmetry; $c = 2$ was used in the toolkit

β - constant angle slice for Geons with curved axis

i - number of slices in the y-direction

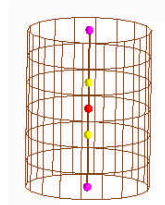
Geon 1: Axis - Straight; Cross Section Edge - Curved; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Constant

This object is simply a cylinder. Its parametric representation is the following:

$$x(l, \theta) = r \cos(\theta)$$

$$y(l, q) = l$$

$$z(l, q) = r \sin(q)$$



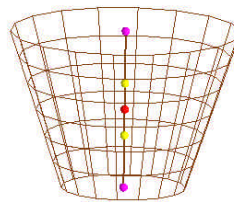
Geon 2: Axis - Straight; Cross Section Edge - Curved; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Expanded

This object is a conical frustum. Its parametric representation is the following:

$$x(l, \theta) = (r_1 + (r_2 - r_1) * l / i) * \cos(\theta)$$

$$y(l, \theta) = l$$

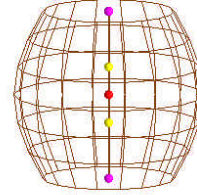
$$z(l, \theta) = (r_1 + (r_2 - r_1) * l / i) * \sin(\theta)$$



Geon 3: Axis - Straight; Cross Section Edge - Curved; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Expanded & Contracted

This Geon resembles a barrel. Its parametric representation is:

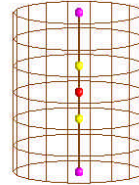
$$\begin{aligned}x(l, \theta) &= (r + (a_y * \cos(l))) * \cos(\theta) \\y(l, \theta) &= l \\z(l, \theta) &= (r + (a_y * \cos(l))) * \sin(\theta)\end{aligned}$$



Geon 4: Axis - Straight; Cross Section Edge - Curved; Cross Section Symmetry - Reflectional; Cross Section Size - Constant

This Geon representation is similar to the cylinder with the only difference being that it has an elliptical cross section. In the toolkit the variable c controls the ratio between the major and minor axes of the ellipse. Its parametric representation is the following:

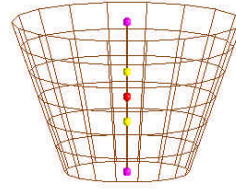
$$\begin{aligned}x(l, q) &= r \cos(q) \\y(l, q) &= l \\z(l, q) &= r \sin(q) / c\end{aligned}$$



Geon 5: Axis - Straight; Cross Section Edge - Curved; Cross Section Symmetry - Reflectional; Cross Section Size - Expanded

This is the same as Geon 2 with an elliptical cross section. Its parametric representation is the following:

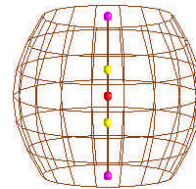
$$\begin{aligned}
 x(l, q) &= (r + (r_2 - r_1) * l/i) * \cos(q) \\
 y(l, q) &= l \\
 z(l, q) &= (r + (r_2 - r_1) * l/i) * \sin(q) / c
 \end{aligned}$$



Geon 6: Axis - Straight; Cross Section Edge - Curved; Cross Section Symmetry - Reflectional; Cross Section Size - Expanded & Contracted

This Geon resembles a barrel with an elliptical cross section. Its parametric representation is:

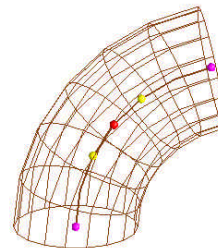
$$\begin{aligned}
 x(l, q) &= (r + (a_y * \cos(l))) * \cos(q) \\
 y(l, q) &= l \\
 z(l, q) &= (r + (a_y * \cos(l))) * \sin(q) / c
 \end{aligned}$$



Geon 7: Axis - Curved; Cross Section Edge - Curved; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Constant

This object is a cylinder but curved along its axis. Its parametric representation is different than that for the cylinder on the x- and y-values. The parametric form is:

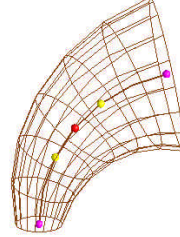
$$\begin{aligned}
 x(l, q) &= \cos(180 - b) * (l - r \cos(q)) \\
 y(l, q) &= \tan(180 - b) * x \\
 z(l, q) &= r \sin(q)
 \end{aligned}$$



Geon 8: Axis - Curved; Cross Section Edge - Curved; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size: Expanded

This is a conical frustrum with a curved axis. Its parametric representation is the following:

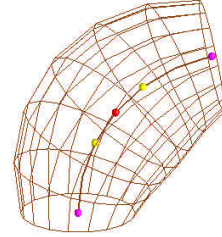
$$\begin{aligned}x(l, q) &= \cos(180 - \mathbf{b}) * (l - (r + (r_2 - r_1) * l/i)) * \cos(q) \\y(l, q) &= \tan(180 - \mathbf{b}) * x \\z(l, q) &= (r + (r_2 - r_1) * l/i) * \sin(q)\end{aligned}$$



Geon 9: Axis - Curved; Cross Section Edge - Curved; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Expanded & Contracted

This Geon resembles a bent barrel. Its parametric representation is:

$$\begin{aligned}x(l, q) &= (r + (a_y * \cos(l)) + ((r_2 - r_1) * l/i)) * \cos(q) \\y(l, q) &= \tan(180 - \mathbf{b}) * x \\z(l, q) &= (r + (a_y * \cos(l)) + ((r_2 - r_1) * l/i)) * \sin(q)\end{aligned}$$



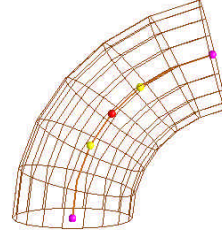
The next three Geons are different than those derived in 7), 8), and 9) in that they all have elliptical cross-sections.

Geon 10: Axis - Curved; Cross Section Edge - Curved; Cross Section Symmetry - Reflectional; Cross Section Size - Constant

The Geon is the same as in 7 with z-value being half that of the one derived in Geon 7.

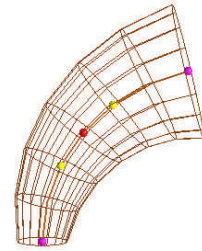
The parametric form is:

$$\begin{aligned}
 x(l, q) &= \cos(180 - b) * (l - r \cos(q)) \\
 y(l, q) &= \tan(180 - b) * x \\
 z(l, q) &= r * \sin(q) / c
 \end{aligned}$$



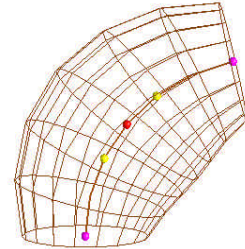
Geon 11: Axis - Curved; Cross Section Edge - Curved; Cross Section Symmetry - Reflectional; Cross Section Size - Expanded

$$\begin{aligned}
 x(l, q) &= \cos(180 - b) * (l - (r + (r_2 - r_1) * l/i)) * \cos(q) \\
 y(l, q) &= \tan(180 - b) * x \\
 z(l, q) &= (r + (r_2 - r_1) * l/i) * \sin(q) / c
 \end{aligned}$$



Geon 12: Axis - Curved; Cross Section Edge - Curved; Cross Section Symmetry - Reflectional; Cross Section Size - Expanded & Contracted

$$\begin{aligned}
 x(l, q) &= (r + (a_y * \cos(l)) + ((r_2 - r_1) * l/i)) * \cos(q) \\
 y(l, q) &= \tan(180 - b) * x \\
 z(l, q) &= (r + (a_y * \cos(l)) + ((r_2 - r_1) * l/i)) * \sin(q) / c
 \end{aligned}$$



Geon 13: Axis - Straight; Cross Section Edge - Straight; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Constant

This Geon is the unit box if $h = 1$ and $r = 1$. Its parametric representation is the following:

for $45 < q < 135$

$$x(l, q) = r * \cos(q)$$

$$y(l, q) = l$$

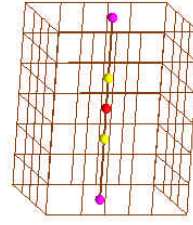
$$z(l, q) = r$$

for $q \geq 45$ & $q \leq 135$

$$x(l, q) = r * \tan(q)$$

$$y(l, q) = l$$

$$z(l, q) = r * \tan(q)$$



Geon 14: Axis - Straight; Cross Section Edge - Straight; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Expanded

This Geon represents a pyramidal frustrum. Its parametric representation is the following:

for $45 < q < 135$

$$x(l, q) = (r + (r_2 - r_1) * l/i) * \cos(q)$$

$$y(l, q) = l$$

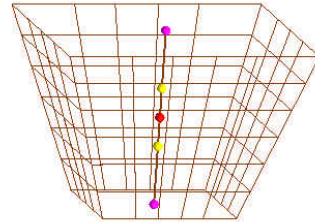
$$z(l, q) = (r + (r_2 - r_1) * l/i)$$

for $q \geq 45$ & $q \leq 135$

$$x(l, q) = (r + (r_2 - r_1) * l/i) * \tan(q)$$

$$y(l, q) = h$$

$$z(l, q) = (r + (r_2 - r_1) * l/i) * \tan(q)$$



Geon 15: Axis - Straight; Cross Section Edge - Straight; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Expanded & Contracted

This Geon resembles a barrel with a square cross-section. Its parametric representation is:

for $45 < q < 135$

$$x(l, q) = (r + (a_y * \cos(l))) * \cos(q)$$

$$y(l, q) = l$$

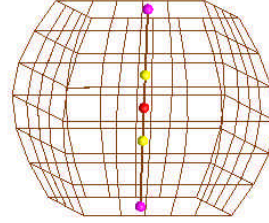
$$z(l, q) = (r + (a_y * \cos(l)))$$

for $q \geq 45$ & $q \leq 135$

$$x(l, q) = (r + (a_y * \cos(l))) * \tan(q)$$

$$y(l, q) = l$$

$$z(l, q) = (r + (a_y * \cos(l))) * \tan(q)$$



Geon 16: Axis - Straight; Cross Section Edge - Straight; Cross Section Symmetry - Reflectional; Cross Section Size - Constant

This Geon is the unit rectangle if $h = 1$ and $r = 1/c$. Its parametric representation is the following:

for $45 < q < 135$

$$x(l, q) = r * \cos(q)$$

$$y(l, q) = l$$

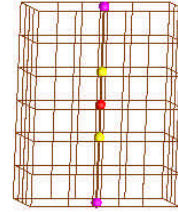
$$z(l, q) = r / c$$

for $q \geq 45$ & $q \leq 135$

$$x(l, q) = r * \tan(q)$$

$$y(l, q) = l$$

$$z(l, q) = r * \tan(q) / c$$



Geon 17: Axis - Straight; Cross Section Edge - Straight; Cross Section Symmetry - Reflectional; Cross Section Size - Expanded

This Geon represents a pyramidal frustrum with mirror symmetry. Its parametric representation is the following:

for $45 < q < 135$

$$x(l, q) = (r + (r_2 - r_1) * l/i) * \cos(q)$$

$$y(l, q) = l$$

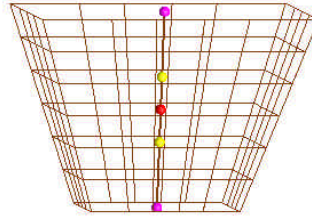
$$z(l, q) = (r + (r_2 - r_1) * l/i) / c$$

for $q \geq 45$ & $q \leq 135$

$$x(l, q) = (r + (r_2 - r_1) * l/i) * \tan(q)$$

$$y(l, q) = l$$

$$z(l, q) = (r + (r_2 - r_1) * l/i) * \tan(q) / c$$



Geon 18: Axis - Straight; Cross Section Edge - Straight; Cross Section Symmetry - Reflectional; Cross Section Size - Expanded & Contracted

This Geon resembles a barrel with straight edges and reflection. Its parametric representation is:

For $45 < q < 135$

$$x(l, q) = (r + (a_y * \cos(l))) * \cos(q)$$

$$y(l, q) = l$$

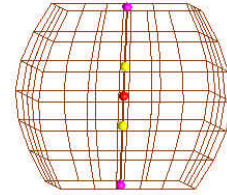
$$z(l, q) = (r + (a_y * \cos(l))) / c$$

for $q \geq 45$ & $q \leq 135$

$$x(l, q) = (r + (a_y * \cos(l))) * \tan(q)$$

$$y(l, q) = l$$

$$z(l, q) = (r + (a_y * \cos(l))) * \tan(q) / c$$



The next 6 representations are the same as the representations above with the axis curved.

Geon 19: Axis - Curved; Cross Section Edge - Straight; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Constant

This Geon is the unit box if $h = 1$ and $r = 1$. Its parametric representation is the following:

for $45 < q < 135$

$$x(l, q) = \cos(180 - b) * (l - r \cos(q)) * \cos(q)$$

$$y(l, q) = \tan(180 - b) * x$$

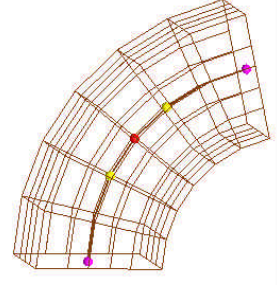
$$z(l, q) = r$$

for $q \geq 45$ & $q \leq 135$

$$x(l, q) = \cos(180 - b) * (l - (r + (r_2 - r_1) * l/i)) * l$$

$$y(l, q) = \tan(180 - b) * x$$

$$z(l, q) = r * \tan(q)$$



Geon 20: Axis - Curved; Cross Section Edge - Straight; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Expanded

This Geon represents a pyramidal frustrum with a curved axis. Its parametric representation is the following:

for $45 < q < 135$

$$x(l, q) = \cos(180 - b) * (l - (r + (r_2 - r_1) * l/i)) * \cos(q)$$

$$y(l, q) = \tan(180 - b) * x$$

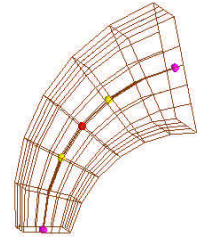
$$z(l, q) = r$$

for $q \geq 45$ & $q \leq 135$

$$x(l, q) = \cos(180 - b) * (l - (r + (r_2 - r_1) * l/i)) * l$$

$$y(l, q) = \tan(180 - b) * x$$

$$z(l, q) = r * \tan(q)$$



Geon 21: Axis - Curved; Cross Section Edge - Straight; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Expanded & Contracted

for $45 < q < 135$

$$x(l, q) = (r + (a_y * \cos(l)) + ((r_2 - r_1) * l/i)) * \cos(q)$$

$$y(l, q) = \tan(180 - b) * x$$

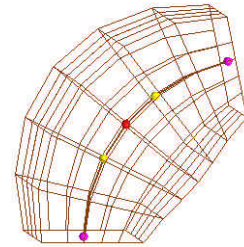
$$z(l, q) = (r + (a_y * \cos(l)))$$

for $q \geq 45$ & $q \leq 135$

$$x(l, q) = (r + (a_y * \cos(h)) + ((r_2 - r_1) * h/i)) * \tan(q)$$

$$y(l, q) = \tan(180 - b) * x$$

$$z(l, q) = (r + (a_y * \cos(h))) * \tan(q)$$



The last three Geons are the same as above with the z-value being reduced by a factor.

Geon 22: Axis - Curved; Cross Section Edge - Straight; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Constant

for $45 < q < 135$

$$x(l, q) = \cos(180 - b) * (l - r \cos(q)) * \cos(q)$$

$$y(l, q) = \tan(180 - b) * x$$

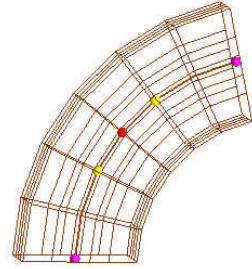
$$z(l, q) = r / c$$

for $q \geq 45$ & $q \leq 135$

$$x(l, q) = \cos(180 - b) * (l - (r + (r_2 - r_1) * l/i)) * \cos(q)$$

$$y(l, q) = \tan(180 - b) * x$$

$$z(l, q) = r * \tan(q) / c$$



Geon 23: Axis - Curved; Cross Section Edge - Straight; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Expanded

for $45 < q < 135$

$$x(l, q) = \cos(180 - b) * (l - (r + (r_2 - r_1) * l/i)) * \cos(q)$$

$$y(l, q) = \tan(180 - b) * x$$

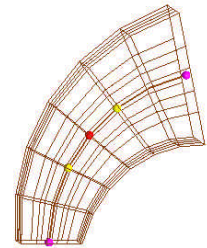
$$z(l, q) = r / c$$

for $q \geq 45$ & $q \leq 135$

$$x(l, q) = \cos(180 - b) * (l - (r + (r_2 - r_1) * l/i)) * \cos(q)$$

$$y(l, q) = \tan(180 - b) * x$$

$$z(l, q) = r * \tan(q) / c$$



Geon 24: Axis - Curved; Cross Section Edge - Straight; Cross Section Symmetry - Rotational & Reflectional; Cross Section Size - Expanded & Contracted

For $45 < q < 135$

$$x(l, q) = (r + (a_y * \cos(l)) + ((r_2 - r_1) * l/i)) * \cos(q)$$

$$y(l, q) = \tan(180 - b) * x$$

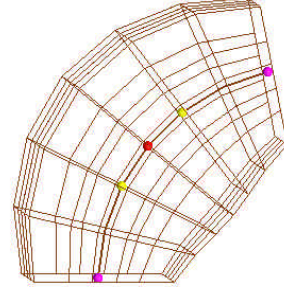
$$z(l, q) = (r + (a_y * \cos(l))) / c$$

for $q \in 45$ & $q \in 135$

$$x(l, q) = (r + (a_y * \cos(l)) + ((r_2 - r_1) * l/i)) * \tan(q)$$

$$y(l, q) = \tan(180 - b) * x$$

$$z(l, q) = (r + (a_y * \cos(l))) * \tan(q) / c$$



4.3.2) Resizing Geons

According to Biederman's theory of recognition-by-components, comparison between the relative sizes of Geons can assist in entry-level classification of objects [Biederman87]. This is the case when comparing certain classes of animals to humans. The toolkit facilitates the resizing of Geon axis length and Geon radii by allowing the user to manually enter the values for these parameters. Resizing can also be done using a scaling operation that can stretch the object in either of the x, y, or z directions.

4.3.3) Positioning Geons to Create Diagram Structures

An important feature of the toolkit is positioning Geons to create diagrams. The Geon diagram is a composition of selected Geons. It is created manually by the user moving the Geons so that they resemble a connected structure. The positioning takes place when the user selects and drags the Geon to the required position. The user can also manually enter values for the x, y, and z locations of the Geon.

4.3.4) Mapping Color, Texture, Transparency, and Labeling onto Geons

In order to validate the model described in section 4.1, the toolkit mapped color and texture onto Geons. The experiments in chapter 5 and 7 use these properties of the toolkit. Color is mapped from the range of 0 to 255 for all three components, red, green, and blue. The choice of color is not restricted and the experiments used the full range of available colors.

A limited set of bitmap textures was provided in the toolkit (Figure 4.3). The textures were created using RenderSoft IllusionaeTM¹. The textures were used to denote attributes of objects, as was color.



Figure 4.3 - Sample textures used in the toolkit.

Transparency was added to the toolkit to facilitate the visualization of containment. This property is later used in the experiments described in chapter 7. The level of transparency is modifiable by the user but was set to 30% as a default value used in the experiments.

Labeling, although not essential for the experiments conducted in the thesis, is an important feature of all diagrams. The toolkit displays labels using Bitmap Fonts, which

¹ Version 2.0 of RenderSoft Illusionae was used for creating the textures. A shareware version is available at <http://web.singnet.com.sg/~rendsoft/>.

is one of three types of fonts available in OpenGL. Bitmap Fonts were chosen, as they are efficient to render.

4.3.5) Saving and Opening Geon Diagrams

Diagrams created with Geon toolkit can be saved for future access, in particular during experiment trials. The diagrams are saved in a text file (".gnf" extension) with each line of the file defining a Geon, its position, and the values of its attributes. Also, the user can choose to open a pre-saved file. During experiment trials Geon files are automatically opened and closed. Opening Geon files restores the diagrams and its properties.

4.4 Geon Toolkit User Interface

This section describes the user interface built for assisting the researcher in fulfilling the functional requirements described above. It does not discuss all the user interface issues involved in the construction of toolkit, but focuses on describing the methods for resizing and positioning Geons in the diagram and the interface for mapping surface properties of color, texture, and labels to Geons,

4.4.1) Modifying Geon Sizes

The size of a Geon can be modified by first selecting the Geon. To select a Geon the user needs to set the display in SELECT mode. This is done by clicking on the |Select| button on the toolbar. Once the program is set in select mode, the user can click on any object and the object will be rendered in Select Style. An object in Select Mode is displayed with a wireframe mesh around its surface (see Figure 4.4). This gives the user immediate

feedback about the selection and allows the user to edit the object's properties. The object can be unselected by clicking on it again.

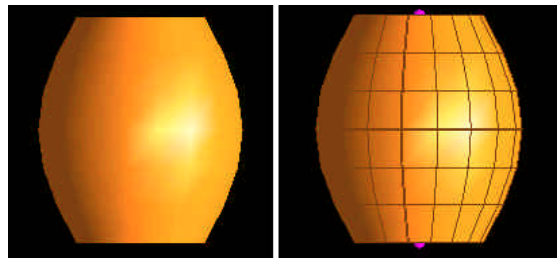


Figure 4.4 - Effect of selecting a Geon. Notice the second part has a wireframe on its surface.

After selection, Geon height, and bottom and top radii can be altered via a group of spin buttons under the Geon Properties editor. The group is labeled as group 2 in Figure 4.5 below.

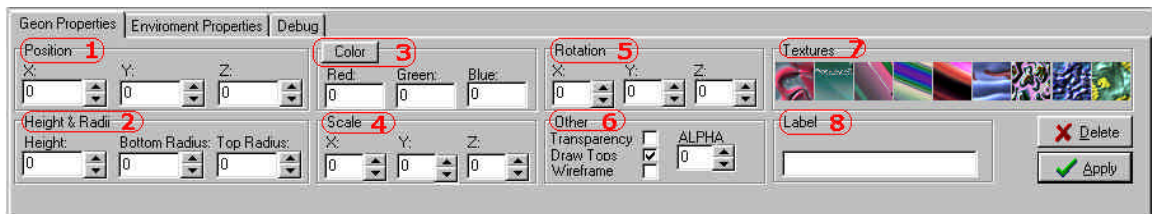


Figure 4.5 - Geon Properties editor displayed with the sections that facilitate modifying Geon attributes.

4.4.2) Positioning Geons

Positioning Geons creates the structural form and connectivity of the Geon diagram. To move an object around the scene the program needs to be set to TRANSLATE mode by clicking on the |Translate| button on the toolbar. Once this is done it is only necessary to click on the object and drag it. During the translation the object is rendered in its wireframe style to perform a smoother translation. When the wireframe of the object is drawn, its axis and points of connection are rendered as well (see Figure 4.6). Translation

can also be performed by selecting the Geon and modifying the x,y,z triplet in the Geon properties editor under the Rotation group of controls. This group is labeled as group 1 in Figure 4.5 above.

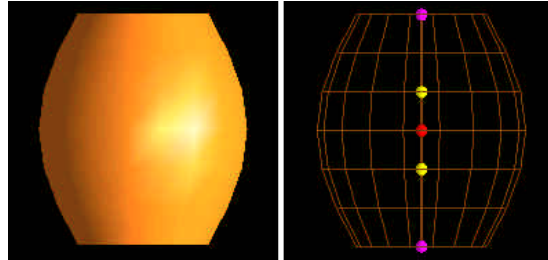


Figure 4.6 - When translating a Geon its wireframe and central axis are drawn to the speed the positioning of the object.

To rotate Geons the user sets the toolkit mode to ROTATE by clicking on the |Rotate| button on the toolbar. One end of the Geon is always fixed, around which the object can be rotated (Figure 4.7). Rotation can also be performed along the Geon's x, y, or z-axes. This is done in the Geon properties editor under the Rotation group of controls. This group is labeled as group 5 in Figure 4.5 above.

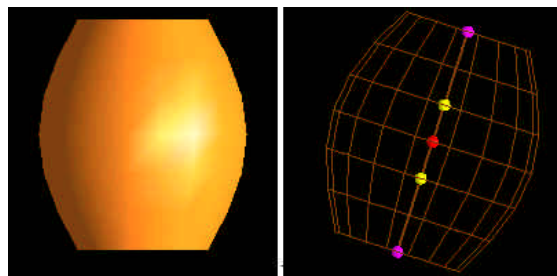


Figure 4.7 - Rotating a Geon.

To display connections in the toolkit, Geons need to be positioned sufficiently close to one another. An object can be connected to the side-of, on-top-of, or to the bottom-of another object [Biederman87]. To facilitate proper layout five connection points are implemented as seen on the axis of the Geon (see Figures 4.6 & 4.7). Biederman defines these points as being on- or off-center [Biederman87]. The theory does not explicitly define the five points as seen in these Figures, but these points are created so that reasonable differences between connections can be inferred. Snap-to-grid techniques are not implemented to facilitate the connection so these points are simply "suggestive locations" for placing connecting Geons. A better implementation would be to develop snap-to-grid techniques that would lock connecting Geons into one of the above locations. In order to provide this functionality, further research is needed to report the benefit of restricting the connectivity in node-link diagrams to these locations.

4.4.3) Mapping Surface Properties

Geon surface properties such as color, texture, transparency, and labeling can be altered using the Geon properties editor displayed in Figure 4.5 above. In order to modify any of these properties the Geon has to be selected.

Each Geon's RGB component can be modified using either the standard windows color palette or by using the controls under group 3 in Figure 4.5. The color is applied immediately to the Geon once the color on the palette is selected or when the spin buttons are incremented or decremented.

Texture can be mapped onto a Geon using the controls in group 7 in Figure 4.5. The user first selects a Geon and then clicks on the texture to be mapped onto the Geon.

Transparency can be turned on or off using the controls under group 6 in Figure 4.5. The user can increase or decrease the transparency level by modifying the Alpha value.

A label can be attached to a Geon using the controls in group 8 in Figure 4.5. The user enters a text label and then applies the changes to the Geon.

4.5 Discussion

The Geon diagramming model defined in this chapter is motivated by theories developed recently in object perception research. The Geon diagram is a form of node-link diagram that is characterized by Geon elements as nodes and links, and mapping of data attributes to surface attributes of Geons. As such, the Geon diagram is intrinsically a structural representation of entities and relationships between them.

To validate the Geon diagramming model, a toolkit was developed to facilitate the creation of Geon diagrams. It provides for drawing Geons in a principled way. The toolkit creates 3D volumetric solids using parametric descriptions derived from the properties of Geons as described by Biederman. The toolkit also supports the mapping of surface properties such as color, texture, and transparency onto Geons.

In order to pursue further research using the toolkit, several enhancements could be implemented. Drag-and-drop, snap-to-grid and constrained deformation techniques will speed the creation of diagrams. Controls can be added to adjust lighting conditions and to display drop shadows. The Geon Structural Description (GSD) is an important element in the theory of recognition-by-components, and maintaining a GSD during the creation of diagrams in the toolkit can extend the investigation of mapping data elements onto GSD like structures.

The toolkit was created with the primary intention of serving as a research tool. Diagrams with various properties were created and saved in order to perform the experiments that were developed to validate the Geon diagramming model. These experiments and their results are described and discussed in the chapters that follow.

Chapter 5 - Comparison of Geon and UML Diagrams for Sub-structure Identification and Recall¹

This chapter continues with a review of the perceptual theory underlying the Geon Diagram concept and reports three experiments that evaluate the effectiveness of Geon diagrams.

A critical claim for the Geon diagram is that it will allow the visual system to carry out a kind of automatic perceptual parsing of an object's structural skeleton. If the data structures are mapped to this structure then the diagram should be easier to interpret (parse). We therefore begin with a more in-depth discussion of the research literature

¹ A preliminary report of the work in this chapter has been provided in "Irani, P. & Ware, C., *3D Diagrams Based on Theories of Structural Perception*, Proceedings in Advanced Visual Interfaces, Palermo, Italy, May, pp. 61-67, 2000" and in "Irani, P. & Ware, C., *Diagramming Information Structures using 3D Perceptual Primitives*, ACM Transactions on Computer Human-Interaction (in review)."

that relates to the parsing of visual structure before presenting the first experiment that is based on a sub-structure detection task.

5.1 Parsing Visual Structures

Visual parsing for the extraction of primitives from an object occurs at a higher level than feature extraction. Biederman et al [Biederman87] have shown that parsing of 2 - 3 Geons is sufficient for a fast and accurate recognition of an object. In one of their experiments, subjects were asked to identify familiar objects after being exposed to parts of a scene. They were capable of naming the objects after detecting 2 or 3 Geons. However, complex objects (6 or more Geons) took longer to recognize than simple objects. Such complex objects presented with less than half their components were accurately named in 75% of the trials, further supporting the idea that only a few Geons are necessary to recover an object. On the other hand, they found that distinguishing single-Geon objects that vary only by color or texture, (plums vs. peaches) took longer to recognize than more complex objects supporting the idea that color and texture are secondary attributes.

The goal of another experiment by Biederman et al [Biederman87] in relation to visual parsing of elements and structure was to determine whether the addition of a fourth but inappropriate Geon to the 3-Geon partial object that was already recognized would reduce recognition speed. For example, if a stool could be recognized by its circular base and two legs (3-components altogether), then adding a fourth Geon such as a barrel on

top (Figure 5.1) will not destroy our ability to perceive the stool. Their results showed that if three Geons were sufficient to activate recognition of the object's representation, then unless the added Geon activates a competing object, no interference occurs and the object is recognized rapidly and accurately.

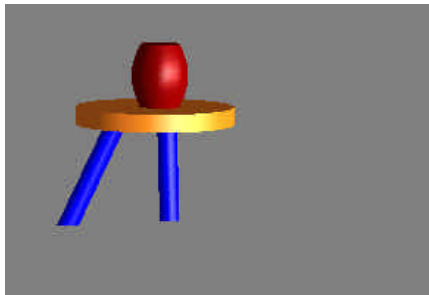


Figure 5.1 - Adding an additional Geon (barrel on top) does not interfere with recognition of the initial object with three Geons (stool). This suggests that when the number of Geons is sufficient to recognize an object, an inappropriate Geon does not affect recognition.

At an initial stage, the visual parsing needs to properly identify and segment the object at regions of concavity. Biederman et al [Biederman87] conducted an experiment in which the images were presented without showing the contours at critical regions in the image. The subjects were unable to identify the objects even after long periods of viewing (5 seconds). The same number of contour removals was made, but not to those regions of sharp concavities. In these scenes, subjects identified the objects with more ease but still at a slower speed than the intact objects. These results suggest that sections of sharp concavities are essential in identifying parts out of the whole.

5.2 Role of Structural Description for Recognition

For recognition to take place, it is important that the entire structural description is identified. In an experiment comparing the perception of degraded objects to that of partial objects, Biederman et al [BG93] presented subjects with objects having missing parts (partial objects) and objects with their contours degraded. They recorded the accuracy of naming objects under these two conditions. It was found that at longer exposure times (around 200 ms) performance with objects having degraded contours was better than with partial objects. Their explanation to this was that once the primitives were recognized from the degraded contours the entire structure was perceivable and recognition would be more accurate. Whereas with the partial objects, certain primitives that were missing could not help in constructing the structure of the object.

To test the theory that we build and store structural models, several studies have investigated the internal representation of structural information of possible and impossible objects. Impossible objects (Escher-like scenes) are those that have certain structural violations prohibiting them from existing in the real world (Figure 5.2). The purpose of using such stimuli is that they could help determine whether structure is stored in our visual memory system, since impossible objects cannot have an internal representation.

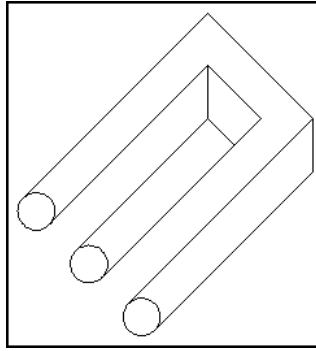


Figure 5.2 - The object appears to be a 3-D object, but violates certain physical properties depending on how it is viewed. By covering the left side, the object is viewed consisting of two prongs. Covering the right side makes it appear as an object with *three* prongs. By viewing the entire image at once, the object switches between having two and three prongs. This impossibility confuses the mind.

A study by Schacter et al [SC93] made use of an object decision task, in which a line drawing of a scene must be identified as to whether it depicts a possible 3D object. In their task subjects were asked to distinguish between representations of possible and impossible structures. The task was designed such that performance would require analysis of information about global, three-dimensional object structure. Their results showed that enhanced performance was observed after encoding of global three-dimensional structure but not local or two-dimensional features, suggesting that visual memory for unfamiliar objects depends on access to structural descriptions.

Williams [Williams95] extended the study by Schacter et al [SC93] to determine the effects of picture-plane orientation and size transformations on object recognition. In a yes/no recognition task subjects were asked to indicate whether a stimulus had been presented to them previously. The results showed that changing image size did not affect recognition at all, while rotating objects 60° or more in the picture-plane degraded

performance. These results indicate that appropriate orientation in the picture-plane can assist in recognizing objects. This is in line with Biederman's idea of verticality in which relative orientations between component parts plays a significant role in recognition of the structure.

Although internal representations of structural descriptions suffer from picture-plane rotations, recognition of objects rotated in depth are assisted by the structure and its viewpoint invariant properties. If the viewpoint provides the same structural information and primitives, then we are capable of identifying it. Bar et al [BI96] performed a study in which they compared the recognition of objects under changes of their metric properties (MP) or their viewpoint invariant properties (VIP). Metric properties are such things as lengthening or shortening of a Geon, increasing the degree of curvature, etc. A viewpoint invariant property indicates whether a Geon is curved or straight, for example. They found that presenting objects at different orientations in depth had no effect on detecting differences when the objects differed in a viewpoint invariant property, but reduced detectability of metric property differences significantly. The metric property differences that were detected showed a statistically significant increase in reaction times. These suggest that our system employs viewpoint invariant properties to achieve invariance in recognition of objects rotated in depth. According to Biederman's theory, these properties are essential in extracting the correct shape primitives from the image.

In visually parsing a diagram, the types of connections, the set of interconnections between nodes, and the attributes of nodes and links are all important for creating a

mental model of the abstract system that leads to a proper interpretation. This process is similar to that in which our visual system extracts information when recognizing objects around us, as suggested by Biederman's Geon theory. Therefore we should be capable of visually parsing a diagram made up with Geons better than one using a line and box notation. This claim is central to this thesis and has lead to the investigations described in the experiments that follow.

5.3 Empirical Studies

The remainder of this chapter describes three different experiments designed to test the value of using Geon primitives in making diagrams of structured data. To measure the ease of interpretation, a task requiring the identification of sub-components of a larger diagram was devised (Experiment 1). This requires a kind of visual parsing that is an analytical skill. Both, speed and accuracy for this task were measured. To measure how easy Geon diagrams are to remember, subject's ability to recall briefly presented diagrams was evaluated (Experiments 2 and 3).

In all three experiments, the Geon based diagrams were compared to diagrams made using UML class diagram notations. UML was chosen because it is a rich diagramming notation that combines several diagramming techniques and is a *de facto* standard for many diagrams used in Software Engineering. From the wide range of UML diagrams, the Geon objects were compared to notations of UML class diagrams as it has the greatest variety in its notation set.

5.4 Experiment 1: Sub-Structured Identification with Geon Versus UML Diagrams

The purpose of the first experiment was to determine the ease with which people can visually parse Geon diagrams in comparison with equivalent UML diagrams. A mapping convention of Geon elements to UML objects was devised (Figure 5.3). This mapping uses compact Geons as nodes in the diagrams and elongated Geons as links between nodes. In this type of mapping, the primary concern is the types of links and nodes in both conventions. For example, a bend in a link that represents a composition in the Geon diagram is equivalent to one that does not have a bend but has a closed diamond at one end of the edge in the UML diagram (see Figure 5.3, Composition). A sample mapping of a Geon sub-structure to a UML sub-structure is shown in Figure 5.4. The variables that were measured in this experiment were the response time and the error rate for a subject to recognize a sub-structure of a diagram.

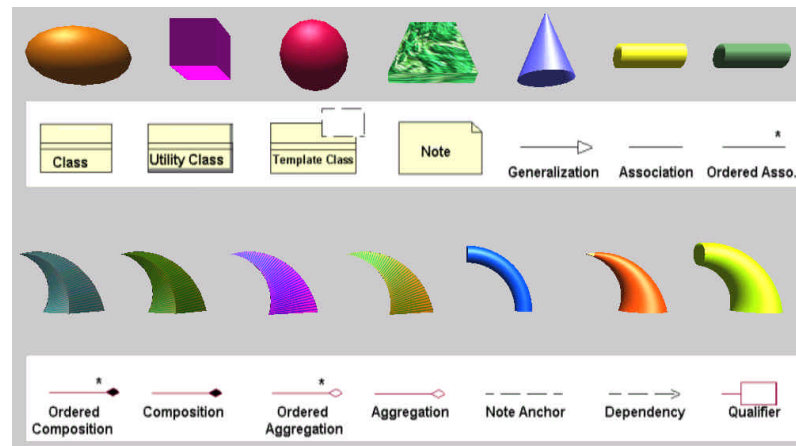


Figure 5.3 - Mappings between Geon and UML components for experiments 1 and 2. Colors on Geon components are used to map to different object names in UML. For example, an orange ellipsoid maps to Class A and a red ellipsoid maps to Class B. Also in the case of Association/Ordered Association, Composition/Ordered Composition, and Aggregation/Ordered Aggregation the only differentiating factor was the texture of the Geon components.

5.4.1 Task

The sub-structure identification task consists of finding sub-structure within a larger structure and therefore requires that the entire structure be visually parsed. In order to perform the task successfully, theory suggests that subjects will store the target sub-structure with a set of structural description rules in visual working memory. For example, for the target sub-structure of Figure 5.4, subjects could have retained information such as: the target has 2 nodes, one ellipsoid and one sphere. The ellipsoid is connected to the sphere using a blue-curved-pyramid. For the same task to be performed with UML diagrams, a similar method would be required but the same level of automatic processing cannot be expected.

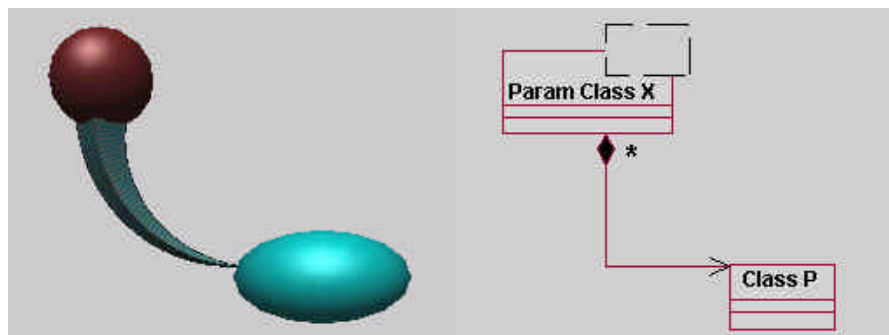


Figure 5.4 - Geon and UML sub-structures with 2 nodes or 3 components.

5.4.2 Hypothesis

Identification of sub-structures is faster and more accurate with Geon diagrams than with UML diagrams.

5.4.3 Experiment Control Software

Experiment control software was developed to randomize the trials, present the images and record the subjects' input times.

5.4.4 Equipment

For this experiment all the diagrams were presented on a 17-inch computer monitor. The experiment was run on a Pentium 433 MHz Silicon Graphics machine in a Windows NT environment.

5.4.5 Method for Experiment 1

In this experiment, subjects were required to determine whether or not a previously seen diagram sub-structure was part of a diagram. Subjects were informed that time was being recorded but were also told not to hurry.

Diagrams

Two sets of ten UML diagrams were drawn using Rational RoseTM, and two equivalent sets of Geon diagrams were constructed using the Geon toolkit. The UML diagrams were artificial in the sense that they did not depict any particular system but included most of the boxes and arcs used in UML class diagrams. The UML diagrams also used text to denote the names of classes and the type of the class (i.e. Parameterized Class A, or Utility Class B, etc.). The Geon diagrams made use of color and texture instead of text labels to distinguish between the different labels used on the UML nodes. For example, if the UML diagram contained Class A and Class B as two separate entities, our mappings produced an equivalent Geon diagram with a red ellipsoid for Class A and a blue ellipsoid for Class B.

Figure 5.6 gives an example of a Geon diagram and its equivalent UML structure. For each set a sub-structure was constructed (target); for the first sets the sub-structure contained 2 nodes (Figure 5.4, above) and for the second sets it contained 4 nodes (Figure 5.5). The sub-structure only was present in half the diagrams. A diagram was defined as containing the target if the sub-structure's components and their connections were present. However, the sub-structure layout could be different, meaning that sub-structure recognition could not be performed by a simple template match or a simple picture-plane rotation. Subjects were also told that the metric properties of some components of the sub-structures could differ; such as the length of a Geon link, or the bend in a UML edge, but the presence of such changes would not result in an exclusion of the sub-structure.

In order to perform this task successfully, subjects would likely store the target sub-structure with a set of structural description rules in visual working memory. For the target sub-structure of Figure 5.5, subjects could have retained information such as: the target has 4 nodes, one ellipsoid, 2 spheres, and a cube. The ellipsoid is connected to one sphere with a blue-curved-pyramid and to the other with an orange-curved-cone. It is important to build the mental model based on this structural information as the local features of color and texture alone could mislead the parsing process.

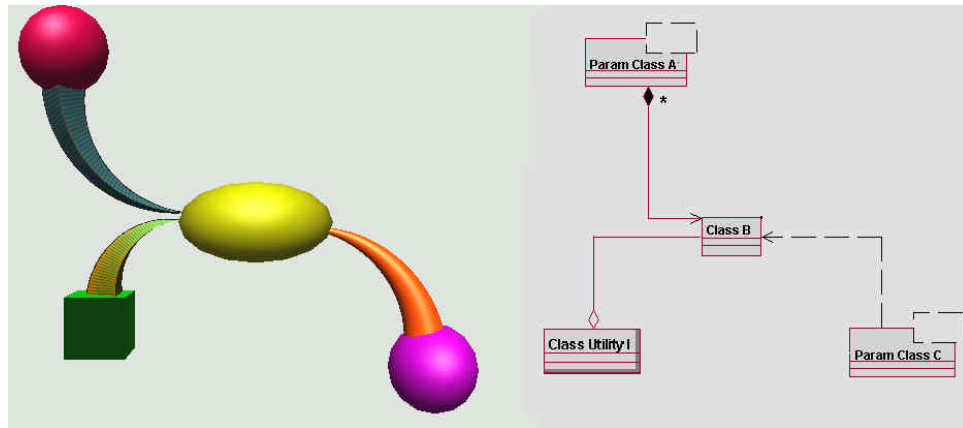


Figure 5.5 - Geon and UML sub-structures with 4 nodes or 7 components.

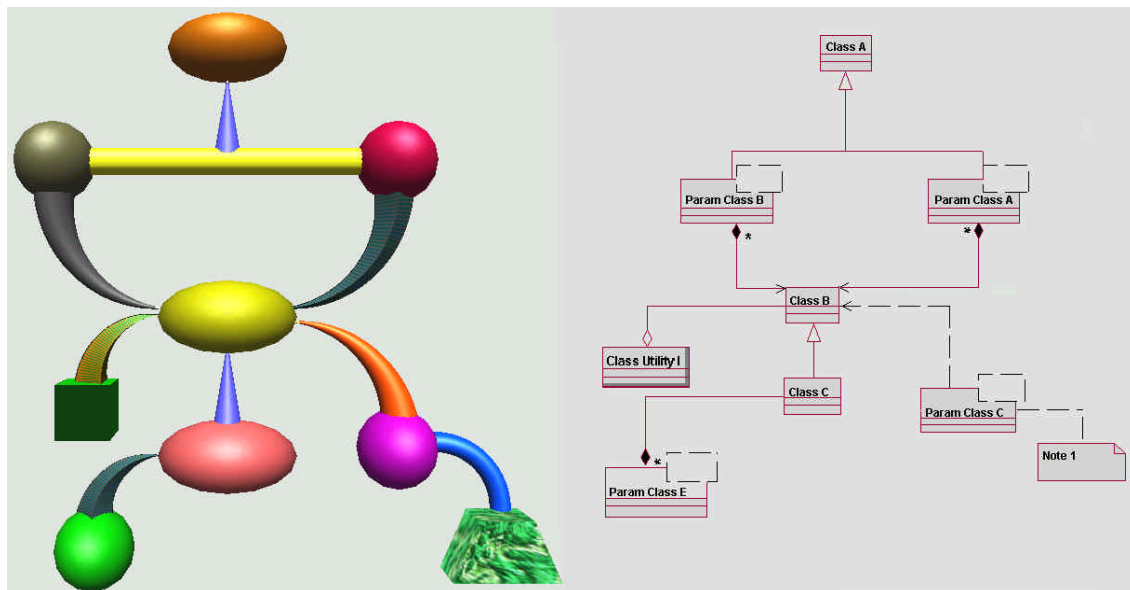


Figure 5.6 - An example of a Geon diagram and its UML equivalent in which the substructures of Figure 5.5 are present.

Procedure

On each trial the subject was first shown the sub-structure for 15 seconds and was then given 3 practice trials on which they were shown the presence of the sub-structure. A time interval of 15 seconds has been used in studies investigating recall of briefly

glimpsed pictures in which subjects were asked to remember details of the pictures [Coltheart99], and would thus suffice to store all the primitives and their interconnections. After the subjects completed the practice trials, the program presented a random sequence of 10 diagrams. For each diagram the participant pressed the 'Y' key if the sub-structure was present or the 'N' key if it was not. The response time of the user was captured along with the correctness of the response.

The order in which the sets were presented to subjects was randomly selected from one of the following sequences {G1,U1,U2,G2}, {U1,G1,G2,U2}, {U1,G1,U2,G2}, {G1,U1,G2,U2}. Where G denotes a set of Geon diagrams and U a set of UML diagrams.

Subjects

The 15 subjects were all computer science students who had previously been exposed to some form of entity-relationship diagram with nodes and links.

5.4.6 Results of Experiment 1

The results are summarized in Table 5.1. These show that substructures were identified both faster and more accurately with the Geon diagrams. From the 15 subjects, 11 subjects correctly identified the sub-structure in more Geon than UML diagrams, 1 subject identified the sub-structure equally often with the Geon diagrams as with the UML diagrams, and the remaining 3 were more accurate with the UML diagrams. A sign test shows this difference to be significant ($p < 0.05$).

Because the data were highly skewed with a few outliers they were log transformed for analysis. The means were inverse transformed back to seconds for clarity of presentation. An analysis of variance model revealed that diagram type (Geon or UML) was a significant factor in the time it took to correctly or incorrectly identify the sub-structure. $F(1,14) = 41, p < 0.001$. On average the subjects took 4.3 seconds to identify (correctly or incorrectly) the presence of the sub-structure in the Geon diagram and 7.1 seconds for the UML diagrams. Of the set of 15 subjects, 13 identified the Geon sub-structure faster than the UML sub-structure.

	Geon Diagram	UML Diagram
Identification Time (sec)	4.3 (± 0.4)	7.1 (± 0.5)
Error Rate	13.33%	26.33%

Table 5.1 - Summary of Results of Experiment 1.

There was a small but significant effect of the number of components in the sub-structures. $F(1,14) = 10.5, p < 0.005$. The results for the 3 component and 7 component sub-structures are summarized in Table 5.2. Subjects took longer to identify the presence or absence of a substructure when it was comprised of more components. These results therefore suggest that Geon components facilitate visual parsing for complex and simple sub-structures.

	Geon Diagram	UML Diagram
3-Component (sec)	3.4	6.5
7-Component (sec)	5.1	7.7

Table 5.2 - Reaction Times for parsing diagrams with 3 and 7 components.

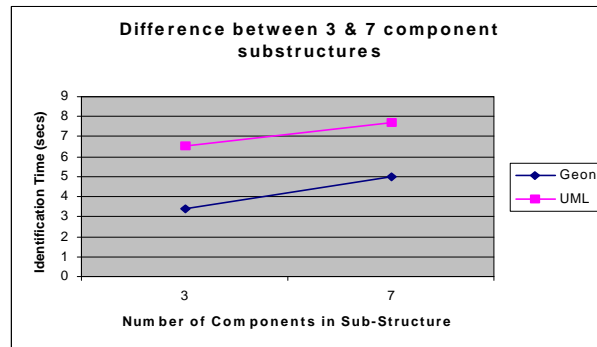


Chart 5.1 - The time required for parsing a sub-structure increases with the number of components.

5.4.7 Discussion

A significant difference in performance between both types of diagrams has been observed, supporting the hypothesis that identification of sub-structures is faster and more accurate with Geon diagrams than with UML diagrams. Overall, subjects were 39% faster and twice as accurate with the Geon diagram than with the UML diagram. The performance difference for recognizing the sub-structure with fewer primitives is also consistent with the results of Biederman et al [Biederman87] which suggest that recognition speeds are slower when it involves identifying complex objects containing more Geons.

The primary variation between the Geon and UML diagrams was in the nature of the shapes and size of primitives and connections between components. The most plausible explanation for improved performance with Geon diagrams is that Geon shapes could have facilitated memory of the connections, since they have more natural characteristics

in comparison to UML components. The connections and appearance of Geons are also more salient than those of the box and line shaped components in UML.

Explanations other than the one proposed here are possible. It might not have been the use of Geons that resulted in the improved performance. Some other factors could have been critical. There are many differences between both types of diagrams at the feature level (surface attributes) as well as the object level (form). For example, text labels on UML diagrams were mapped to color and texture on the Geon primitives. Another variation at the feature level was the length or size of the connections, or that square bends in UML links would have been mapped to curved Geons. These variations could have caused the difference in performance. A more carefully controlled study to assess the importance of 3D structures is the theme of chapter 6.

5.5 Experiment 2: Recall of Geon Versus UML Diagrams

The purpose of the second experiment was to determine whether Geon diagrams can be remembered more easily than UML diagrams. Evidence suggests that the unit of capacity of visual memory is at the object level rather than at the feature level [LV97]. Luck et al found that when subjects were presented with several objects containing combinations of features such as color, size and orientation, that these subjects were capable of retaining the collection of all features by organizing these into objects. These findings suggest that our capacity for storing visual information is greatly enhanced by grouping features (including shape, orientation and surface properties) into objects. Such a grouping can be

achieved using single Geon objects, which have distinctive shapes, orientations and surface attributes.

Studies have reported that visual memory also stores relational information between individual objects [JOC00]. Jiang et al [JOC00] further propose that relational information of individual objects is stored based on global spatial configurations, suggesting that retaining the relative locations and relationships between elements in the diagram are important for appropriately recalling the image.

5.5.1 Hypothesis

Since Geon diagrams resemble real-world structures more than comparable UML diagrams, they will be more readily recalled after a brief exposure.

5.5.2 Method for Experiment 2

This experiment was formulated to compare the accuracy of recalling Geon diagrams versus equivalent UML diagrams.

Diagrams

A set of 14 UML diagrams was developed using Rational RoseTM. Using the mapping convention of Geons to UML entities discussed in 5.4, an equivalent set of 14 Geon diagrams was produced using the Geon toolkit. A sample is shown in Figure 5.7. Both sets of diagrams were printed in color on 8.5 by 11" transparencies and projected on an 6x6 ft overhead screen.

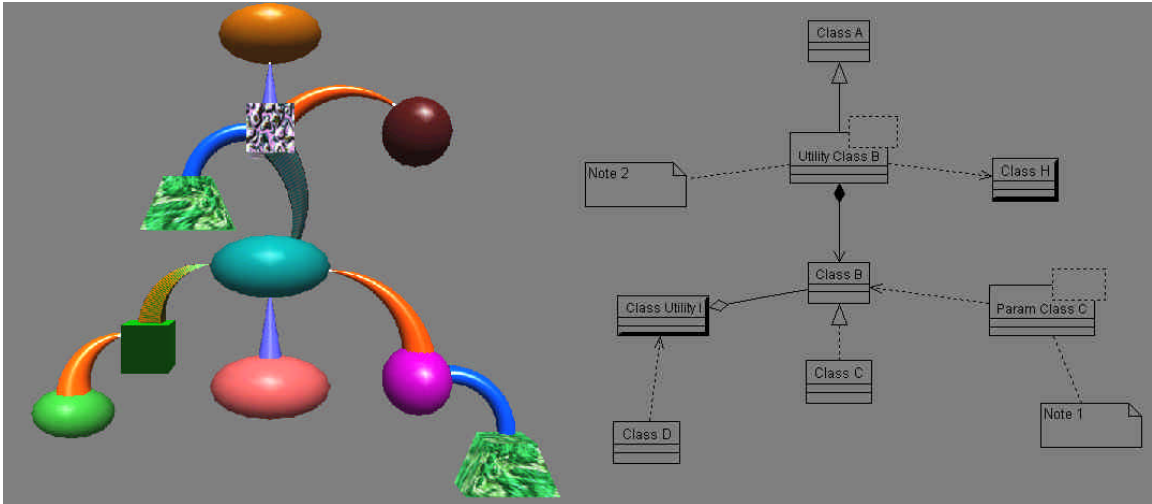


Figure 5.7 - A Geon and its equivalent UML diagram used in Experiment 2.

Procedure

The experiment was conducted in a university class on software engineering. At the beginning of the lecture the first set of students were shown half (seven) of the set of Geon diagrams in random order for 15 seconds per diagram. After being presented with the first half of Geon diagrams, they were then presented with the seven corresponding UML diagrams, randomly, at intervals of 15 seconds. At the end of the lecture, approximately fifty minutes later, the students were shown the full set of 14 Geon diagrams and 14 UML diagrams. Each diagram was shown for 10 seconds and the subject indicated on a printed sheet whether that diagram had been part of the initial set. To counterbalance the results, the same procedure was applied to the second set of students with the UML diagrams being presented first and the Geon diagrams second.

There were 18 students who participated in the trial with Geon diagrams presented first, and 17 students that participated in the trial with the UML diagrams first giving a total of 35 subjects.

Subjects were typically at a distance of 10 to 30 feet from the screen.

Subjects

The experiment was conducted with 35 students in senior level computer science courses.

If a student had previously performed a similar experiment they were asked to indicate this on the results sheet handout and their data was later discarded. All the students were familiar with UML notation.

5.5.3 Results of Experiment 2

The results are summarized in Table 5.3. Subjects made less than half the errors in recalling Geon diagrams than they did for the equivalent UML diagrams; 18% error rate for the Geon diagrams vs. 39% for the UML diagrams. This difference is especially striking considering that chance performance is 50%. From the 35 subjects, 26 correctly recalled more Geon than UML diagrams while 5 correctly recalled the same number of Geon diagrams as UML diagrams and 4 subjects correctly recalled more UML diagrams. A sign test showed this difference to be highly significant ($p < 0.005$).

	Geon Diagram	UML Diagram
Error Rate	18%	39%

Table 5.3 - Summary of Results of Experiment 2.

5.5.4 Discussion

The results support the hypothesis that Geon diagrams are easier to remember than 2D box and line diagrams such as those found using UML. As stated earlier, one explanation for this performance could be due to the strong structural impact Geon diagrams have on the visual system. If a subject were to store the diagram during the viewing, that representation could be formulated as a description such as: a sphere on top, connected to an ellipsoid via a curved-cone, etc. On the other hand, the structural information stored for the UML diagrams would necessitate a more complex description in memory consisting of something such as: a box with double lines, connected to a plain box with a fold in one corner via a dashed line, etc. In the case of the UML diagrams the subject would also need to store the labels rather than colors or textures, which can contribute to an overall deficiency in recalling the image.

5.6 Experiment 3: Recall of Geon Versus UML Diagrams Without Surface Attributes

The Geon diagrams used in Experiments 1 and 2 used Geons that were distinguishable using surface attributes such as color and texture, as well as 3D Geon shape. It might therefore be the case that color and texture were more important than the use of 3D Geons in making these diagrams more effective. Observers might have memorized elements in the diagrams based on their surface properties such as color. For example, they might have encoded a given diagram as a "blue colored object connected to a red one".

The third experiment was designed to determine whether recall of Geon diagrams without surface attributes would still be better than the recall of UML diagrams. The one-to-one mapping between Geon elements and UML objects is shown in Figure 5.8.

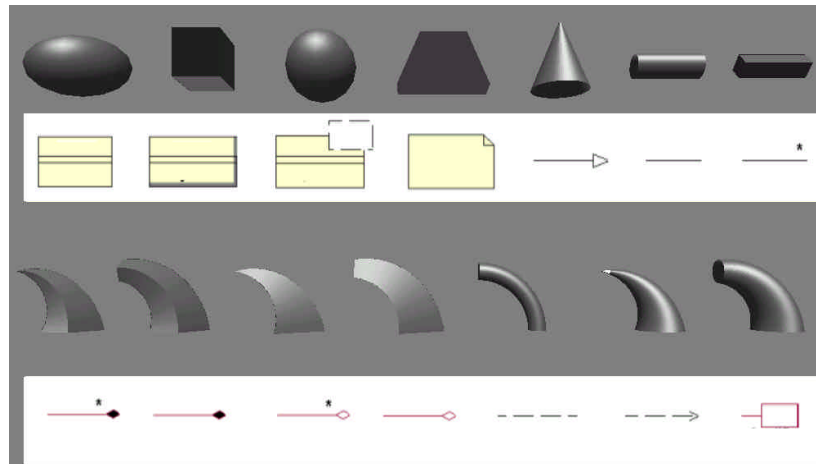


Figure 5.8 - Mapping used for Experiment 3. The Geons did not have color or texture attributes and the labels from the UML components were removed.

5.6.1 Hypothesis

Geon diagrams without surface attributes of color and texture would be more readily recalled after a brief exposure than comparable UML diagrams without labels.

5.6.2 Method for Experiment 3

The procedure was the same as for Experiment 2. The first group of students was shown the UML diagrams first and the second group of students was shown the Geon diagrams first. After the experiment, the data from those students who had performed a similar experiment before were discarded as well as three randomly chosen data sets giving a total of 18 students in the first group and 17 students in the second. This brought the total number of subjects to 35 as in Experiment 2. A sample Geon diagram and UML equivalent used in this experiment is shown in Figure 5.9. The diagrams were printed in

black and white on 8.5x11" transparencies and projected on an overhead screen 10 to 30 feet away from the subjects.

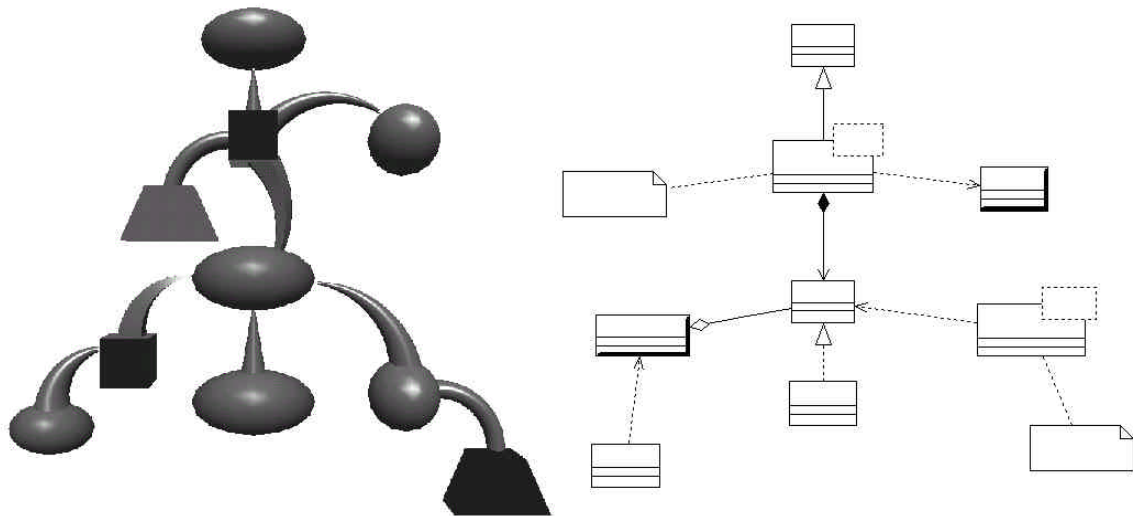


Figure 5.9 - A Geon and its equivalent UML diagram used in Experiment 3.

5.6.3 Results of Experiment 3

The results are summarized in Table 5.4. Subjects on average had an error rate of 22.5% for recalling the Geon diagrams and 42% for recalling the equivalent 2D UML diagrams. From a total of 35 subjects, 25 recalled more Geon diagrams, 2 recalled the same number of Geon diagrams as UML diagrams, and 8 subjects recalled more UML diagrams. A sign test shows this difference to be significant ($p < 0.01$).

	Geon Diagram	UML Diagram
Error Rate	22.5%	42%

Table 5.4 - Summary of Results of Experiment 3.

5.6.4 Discussion of Experiment 3

The results strongly support the hypothesis that remembering Geon diagrams is easier than remembering UML diagrams even when the Geon diagrams are not presented with

surface attributes. Although a direct comparison between these results and those of Experiments 2 and 3 cannot be made statistically because of some differences between the protocols, they strongly suggest that surface properties may not play a significant role in recalling structural information from the diagrams. This result further supports the idea that spatial configurations of visual objects is more important for visual memory than is the configuration of object features such as texture or color [JOC00].

5.7 General Discussion

The practical implications of these experiments are that we can dramatically improve how easily sub-structures are identified and remembered in diagrams by using the Geon diagram principles. The results of these experiments support the theories of object perception that suggest that matching structural descriptions of objects plays an important role in the object recognition process. The results of Experiment 1 show that visual parsing of diagrams is facilitated with the use of Geon primitives. Experiments 2 and 3 were developed to investigate whether Geon structures are easier to recall than 2D UML structures. The results showed that with and without surface attributes, diagrams made with Geon primitives are easier to recall than the traditional box and line drawings, with half as many errors.

The results also support the recent findings in neuro-physiology research suggesting the possibility of a separate memory system for structure [CS84,Kosslyn94]. According to these theories, information about an object's component parts or global structure and

information about its functioning and characteristics might be represented by separate systems.

One drawback with the mappings used in these experiments was that layout of the diagrams' structures were not exactly equivalent. For example, links in Geon diagrams were mostly made up of curved features whereas those in the UML diagrams were presented with abrupt changes in the line directions. As a result, one can argue that the overall semblance of the diagrams was not equivalent, which could have contributed to the performance results. In addition to layout there are a great many differences between UML and Geon diagrams that make it impossible to say which aspects of the diagrams have the strongest contribution. It might have been the different types or connectors, specific shapes might have been hard to remember, or the specific use of colors and textures. However, Experiment 3 does eliminate the role of color and texture as key factors. The specific issue of whether 3D shapes are more effective than 2D shapes are addressed in chapter 6.

Chapter 6 - Importance of Shading¹

Although the first three experiments strongly support the idea that using 3D primitives in diagrams can make them easier to interpret and remember than UML diagrams they do not allow us to make the generalization that it was the 3D aspect of Geons that made the difference. There were many other differences between the 3D Geon-based diagrams and 2D UML diagrams, in particular the outline shapes of the elements used. Thus the results may be due to other factors than the use of shaded 3D primitives.

The experiments in this chapter specifically isolate the importance of 3D shape-from-shading in determining the effectiveness of Geon primitives. For clarity, Figure 6.1

¹ A preliminary account of the work in this chapter has been reported in "Irani, P. & Ware, C., *Should the Elements of Diagrams Be Rendered in 3D?*, IEEE Symposium on Information Visualization, CD Proceedings, Salt Lake City, Utah, November 2000" and in "Irani, P. & Ware, C., *Diagramming Information Structures using 3D Perceptual Primitives*, ACM Transactions on Computer Human-Interaction (in review)."

shows the difference between 3D shaded objects, 2D outline elements, and 2D outlined-silhouette components. The 3D shaded objects are Geons. The 2D outline elements are more similar to box and line elements that are used in standard diagramming conventions such as UML. However they lack critical color and contour information. The 2D outlined-silhouette components have been given key internal contours and match the overall color of the Geon elements.

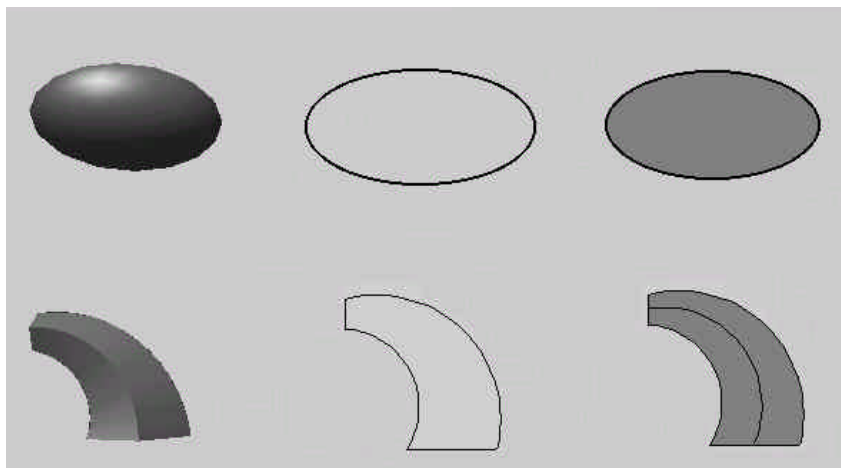


Figure 6.1 - Difference between 3D shaded objects, 2D outline shapes, and 2D outlined-silhouette components.

The two experiments described in this chapter were designed to be similar to Experiments 1 and 2 in which subjects were required to perform a visual analysis of the structural elements in a diagram. Like Experiment 1, Experiment 4 measured the amount of time it takes a subject to identify a sub-structure in a diagram. It also measured the accuracy with which the user identified the sub-structure. Like Experiment 2, Experiment 5 measured the accuracy of recalling Geon diagrams in comparison with their 2D

outlined-silhouette equivalents. Before discussing these experiments, a brief review is presented of the evidence that shading and 3D shapes are primitives of object perception.

6.1 Shape-From-Shading

A study by Enns and Rensink [ER90] investigated the influence of scene-based properties such as direction of lighting, surface locations and orientations, and surface reflectance, on visual search. Their targets were composed of colored polygons with white, gray, and black pixels (see Figure 6.2) some of which could be interpreted as three-dimensional objects. The task consisted of locating single target items among 1, 6 and 12 objects. They found that observers were significantly slower in finding the target when the items were two-dimensional. They concluded that rapid search is possible when the items consist of spatial and intensity relations that can be interpreted as three-dimensional objects.

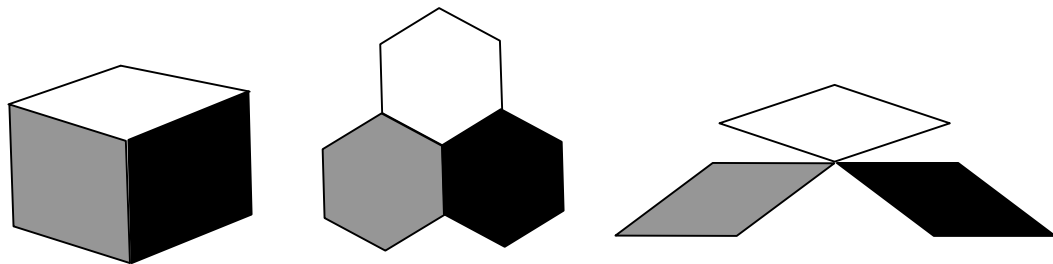


Figure 6.2 - Targets used in Enns and Rensink's experiments. The target to the left corresponded to projections of simple blocks under various lighting conditions. The pattern on the left was perceived faster than the 2D patterns on the right. Adapted from [ER90],

Sun and Perona [SP96] extended the work by Enns and Rensink [ER90] in investigating the preattentive perception of elementary three-dimensional shapes. To determine whether shading was more important than internal line crossings (these can contribute to

determining the shape of a three-dimensional object) they compared the speed of processing single target patterns consisting of 3D shaded top-lit polyhedrals to their counterpart unshaded line drawings (see Figure 6.3). Their results suggest that the shaded objects were processed faster and in parallel while the line drawings of the 3D shaded objects were processed serially. Their results are in line with those of Braun [Braun93] which showed that smoothly shaded circular targets, without any internal line edges, and which resemble 3D shaded bubbles, are processed in parallel and preattentively based on the perception of their 3D shape.

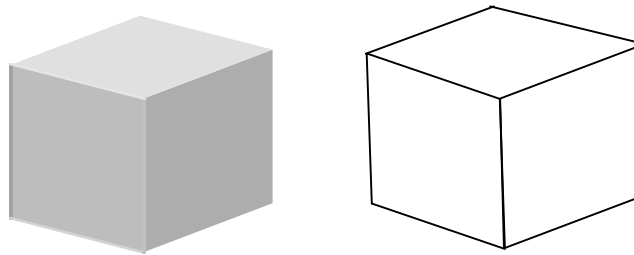


Figure 6.3 - Sample targets used in the experiment by Sun and Perona. A feature in common between the shaded item and the line drawing is the embedded Y-junction that assists in determining the shape of the object. Adapted from [SP96].

Although shading information can be processed preattentively, the human visual system makes certain assumptions about the direction of the light source. These effects, first published by Ramachandran [Ramachandran88], are illustrated in Figure 6.4 below. In this figure, the shaded discs in the upper right and lower left appear as convex items. Their shadowed tops are the result of a perceptual bias that light is shining from above. The disc in the lower right appears concave, as the shadow resulting in its lower portion results from the imaginary-protruding surface. The effect is amplified when a top-shadowed disc is superimposed on a bottom-shadowed disc, giving the effect of a crater on a bump.

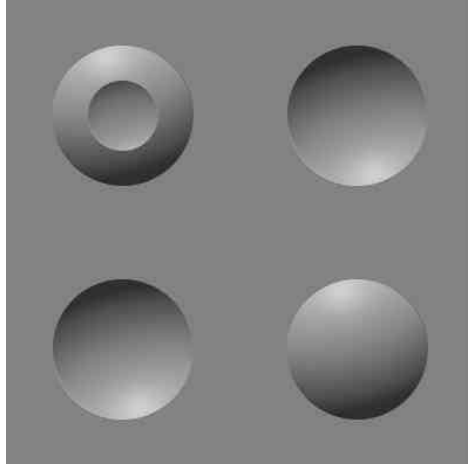


Figure 6.4 - Our visual system makes assumptions about the directions of light source. Adapted from [Ramachandran88].

The evidence of this bias is further supported by results from a recent physiological study by Hanazawa and Komatsu [HK01]. They suggest that neurons in a visual area of the brain (visual area V4) are adapted to the inclination where the light source is positioned somewhere above the viewer and the object or scene being viewed.

6.1.1 Structure-From-Shading

Shading information can help reveal structures better in visual displays. An example of a software visualization tool using shading information to reveal structure is seen in the cushionmap system [WW99]. Treemaps (see section 2.3 for a discussion of treemaps) have several limitations [Eick97] one of which is that they fall short of visualizing the actual tree structures. With cushionmaps, Wijk et al. applied a shading model to reveal hierarchical structures in treemaps. The human visual system is trained to interpret variations of shades as illuminated surfaces [FAP90]. By applying a shading model the

substructures in the hierarchy can be clearly revealed. Wijk et al. applied a specular shading model to the surface of a treemap to produce concave nodes resembling cushions (Figure 6.5). This visualization technique applies shading information to also better reveal the nesting of substructures.

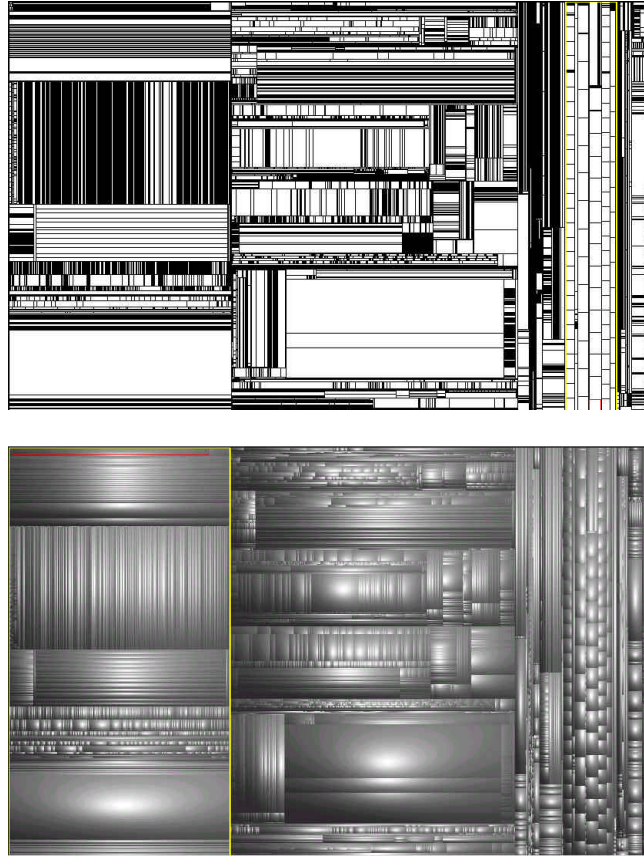


Figure 6.5 - Treemap and cushionmap representations of my hard drive. Folders that are not visible in the Treemap (above left side) are clearly distinguishable in the cushionmap version. The nesting of substructures is distinguishable better with the cushionmap.

6.1.2 Shading models and parameters used for the experiments

The realism of a graphic image depends to a large extent on the simulation of shading or lighting effects. A shading model is used to compute the intensities and colors to display

for a surface; it dictates how light is spread over or reflected from a surface. Models of light used in data visualization are based on ambient light, diffuse scattering and specular reflections. In the Geon toolkit and for the experiments described in this chapter, all three properties are used.

Ambient light is the simplest lighting model and is based on the notion that a non-directional source of light, spreading uniformly, exists and illuminates objects in a scene. Diffuse scattering occurs when incident light falls upon an object and radiates uniformly in all directions off the surface of that object. Real objects do not scatter light uniformly in all directions, therefore the diffuse light model in itself is not effective in revealing the form of shapes very clearly (Figure 6.6).

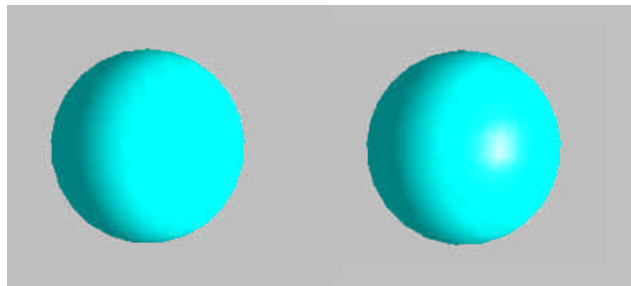


Figure 6.6 - Diffuse shading is applied to the object on the left and in comparison to the object on the right, it can be regarded as a disc. A specular highlight reveals the shape of the right hand object better as being a sphere.

In order to achieve more realism in a scene specular reflections can be added. Specular reflection creates highlights and is dependent on the direction of the viewer, i.e. the position of the highlight on the object changes with the viewing angle. Specular reflection is calculated by taking into account the relationship between the observer, the light

source, and the surface normal (see Figure 6.7). The variables in Figure 6.7 are as follows:

P - Point to be lit
N - Surface normal at point P
 i - incident ray
 r - vector of reflection
 v - vector from point P to viewer
 α - angle between viewer and direction of reflection

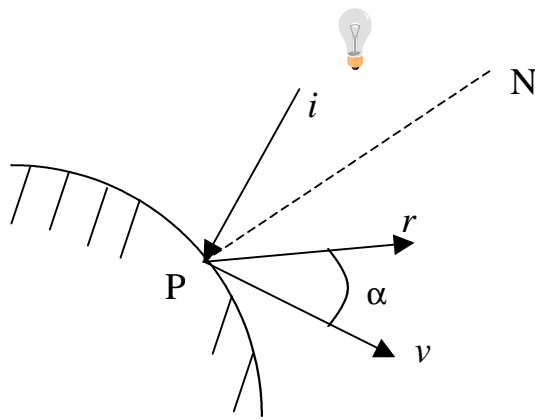


Figure 6.7 - Specular reflection from an object surface. Alpha is the angle between the reflected ray and the viewer.

In addition to applying the lighting model described above, the Geons were rendered to give a more realistic view, using a technique referred to as smooth shading. There are two types of smooth shading algorithms: Gouraud and Phong shading. OpenGL supports only the Gouraud model for shading. When computed correctly, this model of shading can add a high degree of realism to the scene. Gouraud smooth shading essentially de-emphasizes the edges between faces of polygons by smoothly interpolating colors between the vertices of the polygon.

In order to compute the correct color of the vertices, vertex normals are computed by averaging the normals of the adjacent polygons (Figure 6.8). The advantage of this technique is that smooth looking images are obtained with relatively few polygons. The averaging technique is illustrated in the figure below:

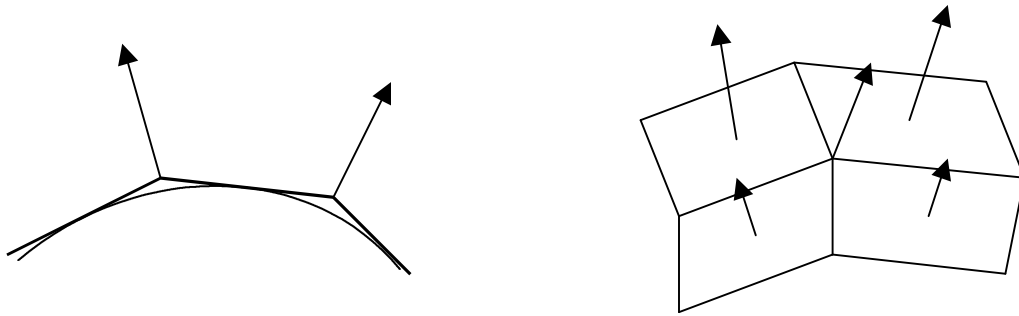


Figure 6.8 - a) Gouraud shading visually approximates an "underlying smooth" surface approximated by a polygon mesh. b) The normal at a point on the "smooth" surface is computed by averaging the normal of the adjacent polygons to that vertex.

The effect of smooth shading on the polygon vertices improves the appearance of those Geons that have round edges and curved axes, as shown in Figures 6.9 and 6.10.



Figure 6.9 - A set of Geons with curved cross sections and curved axes. The objects are modeled using polyhedral or flat-shading. Note the sharp appearances of the polygons underlying the objects. These Geons are modeled using 220 polygons.

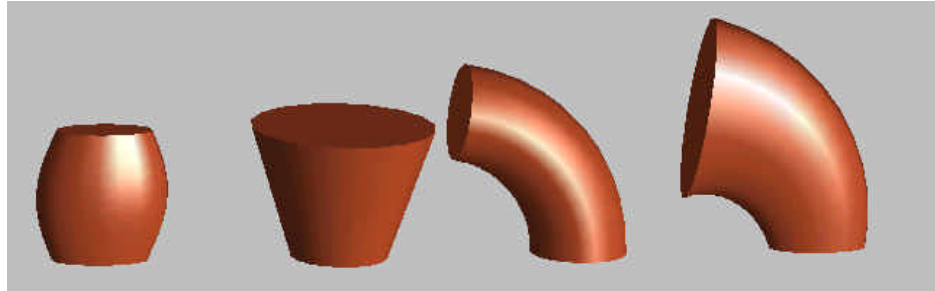


Figure 6.10 - Same Geons modeled with smooth Gouraud shading with 220 polygons each.

The remainder of this chapter describes experiments 4 and 5.

6.2 The importance of shading in Geon diagrams

Results for experiments 1, 2 and 3 suggest that using Geon primitives can enhance visual parsing and recall of node-link diagrams in comparison to box and line figures. In these experiments various factors could have contributed to these results aside from the use of Geon structures. One specifically is the general layout of the figure. The results therefore do not clearly demonstrate that Geon primitives were critical in the better performance obtained with the Geon diagrams. In the following sections two experiments that were designed to address the issue of the importance of 3D shapes for Geon diagrams visual interpretation and recall are described. 3D Geon shape is perceived using shape-from-shading perception and this was the variable that was experimentally manipulated.

The experiments are controlled such that the only variable factor being tested is the shading information provided using Geons. The experiments were designed to determine whether removing this shading component would impair the ability of Geons to convey

structure or to facilitate recall. To meet this goal, neither set of diagrams - Geon or silhouette - was equipped with surface attributes such as color or texture. Furthermore, because the 2D versus 3D was the primary issue, neither set of diagrams was equipped with labels

6.3 Experiment 4: Sub-Structure Identification with Geon Versus 2D-Silhouette Diagrams

The purpose of this experiment was to determine the ease with which people can identify sub-structures in Geon diagrams in comparison with equivalent 2D-silhouette diagrams. We measured the time and error rate for a subject to recognize a sub-structure of a diagram.

6.3.1 Hypothesis

It should be possible to identify sub-structures faster and more accurately in Geon diagrams with 3D shaded components in comparison to diagrams made from flat 2D outlined-silhouette components having the same shapes and layout.

6.3.2 Equipment for experiment 4

For this experiment all the diagrams were presented on a 17-inch computer monitor. The Geon toolkit ran on a Pentium 433 MHz Silicon Graphics machine in a Windows NT environment. The toolkit recorded the subjects' input times.

6.3.3 Method for Experiment 4

The method for this experiment was identical to that for Experiment 1.

Diagrams

Two sets of ten 2D outlined-silhouette diagrams were drawn using Adobe Photoshop™ and equivalent sets of Geon diagrams were constructed using the Geon toolkit. The diagrams were constructed using a one-to-one mapping between Geons and 2D outlined-silhouettes as shown in Figure 6.11. The first four entities in the mapping represented nodes in the diagrams and the elongated Geons represented the links (shown in the lower level of Figure 6.11). In the 2D version, Geons were mapped to a flat shaded gray pattern, with a black outline border, and black internal lines representing internal edges in the equivalent Geons. The outlined-silhouettes were drawn with the same outline shapes as the Geons with the exception of three silhouette links containing a line through the midsection of the entity. This mapping was also used in experiment 5 described in section 6.3.

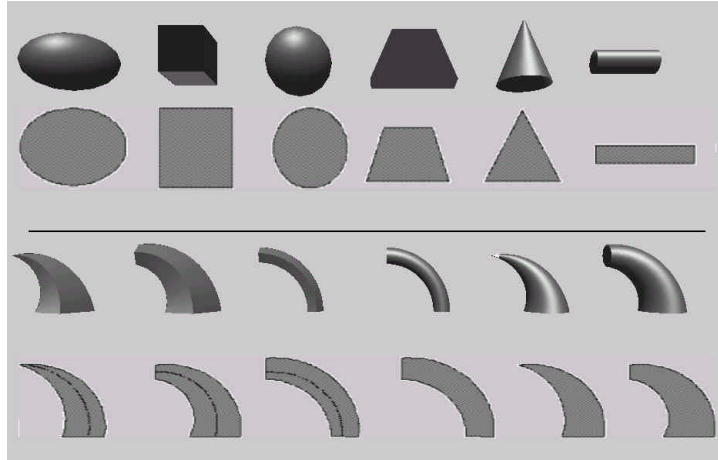


Figure 6.11 - One-to-one mapping between Geon primitives and 2D silhouettes used in Experiments 4 and 5

Examples of diagrams using this mapping are shown in Figure 6.12. For both sets of diagrams a sub-structure diagram was constructed; the sub-structure of the first set contained 3 components and that of the second set contained 7 components (Figure 6.13). The diagrams included a variety of combinations of the primitives in each case. The diagrams were presented on a computer screen.

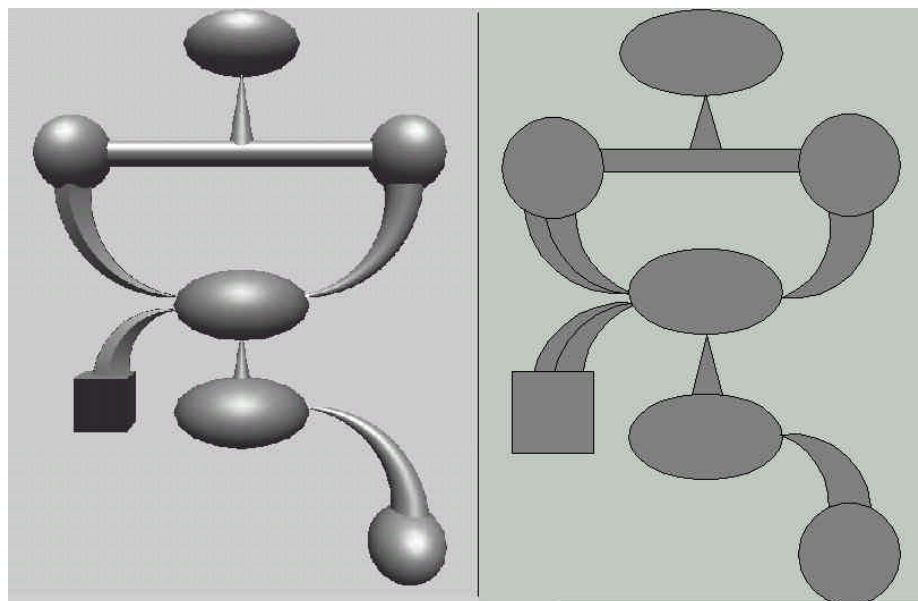


Figure 6.12 - Example of diagrams used for Experiment 4.

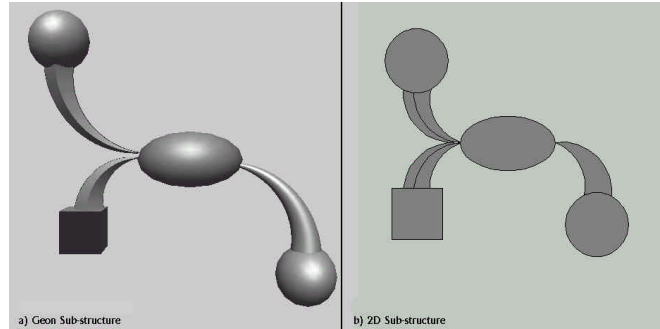


Figure 6.13 - Sub-structures to identify in Experiment 4

Procedure

The procedure used for this experiment was identical to that used for experiment 1. Subjects were first presented the sub-structure and subsequently had to indicate if the substructure was present in a series of larger diagrams. The order in which the sets were presented to subjects was randomly selected from one of the following sequences {G1,S1,S2,G2}, {S1,G1,G2,S2}, {S1,G1,S2,G2}, {G1,S1,G2,S2}, Where G denotes a set of Geon diagrams and S a set of silhouette diagrams. The 16 subjects were all computer science students who had not participated in previous studies.

6.3.4 Results of Experiment 4

Results are summarized in Table 6.1. These show that substructures were identified both faster and more accurately with the Geon diagrams. Of the 16 subjects, 9 correctly identified the sub-structure in more Geon diagrams than 2D outlined-silhouette diagrams, 3 subjects identified the sub-structure equally often with the Geon diagrams as with the

2D outlined-silhouettes, and the remaining 4 were more accurate with the 2D outlined-silhouettes. A sign test shows this to be significant ($p < 0.1$).

Because the data were highly skewed with outliers an analysis of variance model was fitted to the log (time to respond) data. The means were inverse transformed back to seconds for clarity of presentation. The analysis revealed that diagram type (Geon or 2D outlined-silhouette) was a significant factor in the time it took to correctly or incorrectly identify the sub-structure. $F(1,15) = 16.8$, $p < 0.001$. On average the subjects took 4.1 seconds to identify (correctly or incorrectly) the presence of the sub-structure in the Geon diagram and 5.3 seconds for the 2D outlined-silhouettes. Of the set of 16 subjects, 14 identified the Geon sub-structure faster than the 2D sub-structure.

The analysis of variance procedure also detected statistically significant differences in mean response times for the two types of sub-structures (3-component and 7-component). $F(1,15) = 14.2$, $p < 0.002$.

	Geon Diagram	2D Outlined- Silhouette
Identification Time (sec)	4.1 (± 0.5)	5.3 (± 0.8)
Error Rate	12.11%	19.24%

Table 6.1 - Summary of Results for Experiment 4

The results for the 3-component and 7-component sub-structures are summarized in Table 6.2 and the main effects are shown in Chart 6.1. There was a two-way interaction between the number of components in the sub-structure and the types of diagram (Geon vs. 2D outlined-silhouette). $F(3,29) = 32.36$, $p < 0.0001$. Subjects took significantly

longer to identify the presence or absence of a sub-structure in the 2D outlined-silhouette when it was comprised of more components.

	Geon Diagram	2D Outlined- Silhouette
3-Component (sec)	3.8	4.1
7-Component (sec)	4.3	6.5

Table 6.2 - Reaction Times for parsing diagrams with 3 and 7 components.

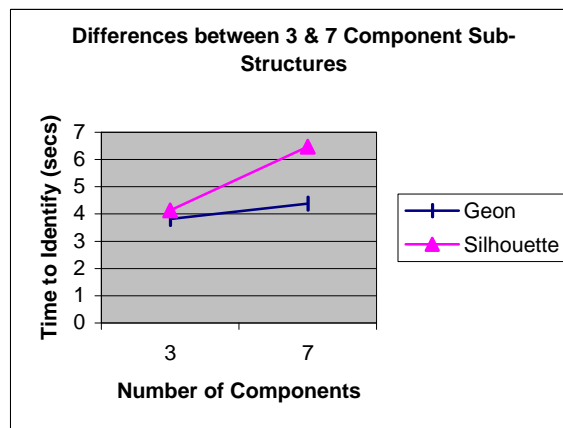


Chart 6.1 - The time required for parsing a sub-structure increases with the number of components.

Average response time is longer with 2D outline-silhouette than with Geon, and longer with more complex sub-structures than with simpler sub-structures. The analysis of variance suggests that there is an interaction between the response time and the number of components in the sub-structure. The difference between mean response times for Geon and 2D outlined-silhouette is greater for difficult diagrams than for simpler diagrams. These results thus indicate that shading plays a more significant role as the number of components in the diagram increases.

6.3.5 Discussion of Experiment 4

The results of this experiment support the hypothesis that sub-structures with shaded elements would be easier and faster to parse than the 2D silhouette structures. The results indicate that it takes longer to parse sub-structures with more components than with fewer. These results add to the evidence that 3D primitives are processed more rapidly than 2D primitives and these are consistent with those of Sun and Perona [SP96] in which priming speeds were slower when the stimuli contained more cubes.

It is also interesting to note that shading plays a more significant role in visually parsing sub-structures with more components. This could suggest that shading is more critical for more complex node-link structures.

6.4 Experiment 5: Recall of Geon Versus 2D-Silhouette

The purpose of this experiment was to determine whether Geon diagrams could be remembered more easily than 2D outlined-silhouette diagrams.

6.4.1 Hypothesis

Geon diagrams will be more accurately recalled after a brief exposure than comparable 2D diagrams.

6.4.2 Equipment for Experiment 5

The diagrams were presented on a projector screen 6x6 ft using transparencies. The projector screen was placed at a distance of 10-30 feet away from the subjects.

6.4.3 Method for Experiment 5

This experiment was formulated to compare the accuracy of recalling Geon diagrams versus 2D equivalent outlined-silhouette diagrams.

Diagrams

Using the one-to-one mapping convention of Geons to 2D shapes (Figure 6.11), a set of 14 Geon diagrams was produced using the Geon toolkit and a set of equivalent 2D silhouettes was drawn using Adobe PhotoshopTM. See Figure 6.14 for an example. Among the entire set, four diagrams were created with similar overall form but with different composite elements. This was done so that, subjects would be required to memorize the types of each of the elements in the diagrams, in addition to the overall shape of the diagrams. Both sets of diagrams were printed in black and white on 8.5 by 11" transparencies.

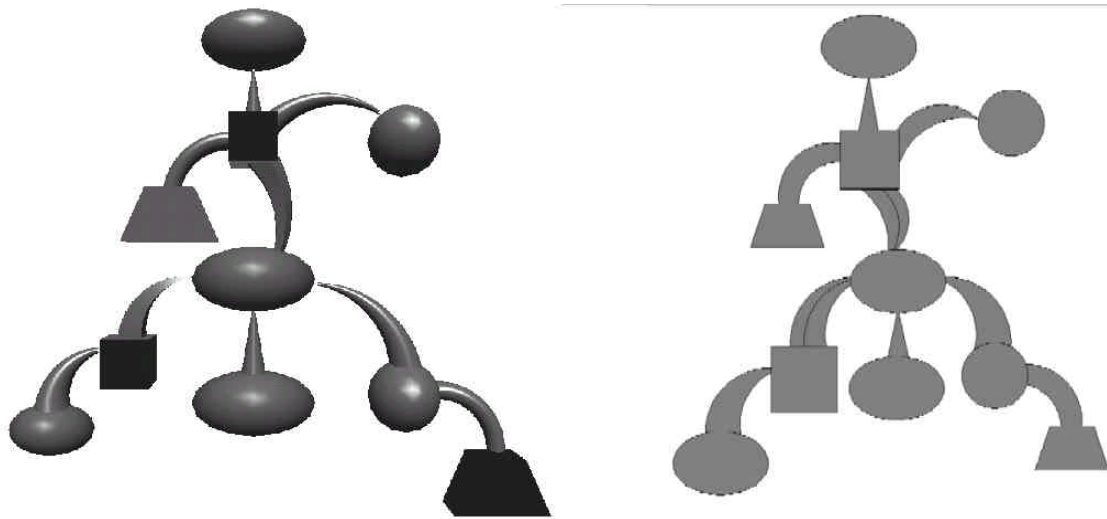


Figure 6.14 - Sample diagrams used in experiment 5.

Procedure

The experiment was conducted using two groups of students in senior level courses. The experiment was performed in a classroom setting with a class of students who had not seen these diagrams before (if a student had performed a similar experiment in a previous lecture, he/she was asked to indicate that on the handout and the result was later discarded).

At the beginning of the lecture the students in the first group were shown half (seven) of the set of Geon diagrams in random order for 15 seconds per diagram. After viewing the first half of the Geon diagrams, they were then presented with the seven 2D outlined-silhouette diagrams at intervals of 15 seconds, again randomly. At the end of the lecture, approximately fifty minutes later, the students were shown the full set of 14 Geon diagrams, then 14 2D outlined-silhouettes. Each diagram was shown for 10 seconds and the subject would indicate on a printed sheet whether that diagram had been part of the initial set. To counterbalance the first set of results, the same procedure was applied to a

second set of students with the 2D-silhouette diagrams being presented first and the Geon diagrams second.

Fifteen students participated with the Geon diagrams being presented first and 19 students participated with the 2D silhouettes presented first, giving a total of 34 subjects.

6.4.4 Results of Experiment 5

The results show that subjects made fewer errors in recalling Geon diagrams than they did for the equivalent 2D silhouettes: there was a 21.7% error rate for the Geon diagrams versus 31.2% for the 2D silhouettes. From the 34 subjects, 25 recalled correctly more Geon than 2D diagrams while 4 recalled correctly the same number of Geon diagrams as 2D-silhouette diagrams and 5 subjects recalled correctly more 2D-silhouette diagrams. A sign test shows this difference to be significant ($p < 0.05$).

	Geon Diagram	2D Outlined- Silhouette
Error Rate	21.7%	31.2%

Table 6.3 - Error rates in recalling Geon diagram vs. 2D outlined-silhouettes.

6.4.5 Discussion of Experiment 5

These results support the hypothesis that Geon diagrams are easier to remember than their equivalent 2D outlined-silhouette diagrams. There are two possible explanations. The first could be attributed to the subjects remembering the individual Geons better than their counterpart silhouette primitives, because of shading information on these Geons.

The second is that the shaded primitives could reveal better the structural information, thereby simplifying the task of recalling the Geon diagrams in comparison to their 2D silhouette counterparts. Further study would be required to investigate the alternatives.

6.5 Discussion

Overall, results of both experiments suggest that using 3D shaded primitives for diagram components can facilitate visual parsing and recall of the diagrams. Although, the results were not as striking as those from experiments 1-3, they strongly support the idea of using Geons instead of silhouette components as elements in node-link diagrams. These results support rule G5 on the use of Geons described in section 4.1.

The experiments described here could be further elaborated to investigate the effect of different types of shading, different directions of light source and other 3D depth cues such as cast shadows on diagram parsing and recall.

All the experiments until this point in the thesis have been primarily focused on investigating the use of 3D primitives for diagram structures. They suggest that using Geons for nodes and links in diagrams can improve the visual parsing and recall of diagram structures. The remainder of the thesis will focus on mapping semantics of object recognition to semantics of structured diagramming.

Chapter 7 - Evaluating Perceptual Semantics¹

Having shown that Geon diagrams are easy to remember and to visually parse (see Chapters 5 and 6), this chapter describes the investigation of the issue of mapping perceptual semantics to the structural representation of data elements. The work of Biederman and others suggests that certain spatial relationships, such as "on-top-of" or "unbalanced", may have a kind of immediately understandable perceptual meaning. If these perceptual semantics can be appropriately mapped to the semantics of systems modeling, then we can take advantage of these to make diagrams that are easier to read.

¹ The work in this chapter has been reported in "Irani, P., Tingley, M. and Ware, C. (2001), *Using Perceptual Syntax to Enhance Semantic Content in Diagrams*, IEEE Computer Graphics & Applications, 21:5, pp: 76-85." and in "Irani, P. Pourang (2002), *Learnability of Diagram Semantics*", Diagrams 2002 - International Conference on Theory and Application of Diagrams, Georgia, USA. (in press)"

Currently the most widely used graphical language for modeling complex systems is the Unified Modeling Language (UML). UML contains a suite of diagramming techniques that allow one to model various aspects of a software system [Fowler96], a real-time application [Douglass98], or an enterprise structure [Marshall99]. Its versatility in several application areas results from the rich semantics it seeks to model. For example, class diagrams in UML, model software structures and include methods for depicting inheritance and composition. When these semantics are used in the realm of enterprise modeling for example, UML can capture relationships between organizations or relationships between the corporation and its employees. However, although considerable attention has been given to making these UML notations general and complete, the actual choice of graphical notations appears to be somewhat arbitrary; only an expert in the field can easily read them.

This chapter first discusses aspects of structured object recognition theory with respect to the set of perceptual semantics that help in building a structural model of an object. It then presents the results of a set of experiments that show how a careful mapping of problem semantics to 3D diagram structures can make the meaning of the diagrams easier to understand with minimal training.

7.1 Perceptual Semantics

In explaining the stages involved in extracting primitives and structural information, Biederman suggests that structural description is not purely topological [Biederman87].

As Figure 7.1 illustrates, two objects, a human figure and a table, can have identical Geons and an identical skeleton in terms of its topology, but still be identified as very different objects. In describing the representational capacity of his set of Geons, Biederman suggests looking at the Geon structural description (GSD). A GSD is determined based on the spatial connections between pairs of Geons. These properties are largely viewpoint independent, preserve their two-dimensional silhouette structure, and are categorical [BG93]. To incorporate the perceptual semantics of Geon theory, we can use GSD properties for displaying diagram semantics. The following is a summary of Biederman's Geon structural description properties (SDPs). We have added an additional containment property.

SDP1) Shape is the primary property of a Geon. Color and texture are secondary properties.

SDP2) Verticality - Geon A can be ABOVE, BELOW or BESIDE Geon B.

SDP3) Centering - Objects can be connected on or off-center. For example, human legs are connected to the right and left of the bottom of the torso. Human arms are at the top of the torso.

SDP4) Connection relative to elongation - Most Geons are elongated, and connecting to the long face versus the short face has important perceptual semantics. Humans and four-legged animals are differentiated in this way.

SDP5) Relative size - One Geon is larger or smaller than another.

SDP6) Containment - An important perceptual task is identifying objects enclosed within larger components. This relationship is inherently hierarchical. However, strict containment can only be displayed using transparency.

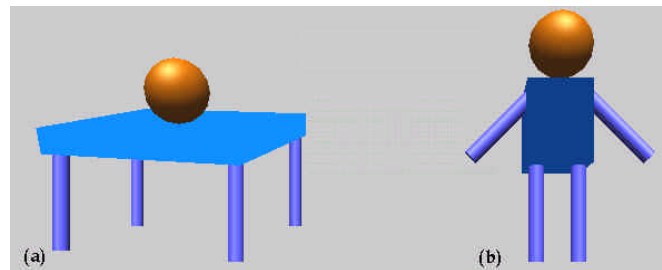


Figure 7.1 - Same Geons and topological arrangement result in different objects. The Geon Structural Description (GSD) is important for identification to take place.

A three-stage approach was adopted for conducting the research into the way Geon structural description properties can be used to convey meaning. In the first constructive stage, with experiment 6 as its main component, a mapping from the perceptual semantics to diagram semantics were constructed and evaluated. In the second validation stage, supported by experiment 7, diagrams were created using the best mappings derived from the constructive stage. They were then validated on how well they were perceptually understood. The representations were tested for their intuitiveness and ease of understanding. The third learnability stage, conducted through experiment 8, the ease of learning a diagrammatic notation based on perceptual principles was compared to the ease of learning with UML.

7.2 Experiment 6: Deriving Structures using Perceptual Semantics

To use the structural descriptive properties (SDPs) suggested by Biederman, it was important that they be mapped to concrete and specific semantics used by the general class of node-link diagrams. To build usable systems, the class of entity-relationship diagrams has evolved to capture models of the real worlds. Among these are ER diagrams, DFD diagrams, and UML diagrams (see chapter 2). The UML set of semantics has evolved to capture and unify the range of existing entity-relationship systems. As a result, UML tools provide systems architects with models to capture the various phases of a system implementation. Within the set of diagramming tools provided by UML, the class diagrams are most commonly used as a medium of communication between members of the software team. Recently, UML semantics also have begun being used as a medium of communication between the developers and users. For these reasons, the concepts captured by the UML system have been used in building a mapping of perceptual semantics to entity-relationship semantics.

UML class diagrams present a view of software structure; in particular, they depict the objects that exist, their internal structure, and their relationships with one another. They do not show temporal or causal information [Fowler96]. The notation has been derived from three main sources, Booch, Rumbaugh and OMT (through the efforts of the Object Management Group - OMG) to standardize software modeling semantics and notations. UML offers a rich semantic base which is used in deriving the visual representations that follow. This does not consist of augmenting UML notation by suggesting the use of

different shaped boxes or different types of edges. Instead, we are exploring the use of certain visual constructs to enhance the understanding and intuitiveness of semantics such as those available through UML. These findings can potentially be generalized to other diagramming applications.

In order to carry out this experiment, a number of different visual representations were constructed for each of the following modeling concepts:

- 1) Generalization (A is-a B),
- 2) Dependency (A depends-on B),
- 3) Strength of relationship (some relationships are stronger or weaker than others),
- 4) Multiplicity of relationship (e.g. one-to-many), and
- 5) Aggregation (A has-a B).

In each case a perceptual principle was used to construct at least one of the instances. The other members of the set were made up of what could be reasonable alternatives. Following the construction of representations, a multi-part evaluation study was conducted whose goal was to find out if subjects agreed on which mappings were the most effective. This experiment was made of five parts, each evaluating the representation that best fit one of the five semantics enumerated above. The experiment was conducted on 40 volunteer students, 20 of whom were familiar with software diagramming notations, in particular UML (experts). The remaining twenty students had not been exposed to any form of diagram modeling and had not been trained in understanding software semantics (novices). The interface to the experiment was a web

browser; HTML and Javascript were used to resize and randomize the appearances of the images. The entire experiment was conducted using a 17" monitor.

For clarity, each of the modeling concept mapping sub-experiments is separated in the following sections, together with the evaluation results.

7.2.1 Generalization

Generalization is the operation of grouping concepts that fit a given pattern. This capability of the human mind has led to the concretization and refinement of ideas over the centuries. In software modeling the semantic of generalization is used to classify objects based on their common functionality. The UML notation guide defines generalization as being the relationship between a more general element and a more specific instance of it [UML97]. It is commonly also referred to as inheritance (the specific object inherits properties of the general object) and is casually referred to as an "is-a" relationship. Thus objects can belong to a common class; for example, rotweilers, boxers and retrievers are all dogs.

The perceptual principle here is based on Biederman's claim that the primitive shapes play a primary role in object classification, whereas surface properties such as color and texture, play a secondary role (see SDP1 in section 7.1) [Biederman87]. The purpose of this part of the experiment is to determine whether shape has a stronger influence than color in classifying objects of the same kind. If shape is a better cue, then we could suggest using same shaped primitives to denote objects of the same kind. This is radically

different than what is available in UML, in which objects "of-the-same-kind" are connected by a solid line arrow with a closed arrowhead pointing to the more general class (see Figure 7.2).



Figure 7.2 - UML representation for Inheritance. A inherits from B or A is-a B.

7.2.1.1 Hypotheses for Generalization

- 1) Shape will be chosen over color to denote objects of the same kind.
- 2) There will be no differences in choice between novices and experts.

7.2.1.2 Method for Generalization

Two types of representations were considered, one which consisted of objects made of a single Geon primitive (one-component case) and the other which consisted of objects having two primitive shapes (two-component case). The procedure used for each case was similar and so they only differed in their representations.

Representations - One-Component case

Three sets of images were constructed. Each set contained eight shaped primitives labeled A to H (Figure 7.3). The types of primitives were different from one set to another. Each set contained two primitives with the same color but different shapes, and

two others that had the same shape but different surface color. For example in the set depicted in Figure 7.3, A and G were the same color (red) while E and C were the same shape (barrels). All the other shapes in the set had either a different color or different shape than the two pairs described above.

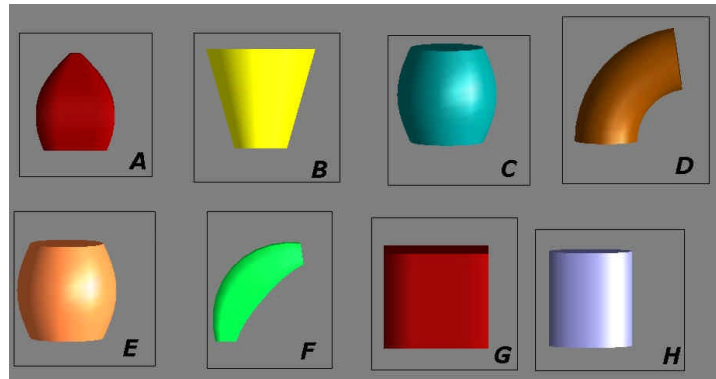


Figure 7.3 - Which two objects are of the same kind? Sample set of representations containing only one component. Contains a matching color pair (A&G) and a matching pair of shapes (C&E).

Representations - Two-Component case

In this case three sets of pictures were created each containing six images. Each image was comprised of two Geon primitives (Figure 7.4) labeled A to F. In all the sets we used one major (bigger) and one minor (smaller) Geon primitive to construct each image. Each set was created with two images containing major and minor components of the same color and different shapes (color-pair B and D) and two others whose minor and major components were made of the same shaped Geons with different colors (shape-pair C and E). All the other images in the set contained primitives whose major and minor components were colored differently than the selected color-pair and did not have the same primitives as the shape-pair.

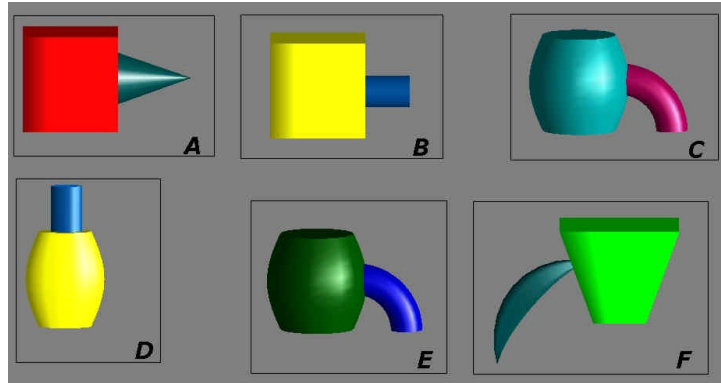


Figure 7.4 - Which two objects are of the same kind? Sample set of two components. Contains a pair with matching major and minor shaped components (C&E) and matching color coding on major and minor components (B&D).

Procedure

Subjects were shown the three sets of images for the one-component case and the three sets of images for the two-component case. They were asked to *"select using your best judgment the pair of images that are of the same kind"*, which they recorded on the experiment handout sheet. For each of the two cases, the three sets were shown in different order for all the subjects.

7.2.1.3 Results for Generalization

We did not observe any statistically significant differences between responses from novice and expert subjects. The test for null hypothesis of agreement between novices and experts yielded P-values of 0.403 for the one-component-case and 0.3091 for the two-component case. Therefore the results were combined. Overall, 92% of the responses favored same shaped objects, whereas 8% of the responses favored the same color. This is highly significant (a test of proportions gives $p < 0.0001$, confidence interval = 95%). This suggests that shape is a better stimulus for identifying objects of the same kind than is color.

7.2.1.4 Discussion of Generalization

The results supported both hypotheses. There were no observed differences between the choice of novices and experts. The observed responses strongly suggest using shape over color for representing objects of the same kind. The results parallel those of Biederman which suggest that for an entry level classification of objects, identification of Geon shapes is more important than surface features [Biederman87].

With the defined set of 36 Geon primitives [Biederman87] a large system is limited in the number of different distinct objects that can be represented. By using shape to identify objects of the same kind, a diagram could not be formed by more than 36 distinct objects. This limitation might be resolved by creating objects with pairs of primitives as was done for the two-component case of this part of the experiment. However, designing compound objects that express single-object identity might be challenging.

The use of same shaped objects for generalization will also be problematic when a system is modeled using multiple inheritance. Two objects of different shapes generalize into a third object whose representation may or may not be a combination of the earlier two. On the other hand, multiple inheritance is regarded by some as an improper structure in software modeling [Horstmann99]. This has led to the exclusion of this feature in Java, for example.

7.2.2 Dependency

In common parlance, dependency means a relationship in which an entity is supported by another. Children financially depend on their parents, humans depend on breathing clean air, and engines depend on fuel. Similarly, in software modeling terms, dependency describes a relationship in which changes to one component can cause changes to the state of the dependent component. Therefore the dependent module is unstable when changes are made to the entity upon which it depends. A goal in proper software modeling is to reduce the number of dependencies [Fowler96].

Certain spatial representations are easily visually recognized [GNC95]. These visual properties include such relations as "longer or shorter", "thinner or thicker", and "above or below". We commonly use one object resting on another as a metaphor for dependency, for example a University supports Learning. Biederman suggests that the spatial property of verticality (see SDP2 in 7.1) accounts for more than 80% of arrangements between visual components during object perception [Biederman87]. Glasgow and Papadias use such spatial properties to construct models that lower the computational costs for AI systems to retrieve and understand the representations of visual images [GP92]. In her study Petre concluded that "secondary notation" such as relative positioning of nodes (e.g. placing two linked or unlinked nodes near each other) plays an important role in conveying meaning [Petre95].

7.2.2.1 Hypotheses for Dependency

- 1) A representation based on the perceptual principle of an object resting-on another will represent best the semantic of dependency.
- 2) There will be no differences in choice between novices and experts.

7.2.2.2 Method for Dependency

The purpose of this part of the experiment was to derive a representation that was best suited for understanding the semantics of dependency. If our perceptual system has deeply engrained within it certain spatial properties to describe and classify objects then perhaps we could use such properties to depict dependency.

Representations

Five different ways of representing the dependency relationship were constructed, as illustrated in Figure 7.5. To show that the red cylinder depends-on the blue barrel, these representations consisted of a broken tube (A), a connected tube (B), the cylinder on-top-of the barrel (C), disconnected objects (D), and the cylinder on-the-bottom-of the barrel (E). Representation A most closely resembles the representation of dependency used in UML which consists of a dashed line with an open arrowhead going from the dependent to the depended (see Figure 7.6) [Fowler96].

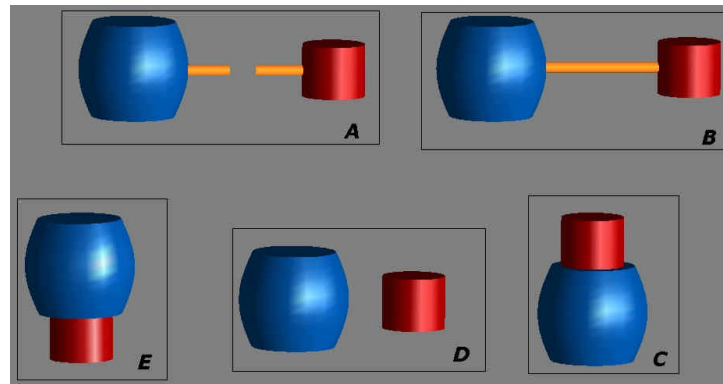


Figure 7.5 - Which representation best denotes the red cylinder depends-on the blue barrel? A sample set containing representations that were ranked for showing dependency.



Figure 7.6 - UML representation of Dependency. A depends-on B.

Procedure

Subjects were asked to *rank from 1-5 (best-to-worst) the representation denoting that one object depends on another*. In the case of the representations illustrated in Figure 7.5, the relationship was that of the red cylinder depending on the blue barrel. They recorded their rankings for all three sets of representations. Each subject was shown the representations in a different random order.

7.2.2.3 Results for Dependency

A χ^2 test on the results shows that there were no statistically significant differences in selecting the best representations between novices and experts (P-value of 0.49 for null

hypothesis of agreement between novices and experts). Therefore the results were combined. The average rankings of all 40 subjects is shown in Chart 7.1. A top-down test of correlation on the average rankings shows a strong agreement between all 40 subjects for the best-ranked representations (P-value < 0.0001 for null hypothesis of no correlation between rankings chosen by 40 subjects). As seen in the chart below, dependency is best depicted using the on-top-of representation (C). This representation is significantly better than the second best representation of a connected tube (B) (with 95% confidence, the probability that any subject, novice or expert, would choose C over B is between 0.58 and 0.86).

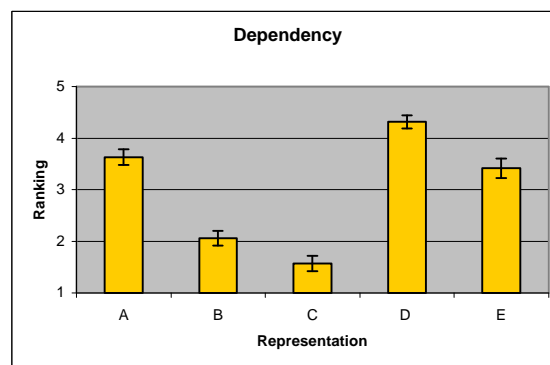


Chart 7.1 - Average rankings for dependency. 1 - best, 5 - worst.

7.2.2.4 Discussions of Dependency

The results of the experiment support the hypotheses. There were no statistically significant differences between novices and experts. The representation of on-top-of is also based on the perceptual principle of verticality. Representing dependency using the spatial property on-top-of is not commonly seen in software structure diagrams but can be found in other types of visual representations. Organizational charts, which stack

different parts of the corporation on a pyramid, make use of such a spatial organization between the represented entities. In such diagrams there is an implicit assumption of the dependency between objects on-top-of and on-the-bottom-of the pyramid. A drawback of such a representation is the amount of space required to show several entities on-top-of any given object.

7.2.3 Relationship Strength

A common semantic found in structured diagrams is strong and weak relationships. This semantic is not directly modeled using UML but is common in other types of structured diagrams such as Entity Relationship diagrams [Chen76]. In UML class diagrams, a strong aggregation is referred to as a composition and as such can denote a strong relationship between two entities. Biederman suggests that during object recognition our perceptual system differentiates parts of an object based on their relative sizes (see SDP5).

7.2.3.1 Hypotheses for Relationship Strength

- 1) A thick connecting Geon will be seen as representing a strong relationship in comparison to a thin connecting Geon.
- 2) There will be no differences in choice between novices and experts.

7.2.3.2 Method for Relationship Strength

The purpose of this sub-experiment was to examine whether the perceptual mechanism of discrimination based on relative ratios can be applied toward the semantic of relationship strength.

Representations

Six different ways of representing the strength of a relationship were created, as illustrated in Figure 7.7. These representations consisted of a thick connected tube (A), adjacent entities (B), a long connected tube (C), a conic connection (D), disconnected entities (E), and a short connected tube (F).

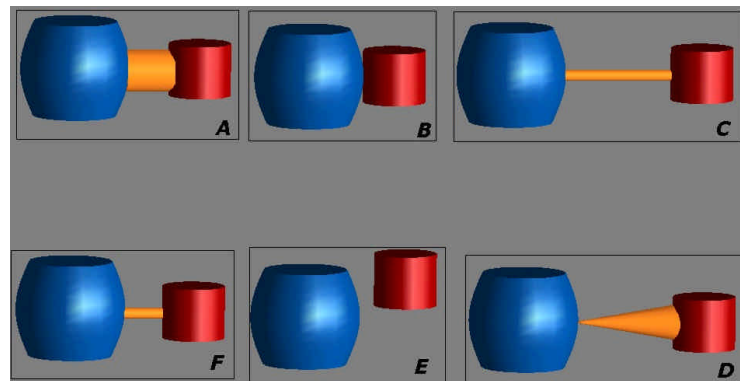


Figure 7.7 - Which image shows best that the red cylinder and blue barrel are strongly related? One of the three sets being ranked for showing relationship strength.

Procedure

Subjects were asked to *rank from 1-6 (best-to-worst) the representation denoting that one object is strongly related to another*. For the example in Figure 7.7, subjects were asked to rank the representations according to how effectively each denoted a strong

relationship between the blue barrel and red cylinder. For each subject, the representations appeared in a different random order.

7.2.3.3 Results for Relationship Strength

A χ^2 test on the results showed that novices and experts were in perfect agreement on their choice of rankings (test of null hypothesis of agreement yields P-value of 1.0). The results were thus combined. The average rankings for 40 subjects are summarized in Chart 7.2. A top-down test of correlation shows that all 40 subjects were in strong agreement in selecting the top best rankings (P-value < 0.0001 for null hypothesis of no correlation between rankings chosen by 40 subjects) and, as depicted in Chart 7.2, the best representation for showing that two entities are strongly related is a thick connection between the objects (A). This representation is significantly better than its alternative representation (B); with 95% confidence, the probability that any subject, novice or expert, chooses A over F is greater than 0.99. These results strongly suggest that using a relatively larger size for a connection can depict strength of relationship.

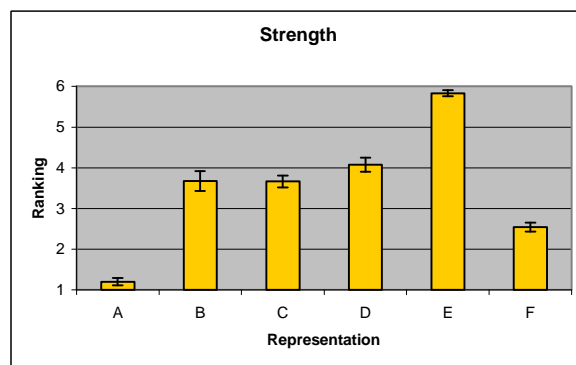


Chart 7.2 - Average rankings for the six representations for relationship strength. 1 - best, 6 - worst.

7.2.3.4 Discussion of *Relationship Strength*

The results strongly support both hypotheses. Firstly, novices and experts agreed on the same representation. Secondly, 99% of all respondents agreed on a thick connecting tube between entities for representing stronger relationships suggesting that such a structure is better suited than a short connection (representation F). From these results we also observe that adjacent entities is a weaker representation for representing strength (representation B). This could be due to the fact that the concept of a relationship is primarily visible using some form of connection and this connection is not explicit using adjacency.

7.2.4 Multiplicity (or cardinality)

Numbers are a relatively recent invention in human evolution. In many instances skills involving counting or knowing the exact quantity is not essential in providing a stable image. For example, a shepherd need not count to know whether his group is complete [Arnheim69]. When children are asked to copy a figure made with counters, they do not use the exact number of counters (even if they know to count) but are able to reproduce the shape of the figure [Piaget52].

A common method for representing this semantic attribute is the use of an asterix (*) or an exact numeral (such as 1, 2, etc.) over the link and on the side of the entity which is

associated in multiples (see Figure 7.8). Other common notations are 1..*, 0..* or *. However, numbers are learnt symbols and therefore not perceptually immediate.



Figure 7.8 - UML representation for Multiplicity. A is associated to-many B.

7.2.4.1 Hypotheses for Multiplicity

- 1) A representation based on using multiple objects will best depict the semantic of multiplicity of relationship.
- 2) There will be no differences in choice between novices and experts.

7.2.4.2 Method for Multiplicity

The purpose of this part of the experiment was to derive a representation that would represent that an entity is associated with multiple instances of another.

Representations

Five different ways of representing the multiplicity attribute of a relationship were constructed, as illustrated in Figure 7.9. These representations consisted of a conic connection with multiple glyphs (A), disjoint objects (B), multiple connecting tubes (C), single connecting tube (D), simple conic connection (E). In this illustration, the

representations were created to depict the blue barrel being associated to multiple instances of the red horn.

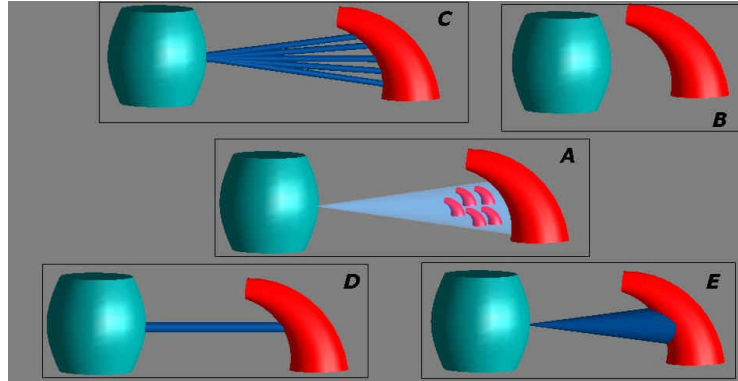


Figure 7.9 - Which representation best depicts that the blue barrel is associated to multiple instances of the red horn? Sample set used in ranking representations of multiplicity.

Procedure

Subjects were asked to *rank from 1-5 (best-to-worst) the representation denoting that one object is associated to multiple copies of another object*. They were presented with three sets of images. For the set shown in Figure 7.9 subjects were asked to rank the representation that best denotes the notion that the blue barrel is associated to multiple copies of the red horn. For each subject the order of the representations was random.

7.2.4.3 Results for Multiplicity

A χ^2 test on the results shows that both groups of subjects agree on rankings (test of null hypothesis of agreement yields P-value of 0.18). The results are therefore combined and summarized in Chart 7.3. A top-down test of correlation on the average rankings shows strong agreement between all 40 on subjects on their rankings (P-value < 0.0001 for null

hypothesis of no correlation between rankings chosen by 40 subjects). As seen in Chart 7.3, multiplicity is best represented using multiple connecting tubes (C). This representation is significantly better than its alternative (A) (with 95% confidence, the probability that any subject, novice or expert, chooses C over A is between 0.52 and 0.82).

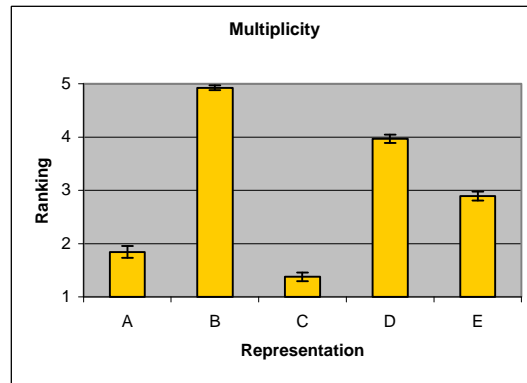


Chart 7.3 - Average rankings for the six representations for relationship strength. 1 - best, 5 - worst.

7.2.4.4 Discussion of Multiplicity

The results support both hypotheses. There are no statistical significant differences between novice and expert choice of representation. Moreover, the notion of multiples is best depicted showing many instances of the relationship.

There are obvious problems with this representation of multiplicity. People are likely to interpret multiple representations in an overly literal sense. For example, if three branches are shown, exactly three instances will be inferred. On the other hand there is a theory that humans and animals have a perceptual sense of numbers, but it is strictly limited.

We may be only able to naturally separate 1, 2, and possibly 3 objects. If there are more objects they are simply perceived as a many [Stanislas98].

7.2.5 Aggregation

In UML methodology, aggregation describes a special form of association in which an object contains another. In software engineering terms, aggregation is also referred to as a "has-a" relationship. For example, an organization has-a president. The UML representation for an aggregation is shown in Figure 7.10 below. A strong aggregation is referred to as a composition. There is evidence suggesting that our perceptual system is capable of separating an object when it is seen as being contained within other objects.



Figure 7.10 - UML representation of Aggregation. A contains B or A has-a B.

7.2.5.1 Hypotheses for Aggregation

- 1) A representation based on the perceptual principle of an object contained within another will represent best the semantic of aggregation.
- 2) There will be no differences in choice between novices and experts.

7.2.5.2 Method for Aggregation

The purpose of this part of the experiment was to establish a representation suitable to denote an aggregation, part-of, or containment relation.

Representations

Five different ways of representing aggregation were constructed, as illustrated in Figure 7.11. These representations consisted of a connected tube (A), an object-on-top-of (B), disjoint objects (C), connection with containment (D), and containment (E).

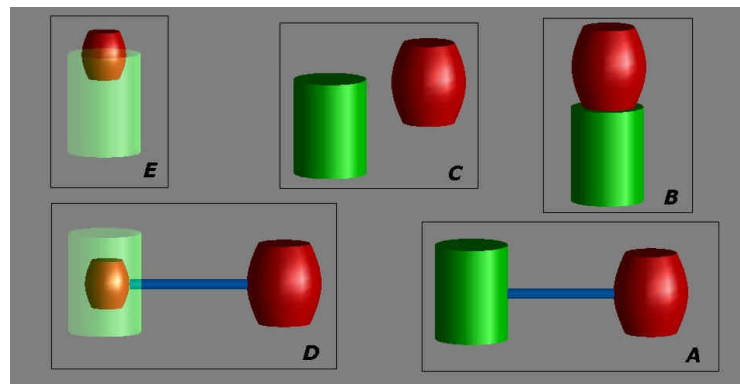


Figure 7.11 - Which image best depicts that the red barrel is contained within the green cylinder? Sample set used in ranking aggregation.

Procedure

The subjects were asked to *rank from 1-5 (best-to-worst) the representation denoting that one object is contained within another*. In the case of the representations given in the set in Figure 7.11, subjects were asked to rank the representation that best denoted that the red barrel is contained within the green cylinder.

7.2.5.3 Results for Aggregation

A χ^2 test on the results shows that both groups of subjects strongly agree on rankings (test of null hypothesis of agreement yields P-value of 0.74). This result allows us to average the rankings of all 40 subjects as summarized in Chart 7.4. A top-down test of correlation on the average rankings shows that subjects are consistent in their ranking (P-value < 0.0001 for null hypothesis of no correlation between rankings chosen by 40 subjects). The results show that the best depiction for aggregation is a connection with containment (D). This representation is not significantly better than its alternative (E) (with 95% confidence, the probability that any subject, novice or expert, chooses D over E is between 0.47 and 0.76).

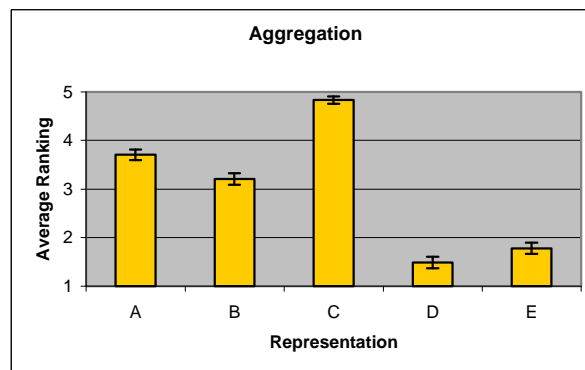


Chart 7.4 - Average rankings for the five representations for aggregation. 1 - best, 5 - worst.

7.2.5.4 Discussion of Aggregation

The results support both hypotheses. It is interesting to note that the results suggest that both representations D and E are good for representing aggregation. The pure form of containment or aggregation could be shown with the representation of the object fully

contained. However, for the experiment, inserting the barrel would result in an ambiguity. Due to transparency effects, the barrel would no longer be "red". This would confound the question which was clearly based on both the shape and color of primitives (eg. *the representation denoting that the red barrel is contained within the green cylinder*). On the other hand, representing a diagram coherently with aggregation shown as an object fully contained could be difficult. For example, combining dependency with aggregation would result in placing the depended within the object containing the dependent. A containment with connection (representation D) is not "perceptually pure" as one could interpret the possibility of having two distinct objects (in this case small and large barrels). However, as the results suggest, such a representation is not misleading.

7.3 The Perceptual Syntax

Figure 7.12 summarizes the set of best representations derived from the experiment. This can be regarded as the basis for a kind of natural visual semantics for drawing diagrams, which will be easily interpreted. In the following section we present the full set of rules defining Geon diagrams.


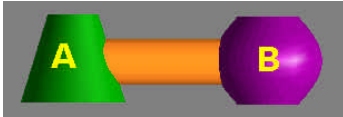
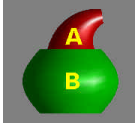
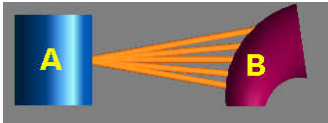

<p><u>Generalization:</u> A "is-a" B</p>  <p>Same shaped primitives (SDP1)</p>	<p><u>Strength:</u> A and B are "strongly" related</p>  <p>Thick link connects A and B (SDP5)</p>
<p><u>Dependency:</u> A "depends on" B</p>  <p>A is on-top of B (SDP2)</p>	
<p><u>Multiplicity:</u> A is associated to "multiple" copies of B</p>  <p>Multiple connections</p>	<p><u>Aggregation:</u> A "has-a" B</p>  <p>Containment with connection (SDP6)</p>

Figure 7.12 - Perceptual notation for showing each of the five semantics investigated in our experiment: Generalization, Relationship Strength, Dependency, Multiplicity, and Aggregation.

As an example, Figure 7.13 shows the representation of several related entities of a *Space Center*. The *Space Center* has-many *Buildings* (containment with multiple connecting lines), and has-many *Spacecraft*. A *Gas Station* and a *Lab* are two different types of buildings (same shape primitive as *Building*). The *Gas Station* has *Fuel* (containment with connection). *Shuttles* are also a type of *Spacecraft* (same shape primitive). *Shuttles* have-many *Wings*, and has-one *Engine*. The *Engine* depends on *Fuel* (depicted on-top-of the *Fuel* entity).

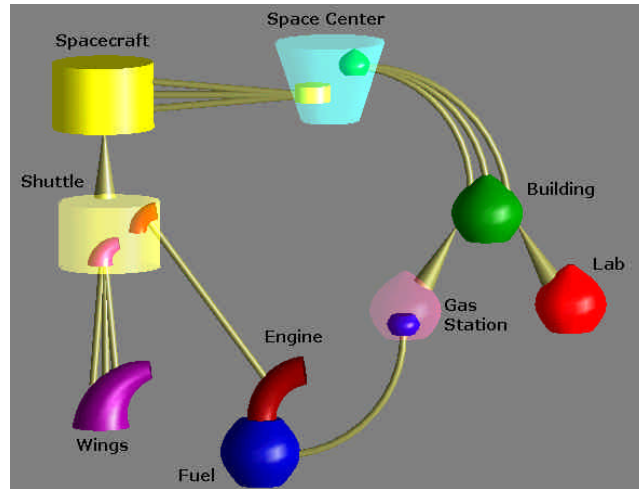


Figure 7.13 - Representing some related entities in a system describing a Space Center.

7.4 Experiment 7: Validating the Perceptual Syntax

In order to evaluate the Geon diagram syntax and semantics we created Geon diagrams to model real world examples. For example, Figure 7.14 illustrates one of the diagrams modeling an academic conference whose components are attendees, speakers, A/V equipment, etc. Using this diagram and others like it, we conducted an experiment to evaluate whether the notations of the Geon diagrams were intuitive and easily describable.

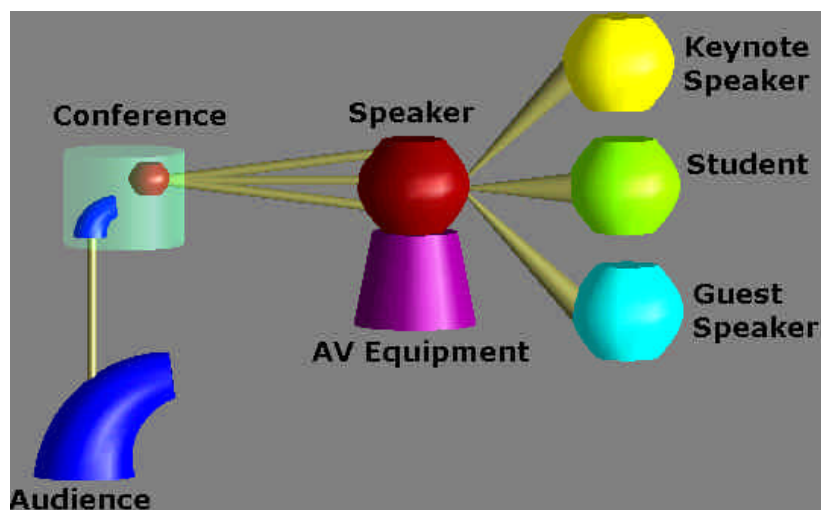


Figure 7.14 - Geon representation for communicating structural information between representable entities in a conference.

7.4.1 Hypothesis for Experiment 7

Without training, Geon diagrams are more intuitive and easier to understand than UML diagrams.

7.4.2 Equipment for Experiment 7

The diagrams were presented on a projector screen 6x6 ft using a laptop. The projector screen was placed at a distance of 10-30 feet away from the subjects.

7.4.3 Method for Experiment 7

The purpose of this experiment was to validate the perceptual syntax derived from the set of experiments described in section 7.2.

Diagrams

A set of 3 UML diagrams and equivalent Geon diagrams were created for the experiment. The diagrams depicted real world models an example of which is shown in Figure 7.14. The entities in the diagrams were arbitrarily labeled, as we were primarily validating the syntax and did not want to influence the choice of semantic based on the labeling (Figure 7.15 and Figure 7.16). All the diagrams were created using the semantics of generalization (inheritance or is-of the same kind-as), dependency (depends-on), aggregation (has-one-of), and multiplicity (has-many-of) and each diagram contained at most one of each type of relationship.

Procedure

In a classroom setting, the subjects were presented with the UML diagrams and equivalent set of Geon diagrams in an interleaved fashion (i.e. first a UML and then a Geon diagram or the opposite). The subjects were generally second year students without

any previous experience with either type of diagramming convention. They were asked to describe the relationships between several pairs of entities in the diagram using multiple choice answers. The choices included:

- is of the same kind as
- depends on
- has many
- has one

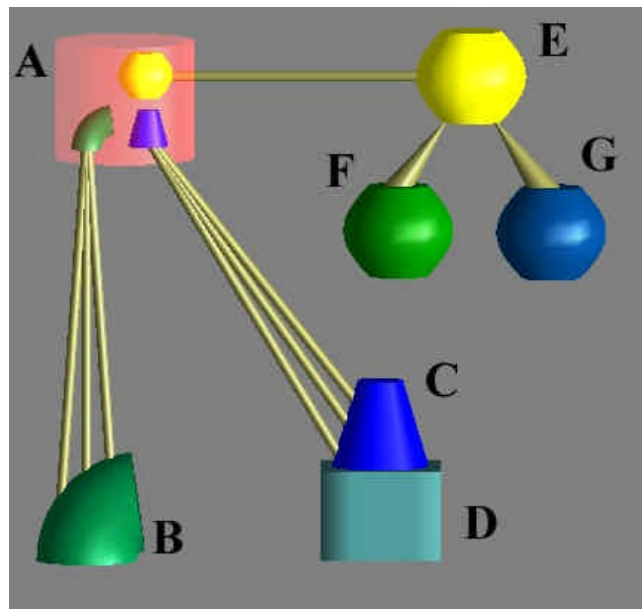


Figure 7.15 - Geon diagram representation used in validating the syntax.

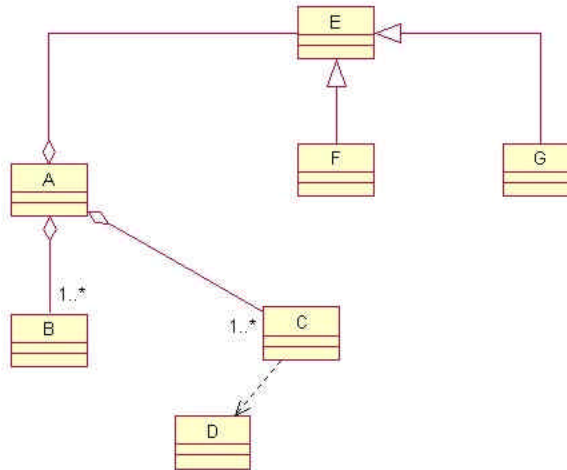


Figure 7.16 - UML representation equivalent to the Geon representation of figure 7.15.

In total there were 11 relationships to match with the multiple choices given above. Subjects were given three minutes to match the descriptions to the represented relationships for each diagram.

Subjects

A group of 35 students (different from the 40 subjects used earlier) who were unfamiliar with software modeling semantics found in UML participated in the experiment.

7.4.4 Results for Experiment 7

The results are summarized in the Table 7.1. They show that there were almost five times as many errors deciphering relationships between entities using UML notation than using the perceptual syntax.

	Geon Diagram	UML Diagram
Error Rate	11.5%	53.6%

Table 7.1 - Comparison of error rates in identifying relationships in Geon diagrams vs. UML diagrams.

From the set of 35 subjects 31 had fewer errors with the Geon diagram than with the UML notation, 3 performed better with the UML notation and 1 subject performed equally well with both notations. A sign test shows that the subjects performed significantly better with the Geon notation ($p < 0.0001$).

7.4.5 Discussion of Experiment 7

The results support the hypothesis that the Geon notation is more intuitive than that of UML. The results are striking considering that without *a priori* knowledge of either the semantics or the notation of UML, subjects were capable of matching the visual structures to the choices provided. These results validate the visual representations obtained from experiment 6. However, they are not sufficient to say anything about the comprehension of the semantics in a given problem domain. They simply suggest that the Geon notations match a set of perceptual principles that have a common semantic. Experiment 8, investigates the ease of learning and matching problem descriptions to diagrammatic representations based on perceptual principles.

7.5 Experiment 8: Learnability of the Perceptual Syntax

The purpose of the experiment was to compare the explanatory effectiveness of Geon diagrams to that of equivalent UML diagrams. The error rate for a subject matching a given problem to one correct diagram out of a set of possible diagrams was measured. The experiment was conducted in two phases. In the first phase (or learning phase),

subjects were instructed in the notations and semantics of object modeling. In the second phase (matching phase), subjects performed the experiment.

7.5.1 Hypothesis for Experiment 8

It should be possible to learn, recall, and match problem descriptions more accurately to diagrams created with the perceptual notation presented above.

7.5.2 Equipment for Experiment 8

For the learning phase, the instruction was presented on a projector screen 6x6 ft using a laptop. The projector screen was placed at a distance of 10-30 feet away from the subjects. Subjects performed the matching task on a 17-inch monitor.

7.5.3 Method for the Learning Phase

Subjects were given an hour-long instruction on the various semantics and their respective notations in UML and in Geon diagrams. The instruction consisted of describing the semantics and providing illustrations of the use of these notations through various diagrams. The notations for each of the semantics were presented side-by-side and given equal time. The subjects were asked to present themselves a week later for the experiment. The instruction was provided at a general level to facilitate comprehension by all subjects, including those who were presented modeling concepts for the first time. The session also included a question and answer period through which more clarification was provided. Appendix A contains the slides used in this part of the experiment.

7.5.4 Method for the Matching Phase

This phase required the subjects to analyze and match the problems to the appropriate diagrams. By asking subjects to return a week later, we were able to test the long-term retention of the perceptual and conventional notations that were described to them during the study phase.

Diagrams

Five problem descriptions representative of real world examples were constructed. These incorporated the semantics of generalization, dependency, multiplicity, and aggregation. The semantics in the problems were clearly presented using their common terminology. For example, to describe related entities of a neighborhood the following text was provided: *"The neighborhood depends on the city for clean water, sewage and garbage disposal. Many families live in this neighborhood. The neighborhood has a school and a pharmacy. It also has several types of stores: a grocery store, a convenience store, and a bakery."* The problem descriptions were created with a comparable number of relationships. For each problem description, a set of four UML diagrams and a set of their four equivalent Geon counterparts were created. Only one of the four diagrams accurately depicted the relationships in the corresponding problem description. The remaining three diagrams misrepresented several relationships. All four diagrams in each set were randomly numbered from 1 to 4.

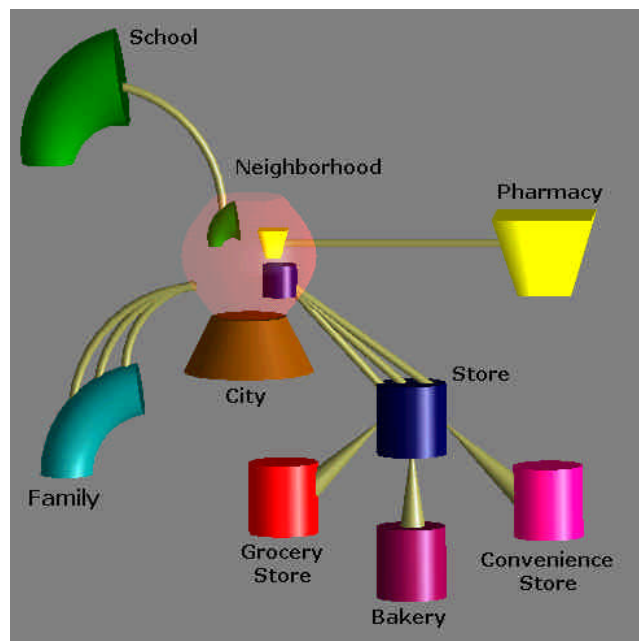
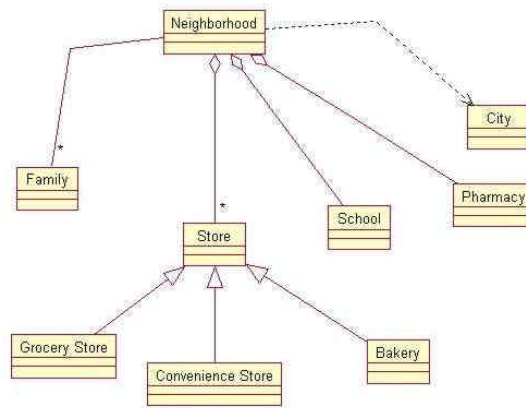


Figure 7.17 - Sample UML and equivalent Geon diagram for representing entities of a neighborhood.

Procedure

After reading each problem description, the subject was asked to match one of the four diagrams created for that problem. The subject marked on the hand-out sheet the number of the diagram. The problem descriptions were available to the subjects while reading the diagrams, and so they could occasionally consult the description. Therefore we were not

testing subject memory of a given problem text. Half the subjects were matching the UML diagrams first and the other half matched the Geon diagrams first. Subjects were given 2 minutes for analyzing and matching each problem description.

Subjects

Twenty-six paid volunteers participated in the experiment. Twelve subjects had previous exposure to UML diagrams through a software engineering course (experts) and the other 14 had never been exposed to UML semantics (novices). None of the subjects had prior exposure to Geon diagrams.

7.5.5 Results for Experiment 8

Each subject was given a score of 1 if they matched the correct diagram to the problem description or a score of 0 otherwise. Results are summarized in Table 7.2, which reports error rates by level of experience and type of diagram. The results are obtained by averaging each subject's scores. A One-Sample T-Test (or Sign Test) statistically shows that all subjects performed better with the Geon diagrams ($p < 0.0001$). Overall, experts performed better with both diagramming conventions. However, both experts and novices performed approximately 22% better in mean error rates with the Geon diagrams. A Two-Sample T-Test (or Mann-Whitney Test) shows that this difference is not affected by the subjects' level of experience with UML diagrams (p-values are 0.704 and 0.425 respectively).

	Geon Diagram	UML Diagram
Novices	22.3%	44.3%
Experts	2.9%	25.9%
Combined	14.6%	36.2%

Table 7.2 - Average error rates of matching Geon and UML diagrams to problem descriptions.

The most striking result is the performance of expert subjects with respect to the interpretation of Geon diagrams (error rate 2.9%). This suggests that although subjects may be well versed in UML, they still preferred and performed better in interpreting the Geon diagrams (more than 9 times better in interpreting the Geon diagram). These subjects also performed better than the novice subjects with respect to both the Geon and UML scores. Overall, combining the results we can say that there were more than twice as many errors in analyzing and matching the UML diagrams than the Geon diagrams.

7.5.7 Discussion of Experiment 8

The results support the hypothesis that with the Geon notation users will learn, recall and match problem descriptions more accurately than with the UML notation. Both experts and novices performed significantly better with the Geon representation, suggesting that such a notation is appropriate for communicating abstract models between users with diverse backgrounds.

It is usually the case that systems are described in terms of diagrammatic representations specifically for verifying the transfer of knowledge between end-users and systems architects. The results of this experiment suggest that perception based notations can facilitate the verification process between domain experts (end-users) and system architects.

It was interesting to observe one particular subject who was sketching out the Geon diagram on paper while reading the descriptions before matching the UML diagrams to

the problem descriptions. This could suggest that applying the Geon notation facilitated recall of the semantics. Subsequently, this subject performed equally well with both sets of notations.

7.6 Applying Theories of Perception to Drawing Diagrams

Together with results from previous studies, the results described above help in refining the rules for what is called the "Geon diagram". In addition to the five rules relating to the use of Geons as 3D primitives (section 4.1), three rules relating to the layout of the Geon structure (section 4.1), the results of the experiments described in this chapter provide nine additional rules for representing certain diagram semantics. These rules extend the syntax of structural description to include mapping of semantics to data elements. The set of "naturally" occurring semantic rules that can be used for constructing effective diagrams follow:

SM1: Use Shape to represent Generality - Geons with same structural geometrical composition (or shape) can be used to denote objects of the same kind.

SM2: Use on-top-of to represent Dependency - If Geon A is on top of Geon B this suggests that Geon A is supported by Geon B. In addition gravity determines that structures are perceived as either being stable or unstable.

SM3: Use Enclosure to representation Aggregation - shows that Geon A is contained within Geon B. Syntactically this can be shown as an internal component attached to the same primitive Geon on the outside.

SM4: Use multiple links to represent Multiplicity - A series of attachments can best denote multiple associations between two entities.

SM5: Use Thickness to represent Strength of connection - Using a thicker connection as opposed to a thinner one can denote a stronger relationship between two entities.

The remaining rules have also been constructed but have not yet been experimentally validated.

SM6: Use linear arrangement of Geons to show sequence - Geons arranged in a line become a metaphor for a chain of operations or some other linear structure.

SM7: Use symmetrical arrangement of Geons to show Symmetry - Some information structures have symmetry; symmetrical arrangement of Geons should be used to show this.

SM8: Place Geons centrally to show higher relevance - If a component is of central importance to a structure this can be illustrated through its position and by the location of the interconnections.

SM9: Use Size to represent Greater Magnitude or Quantity - Understanding of superiority in areas such as finances, economics, and politics and be mapped to larger components.

7.7 General Discussion

The three experiments described in this chapter were aimed at deriving and validating a mapping of perceptual semantics into diagram structures to enhance diagram semantics.

The results complement those described in the previous chapters which suggest using

shaded 3D components such as Geons for creating diagram structures. From the theory of recognition-by-components, a set of relative arrangement between Geons (SDPs) assist in recognizing and classifying objects. These Structural Descriptive Properties can be mapped to commonly used semantics to make the notation of diagrams more intuitive, easier to understand and learn, and easier to recall than arbitrarily assigned mappings.

The major contribution of the results described in this chapter has been to add semantics to the Geon diagram design rules by developing ways of representing the abstract relationships such as generalization, dependency, relationship strength, multiplicity and aggregation. The experimental results suggest that the best representations can be understood intuitively without training. This establishes a kind of natural semantics for diagrams. In comparison with UML diagrams, it is found that using the perceptual semantic rules greatly facilitates the task of identifying relationships between elements. However, the semantic mappings are far from complete. For example a study of the perceived meaning of different kinds of linking objects: broken tubes, dashed tubes, cone-shaped tubes, transparent tubes might further enrich our graphical vocabulary.

As noted in the discussion following the description of each semantic, there is a possible tradeoff between the very concrete nature of Geon diagrams and the representation of certain abstractions. If, for example, size is used to represent magnitude, then it becomes difficult to represent objects or concepts that have arbitrary magnitude. Also, the interaction between the types of Geon primitives for a given semantic may not be optimal. For example, dependency may not be clearly revealed if a cubic Geon is

dependent on a conic Geon which would place the cube at the apex of the cone. Such a representation may lead to the understanding of an unstable system, which is a different semantic. Much more needs to be done to investigate the types of Geon primitives that are best suited for the given semantics.

A finding that reoccurs throughout the first two sets of experiments was that there were no differences in interpretation of the Geon-based notation between UML novices and experts, presumably because the task did not directly involve modeling. In the last experiment, subjects' level of experience played a significant factor in finding the diagram that correctly matched the problem descriptions. However, in the case of both, experts and novices, the notation of Geon diagrams was better recalled and better matched than that of UML. These suggest that the new notation can possibly help experts and non-experts interact. In many projects, interaction between the client, manager, and programmer is essential for the proper development of the system, so a diagramming system that is more easily accessible to non-experts could be useful. Off course we cannot expect the abandonment of UML notations in favor of Geon notations. UML provides the most highly evolved graphical modeling tools available today. However, we may hope that the development of new modeling applications may start to use representations that take advantage of diagrams made with 3D shape primitives.

Chapter 8 - Conclusion

The primary contribution of this thesis has been to demonstrate that applying structural object recognition theories to the design of node-link diagrams can facilitate understanding, memorability and learning of the information being communicated. The benefit of using perceptual primitives and perceptual semantics is further apparent when users without the technical knowledge of a particular application domain can grasp the information structures without significant difficulty. These results can considerably impact the manner in which visualization systems are being created for domain specific analysts as well as users in general.

According to the taxonomy of structured information types presented in chapter 2, diagrams can fulfill several roles. They are used as tools for communicating, identifying

structures, problem solving, creative thinking, and conceptualizing. This thesis has primarily investigated the application of theories of perception to diagrams as a medium of communication. Previously, node-link diagrams have exploited some of the Gestalt principles. However, recent theories of object perception suggest that we process visual input via a decomposition process starting from the 2D silhouette structure of the object. After this, shading information helps in revealing the structure of the objects and the primitive elements, Geons (such as cones, cylinders, wedges), which compose it. The structure, together with the Geons, provides the description that leads to identifying the object. This thesis has shown that the application of these theories, in particular the theory of recognition-by-components [Biederman87], can benefit the development of new diagramming standards for structured data in the form of entity-relationships.

8.1 The Geon Diagram Model

If cylinders and cones are indeed visual primitives, then we can construct diagrams out of Geon-like objects, and such diagrams should be easy to interpret. The Geons and their connections could become metaphors in the diagrams. This reasoning led to what we refer to as the Geon Diagram. The Geon Diagram model consists of a set of rules that facilitates the construction of node-link diagrams using the theory of recognition-by-components. What follows is the entire description of the model, which was provided partially in chapters 4 and 7. Five rules relate to the use of Geons as 3D primitives in the diagrams, and three additional rules relate to the layout of the Geon structure.

G1: Major entities of a system should be presented using simple 3D shape primitives (Geons).

G2: The links between entities can be represented by the connections between Geons. Thus the Geon structural skeleton represents the data structure.

G3: Minor sub-components are represented as Geon appendices – small Geon components attached to larger Geons.

G4: Secondary attributes of entities and relationships are represented by Geon color and texture and by symbols mapped onto the surfaces of Geons.

G5: Geons should be shaded to make their 3D shape clearly visible.

According to the theory of recognition-by-components, object identification heavily relies on clearly perceiving the silhouette. For this reason, even though Geons are 3D shape primitives, a good 2D layout will also be important in determining how easily the structure can be identified. Thus the following layout rules are added.

L1: All Geons should be visible from the chosen viewpoint.

L2: The Geon diagram should be laid out predominantly in the plane orthogonal to the view direction.

L3: Junctions between Geons should be made clearly visible.

These two sets of rules show that Geon diagrams are a kind of a hybrid 3D and 2D system - commonly referred as 2-1/2D. They incorporate 3D information in the shapes used to represent information, but it is recommended that they be laid out, as much as possible, in the 2D plane orthogonal to the line of sight.

In addition to the two sets of rules above, nine additional rules for representing certain diagram semantics define the Geon diagram. These rules include the mapping of semantics found in entity-relationship structures to data elements. These are:

SM1: Use Shape to represent Generality - Geons with same structural geometrical composition (or shape) can be used to denote objects of the same kind.

SM2: Use on-top-of to represent Dependency - if Geon A is on top of Geon B this suggests that Geon A is supported by Geon B. In addition gravity determines that structures are perceived as either being stable or unstable.

SM3: Use Enclosure to representation Aggregation - shows that Geon A is contained within Geon B. Syntactically this can be shown as an internal component attached to the same primitive Geon on the outside.

SM4: Use multiple links to represent Multiplicity - a series of attachments can best denote multiple associations between two entities.

SM5: Use Thickness to represent Strength of connection - using a thicker connection as opposed to a thinner one can denote a stronger relationship between two entities.

SM6: Use linear arrangement of Geons to show sequence - Geons arranged in a line become a metaphor for a chain of operations or some other linear structure.

SM7: Use symmetrical arrangement of Geons to show Symmetry - some information structures have symmetry; symmetrical arrangement of Geons should be used to show this.

SM8: Place Geons centrally to show higher relevance - if a component is of central importance to a structure this can be illustrated through its position and by the location of the interconnections.

SM9: Use Size to represent Greater Magnitude or Quantity - understanding of superiority in areas such as finances, economics, and politics can be mapped to larger components.

These rules provide the framework for drawing node-link diagrams that can be visually parsed more effectively and understood more rapidly in comparison to box and line diagrams.

8.2 Review of Experimental Findings

The experiments were designed to evaluate the Geon Diagram Model. In the first part of the investigation, structural diagrams with Geon primitives were compared to box and line diagrams of the type used in most entity-relationship models. The results of the first experiment showed that visual parsing of sub-structures can be more efficient when the nodes and links are composed of Geon primitives. In the second experiment, the recall of Geon diagrams was compared to that of box and line diagrams. Subjects were capable of recalling the Geon diagrams with half as many errors. In both these experiments the Geon diagrams were created with surface attributes of color and texture which mapped directly to labels in the UML diagrams. The third experiment tested the recall of Geon and UML diagrams when these were void of color/texture and labels respectively. The results from this part of the research suggest that Geon diagrams can facilitate visual comprehension

and recall of node-link diagrams in comparison to UML diagrams, which consist of a box and line notation.

The first three experiments do not clearly identify whether the contributing factor for the improved efficiency with the Geon diagrams is a result of the Geons. There were many factors that could influence subjects' performance, such as the general layout of the diagrams. The second part of the experiment sought to determine whether the Geon primitives themselves contributed to the increased performance in the tasks developed for the first three experiments. This resulted in comparing diagrams with shapes with the same outline, but one containing shaded primitive Geons and the other composed of 2D outlined-silhouette shapes in the form of Geons. The task in experiment 4 was similar to that of experiment 1 where subjects were asked to identify sub-structures within the diagrams. Subjects were more accurate and faster in identifying the diagram sub-structures with the Geon diagrams. Neither type of diagram consisted of surface properties or labels. Experiment 5 consisted of a recall test and the results showed that subjects were more accurate when recalling diagrams created with Geon components. These two experiments confirmed that shaded diagrams elements giving the impression of a 2-1/2D representation facilitated sub-structure parsing and recall of diagram structures.

The first two parts of the thesis focused on examining diagram structures with perceptual primitives. The last three experiments were designed to investigate a mapping of perceptual semantics into diagram semantics. Biederman's theory of recognition-by-

components [Biederman87] suggests that our perceptual system uses a set of built-in perceptual semantics in order to achieve recognition of an object. This phase of the research applied these rules to semantics commonly used by the class of entity-relationship models, specifically the semantics of UML class diagrams. The results of experiment 6 provided a set of visual representations for each of the semantics outlined for the research. These visual constructs were used for building diagrams that were validated and compared to current representations of these semantics in experiment 7. Subjects' interpretation was tested and the results showed that on average subjects performed five times better using the Geon notation. Finally, experiment 8 compared the learnability and long-term memorability of the Geon notation to UML notations. Experts and novices performed over twice as well with the Geon notations. These three experiments confirmed that the application of perceptual semantics to diagram structures can improve memorability, learnability and comprehension of diagrams semantics without any rigorous training.

8.3 Major Contributions of the Thesis

Several major contributions are embodied in this thesis and are described in this section. The first is the analysis and review of previous research leading to the Geon diagram. This review consisted of analyzing the major theories of structural object recognition, which have been incorporated at various stages in the thesis.

The second major contribution is the set of rules that define the Geon diagram (see section 8.1). The Geon diagram drawing rules, the layout rules and the semantic rules can be applied in general for modeling entity relationship information. Structures that have the semantics of generalization or dependency, basic to UML can be revealed in a more intuitive manner. These visual structures can be comprehended quicker and can be particularly developed for users without the technical background that is required in comprehending the box and line based notations that are currently widely used.

The third major contribution is the experimental method of sub-structure identification (termed the sub-structure identification task) that was constructed and employed for the visual parsing experiments. This method provides a new approach for exploring the visual comprehension of node-link diagrams. The sub-structure identification task requires that subjects maintain a mental model of the entities and their relationships. Diagrams facilitate the role of searching by localizing elements [LS87]; therefore the sub-structure identification task can assist in investigating the parsing ability of users with respect to new diagramming conventions.

A fourth contribution is the experimental validation of the Geon diagram rules. The validation was performed in two stages each consisting of several experiments. The first stage evaluated the Geon diagram structure, while the second stage evaluated the semantics. The results show that constructing diagrams using the Geon diagram model can lead to effective diagrams.

A fifth contribution is the three-step process used in the last stage of the thesis for deriving a new syntax for representing entity relationship semantics. At the *constructive* stage, based on the given set of semantics, a series of notations were developed, some of which were based on perceptual principles. The *evaluation* phase obtained subjects preference for each of the visual notations. Finally the *validation* phase tested the users comprehension of the diagrams constructed using the semantic rules developed in the previous phase. Although such a process requires some rigor follow, we believe it is critical and could result in determining whether a notation is better suited to its audience. This process is particularly important in the Information Sciences, where diagramming notations have been arbitrarily developed. Researchers formulating new diagramming standards can use a stepwise procedure similar to this.

8.4 Further Investigations

Future research in this area can follow two broad directions. One would consist of improving the usability of the Geon diagramming toolkit and making it a more practical tool. The other area would consist of investigating theoretical issues that have emerged from the research in this thesis.

8.4.1 Usability and Practicality of the Geon Diagram Toolkit

Further implementation work on the Geon toolkit can follow two directions: making the toolkit's current features more usable and adding features driven by future research

requirements. An exhaustive list of features can be implemented if the toolkit is to evolve into a generic environment for research prototypes in visualization.

Geon diagrams cannot be easily drawn using pen and paper methods, and their construction therefore requires a usable and flexible toolkit. The toolkit needs to provide the fluid interaction that is expected from most drawing packages. For example, the toolkit currently does not support constrained deformation. To assist the researcher by making the toolkit more usable, the Geons should be interactively deformable. This deformation should be constrained so that the natural attributes of the Geon do not get altered so much that a different Geon may be perceived.

Another feature that would make the toolkit more usable would be to introduce connections between elements using a snap-and-drag technique. Connecting Geons with a snap-and-drag effect can constrain how Geons should be connected. Connection points were described earlier; there should be supporting research describing which connections facilitate ease of understanding or facilitate the task of drawing diagram that are more legible according to a set of semantic rules.

With considering applications, it would be interesting to integrate the visual structures developed in this research into the tools currently available for the class of entity-relationship diagrams. This could be achieved by implementing what we call a Geon semantic viewer. The purpose of the viewer would be to translate syntax and semantics used by some of the more popular modeling tools such as Rational RoseTM and to present

it using the Geon notation. This would allow existent tool developers to leverage the capability of their modeling applications as a medium of communication between members of software development teams.

A drawback in providing Geon drawings is access to high quality printing that can clearly show Geon attributes such as texture or transparency. The cost of color printing still remains high but is rapidly falling.

8.4.2 Future Theoretical Investigations

A number of formal studies can be carried out to investigate other aspects of Geon diagrams. For example, a study could be done to compare the visual parsing of diagram structures with surface properties of color or texture to those with Geon shaped primitives. This investigation could apply the sub-structure identification task developed in this thesis, to the study. This experiment would determine whether visual search is better facilitated using shaped primitives in comparison to colored and textured 3D-shaped primitives.

Geon surface attributes have not been explored in this research. The research can thus be extended to investigating the types of surface properties that can convey meaning intuitively. Textures and colors can be possibly mapped to semantics. A sufficient body of literature exists in perception that may suggest how these attributes could be mapped unobtrusively to the display. This would facilitate the display of certain types of semantic

properties on the entities in the diagram. Another study related to surface properties could investigate the types of textures to be used for mapping information into data elements. It could also investigate the threshold of subtle textures or colors (for example pastel vs. non-pastel colors) for avoiding conflict with text.

From a theoretical standpoint it would be interesting to investigate the effect of various shading models for the Geon diagrams. This could consist of similar experiments to those of experiments 1 and 4. The purpose of such an investigation would be to test the effectiveness of diagram comprehension under different shading conditions. A study by Sun and Perona [SP96] suggest that lighting from above can result in faster searches than lighting from below. Psychophysical experiments by Hanazawa and Komatsu [HK01] have also shown that cells respond best to lighting effects when directed from above. Further research can elaborate on the lighting conditions that produce the best results for comprehending structural spatial relations and could be designed to answer questions such as how minimal the shading should be before sub-structure identification and recall of structures are impeded.

A question that has often appeared with respect to the nature of Geon diagrams is of its scalability. If nodes and links are represented by 2-1/2D entities, then complex diagrams with over a hundred nodes will be challenging to represent. Simple box and line diagrams do not occupy as much space on the screen. It would be interesting to investigate methods for making for compact diagrams with 2-1/2D notations.

Finally, this thesis has only begun to explore the potential for creating 3D diagrams. It presents a case for improving diagram understandability and communication. However much more work can be done to investigate the mapping of perceptual principles to diagrams for conceptualizing ideas, creative thinking, and problem solving. Each of these domains have semantics that can be matched to perceptual processes which can result in diagrams that are generally better understood.

A recent conversation with Dr. J. Rambaugh ¹ on the matter of notations in UML revealed that box and line notations were adopted by OMG primarily for the sake of simplicity. For example, the notation for representing a class that was considered during the initial stages of UML was a cloud, a notation that had been used by Booch in his model of class notations [Booch94]. Such a notation was discarded based on its complex geometric features. The second motivation for adopting a simple notation was due to the processing power of computers during the early nineties when such notations were being considered. However, Rambaugh recognized the need for notations that are more understandable for users. The graphical processing capabilities available today make it possible to display Geon diagrams on almost every desktop. Moreover, notations were primarily created with the programmers and systems architects in mind and therefore were not aimed at the general public. Today, clients who are recipients of custom made software contribute significantly to the systems development process and therefore

¹ Dr. James Rambaugh is one of the founders of the Unified Modeling Language and was the primary person supporting the use of a simple notation. This conversation with Dr. Rambaugh took place during "UML 2001" the 4th International Conference on the Unified Modeling Language in Toronto, October 1-5, 2001.

notations that are better suited to this audience are necessary for the successful deployment of a system.

Perceptual Syntax for Diagram Understanding

Visual Interfaces Laboratory Experiment

Stage 1 - Training

Thank you for participating!



Introduction to Experiment

- Compare understandability of two types of diagramming systems - UML and GEON
- Stage 1 (training) - Describe notation of both systems
- Stage 2 (experiment) - Match diagram to description



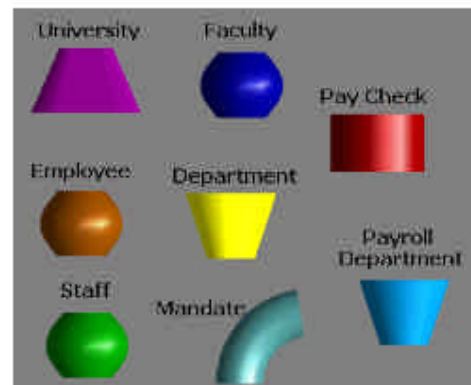
Object Oriented Modeling

- Process used for building software systems
- Take a problem description and break it into objects
- Represent the relationships between objects
- OOM Tools - UML is a graphical language



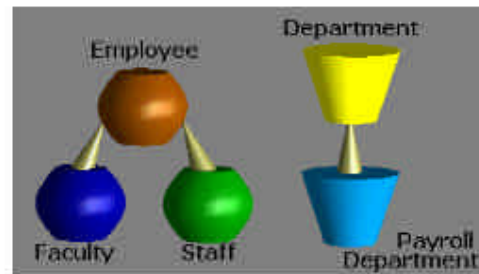
Representing Objects

- An object is an abstract unit that contains data and operations on the data



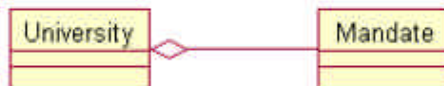
Inheritance (is-a)

- Central idea in OO Programming
- Relationship in which one object inherits properties of another



Composition (has-a)

- Objects are used as building blocks
- An object can be contained within other objects



Dependency

- A dependency exists if changes to one object causes changes to another
- E.g. Employee depends on Pay Check



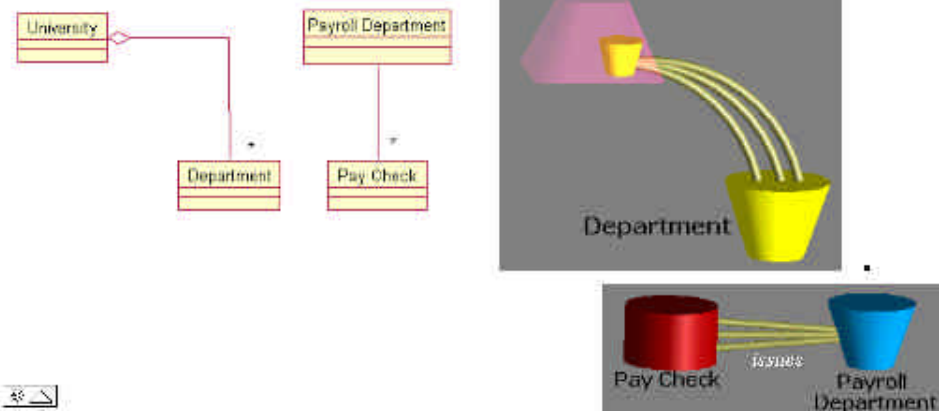
Association

- A common relationship between two objects
- E.g. An employee works for a university

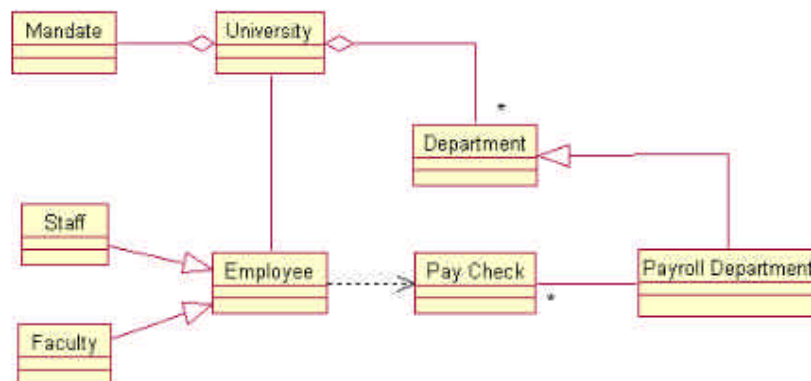


Multiplicity

- Attribute of relationships to denote multiple instances

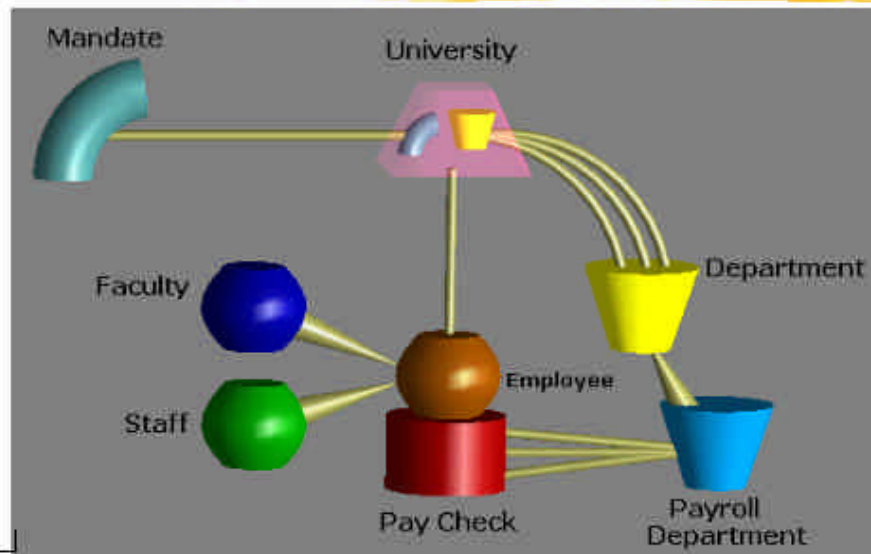


University Example (UML)

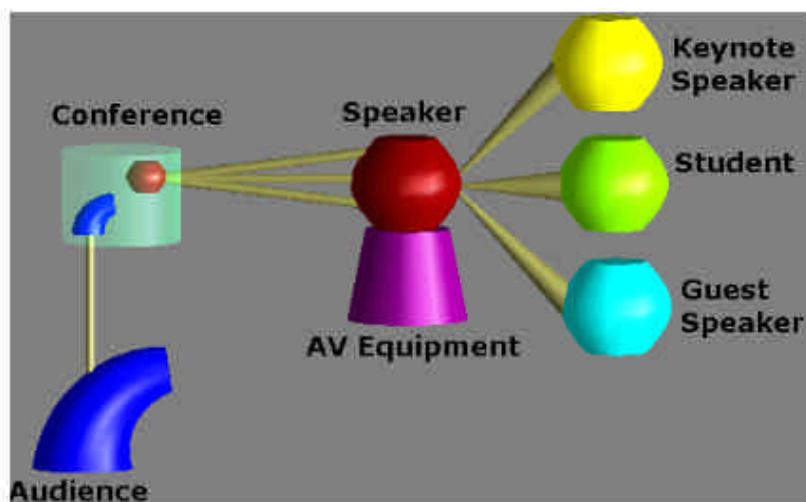


UML Diagram modeling certain aspects of a university

University Example (GEON)

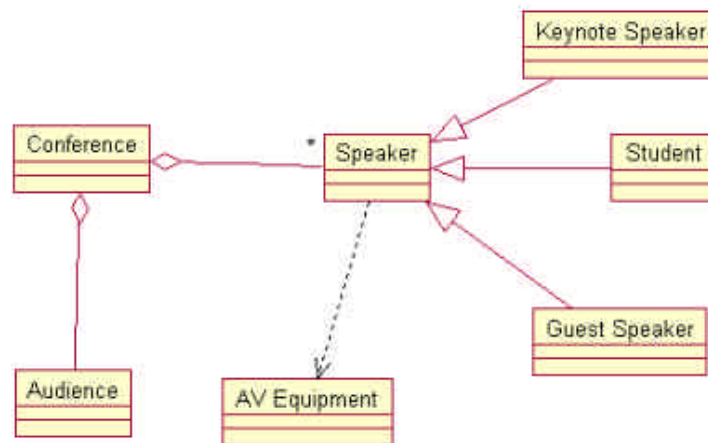


Conference Example (GEON)



Geon Diagram modeling a conference

Conference Example (UML)



UML Diagram modeling a conference

References

- [AH98] K. Andrews and H. Heidegger. *Information Slices: Visualizing and Exploring Large Hierarchies using Cascading, Semi-Circular Discs*, In Late Breaking Hot Topic Paper, IEEE Symposium on Information Visualization (InfoVis'98), Research Triangle Park, North Carolina, pages 9-12, 1998.
- [Albers87] J. Albers. *Interaction of Color*, Yale University Press, New Haven, CT, US, 1987.
- [Andrews96] K. Andrews. *Browsing, Building, and Beholding Cyberspace: New Approaches to the Navigation, Construction, and Visualisation of Hypermedia on the Internet*, PhD thesis, Graz University of Technology, Austria, 1996.
- [Arnheim69] R. Arnheim. *Visual Thinking*, University of California Press, Berkeley, CA, 1969.
- [Arnston98] A. Arnston. *Graphic Design Basics 3rd Edition*, Harcourt Brace Jovanovich College Publishers, 1998.
- [Bachman69] C.W. Bachman. *Data Structure Diagrams*, Data Base, The Quarterly Newsletter of the Special Interest Group on Business Data Processing of the ACM, **1**(2): 4-10, 1969.
- [BCD75] G.H. Bower, M.B. Carlin, and A. Duek. *Comprehension and memory for pictures*, Memory and Cognition, **3**(2): 216-220, 1975.
- [BCS96] A. Buja, D. Cook, and D. F. Swayne. *Interactive High--Dimensional Data Visualization*, Journal of Computational and Graphical Statistics, **5**(1): 78-99, January 1996.
- [BD80a] J.B. Brooke and K.D. Duncan. *An experimental study of flowcharts as an aid to identification of procedural faults*, Ergonomics, **23**(4): 387-399, 1980.
- [BD80b] J.B. Brooke and K.D. Duncan. *Experimental studies of flowchart use at different stages of program debugging*, Ergonomics, **23**(11): 1057-1091, 1980.
- [Behrens84] R. Behrens. *Design in the visual arts*, Prentice-Hall, Englewood Cliffs, NJ, 1984.
- [BF93] C. Beshers and S. Feiner. *AutoVisual: Rule-Based Design of Interactive Multivariate Visualisations*, IEEE Computer Graphics and Applications, **13**(4): 41-49, July 1993.

- [Bertin81] J. Bertin. *Graphics and graphic information processing*, Walter de Gruyter, Berlin, NY, 1981.
- [Bertin83] J. Bertin. *Semiology of Graphics*, The University of Wisconsin Press, Madison, 1983. Translated by William J. Berg.
- [BETT94] G. DiBattista, P. Eades, R. Tamassia, and I. Tollis. *Annotated bibliography on graph drawing algorithms*, Computational Geometry: Theory and Applications, **4**:235-282, 1994.
- [BETT99] G. DiBattista, P. Eades, R. Tamassia, and I. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice-Hall, Englewood Cliffs, NJ, 1999.
- [BEW95] R.A. Becker, S. G. Eick, and A. R. Wilks. *Visualizing Network Data*, IEEE Transactions on Visualization and Computer Graphics, **1**(1):16-28, March 1995.
- [BGG97] V. Bruce, P. R. Green, and M. A. Georgeson. *Visual Perception: Physiology, Psychology, and Ecology*, Psychology Press, East Sussex, UK, 1997.
- [BG93] I. Biederman and P.C. Gerhardstein. *Recognizing depth-rotated objects: Evidence and conditions for three dimensional viewpoint invariance*, Journal of Experimental Psychology: Human Perception and Performance, **19**:1162-1182, 1993.
- [BG95] I. Biederman and P. C. Gehardstein. *Viewpoint-Dependent Mechanisms in Visual Object Recognition: Reply to Tarr and Bulthoff (1995)*, Journal of Experimental Psychology, **21**(6):1506-1514, 1995.
- [BH94] V. Bruce and W. G. Humphreys. *Object and Face Recognition*, Lawrence Erlbaum Associates, East Sussex, UK, 1994.
- [BI96] M. Bar and I. Biederman, One-Shot Viewpoint Invariance in Matching Novel Objects, http://rana.usc.edu:8376/~bar/One_Shot.html.
- [Biederman87] I. Biederman. *Recognition-by-Components: A Theory of Human Image Understanding*, Psychological Review, **94**(2):115-147, 1987
- [BL93] M. I. Bauer and P. N. Johnson-Laird. *How Diagrams Can Improve Reasoning*, Psychological Science, **4**(6):372-378, 1993.
- [Booch94] G. Booch. *Object-Oriented Analysis and Design with Applications*, The Benjamin/Cummings Publishing Company Inc., Redwood City, California, 1994.
- [BPV96] L. Beaudoin, M.-A. Parent, and L. C. Vroomen. *Cheops: A Compact Explorer for Complex Hierarchies*, Proceedings of Visualization'96, IEEE Computer Society, San Francisco, CA, pages 87-92, 1996.

[Braun93] J. Braun. *Shape-from-shading is independent of visual attention and may be a "texton"*, Spatial Vision, **7**:311-322, 1993.

[BSPBR93] E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. D. DeRose. *Toolglass and Magic Lenses: The See-Through Interface*, Proceedings of SIGGRAPH '93, Computer Graphics Annual Conference Series, Anaheim, CA, pages 73-80, August, 1993.

[BWK00] M. Q. W. Baldonado, A. Woodruff, and A. Kuchinsky. *Guidelines for Using Multiple Views in Information Visualization*, Advanced Visual Interfaces, Palermo, Italy, pages 110-119, May, 2000.

[CBTTB92] R. F. Cohen, G. Di Battista, R. Tamassia, I. G. Tollis, and P. Bertolazzi. *A framework for dynamic graph drawing*, Proceedings of 8th Annual ACM Symposium on Computational Geometry, pages 261-270, 1992.

[CCF95] S. T. Carpendale, D. J. Cowperthwaite, F. D. Fracchia. *3-Dimensional Pliable Surfaces: For the Effective Presentation of Visual Information*, Proceedings of the ACM Symposium on User Interface Software and Technology, ACM Press, Pittsburgh, PA, pages 217-226, 1995.

[CEH96] K. C. Cox, S. G. Eick, and T. He. *3D Geographic Network Displays*, ACM Sigmod Record, **24**(4), December 1996.

[Chen76] P. P. Chen. *The Entity-relationship Model - Towards a Unified View of Data*, ACM Transactions on Database Systems, **1**(1):9-36, March 1976.

[Cheng96] P.C.H. Cheng. *Scientific discovery with law encoding diagrams*, Creativity Research Journal, **9**(2): 145-162, 1996.

[Chernoff71] H. Chernoff. *The Use of Faces to Represent Points in n-Dimensional Space Graphically*. Technical Report No 71, Department of Statistics, Stanford University, Stanford, CA, 1971.

[Cleveland85] W. S. Cleveland. *The Elements of Graphing Data*, Wadsworth Press, Monterey, CA, pages 1-323, 1985.

[CMN83] S. Card, T. Moran, A. Newell. *The Psychology Of Human-Computer Interaction*, Erlbaum, Hillsdale, NJ, 1983.

[CMS99] S.K. Card, J.D. Mackinlay, and B. Shneiderman. *Readings in Information Visualization - Using Vision to Think*, Morgan Kauffman Publishers, San Francisco CA, 1999.

- [Codd70] E.F. Codd. *A Relational Model of Data for Large Shared Data Banks*, Communications of the ACM, **13**(6):377-387, 1970.
- [Coltheart99] V. Coltheart. *Fleeting Memories: Cognition Of Brief Visual Stimuli*, MIT Press, Cambridge, MA, 1999. [page 63]
- [CPMPGC98] E. Chi, J. Pitkow, J. Mackinlay, P. Pirolli, R. Gossweiler, and S. Card *Visualizing the evolution of web ecologies*, Conference on Human Factors in Computing Systems (CHI '98), Los Angeles, CA, pages 400-407, 1998.
- [CS84] L. A. Cooper and R. N. Shepard. *Turning something over in the mind*, Scientific American, **251**(6):106-117, 1984.
- [DDMR90] S. J. DeRose, D. G. Durand, E. Mylonas, and A. H. Renear. *What is Text, Really?*, Journal of Computing in Higher Education, **1**(2):3-26, 1990.
- [Douglass98] B.P. Douglass, *Real-Time UML*, Addison-Wesley, 1998.
- [Edelman95] S. Edelman. *Representation of similarity in 3D object discrimination*, Neural Computation, **7**:407-422, 1995.
- [Eick97] S.G. Eick. *Visualization and Interaction Techniques*, In CHI97 Tutorial notes on Information Visualization. ACM SIGCHI, March 1997.
- [ER90a] J. T. Enns and R. A. Rensink. *Influence of Scene-Based Properties on Visual Search*, Science **247**:721-723, 1990.
- [ER90b] J. T. Enns and R. A. Rensink. *Sensitivity to Three-Dimensional Orientation in Visual Search*, Psychology Science, **1**(5):323-326, 1990.
- [Erickson91] T. Erickson. *Working with Interface Metaphors*, in Laurel B. "The Art of Human Computer Interface Design", Addison Wesley, pages 65-73, 1991.
- [ERSPKR99] D. S. Ebert, R. M. Rohrer, C. D. Shaw, P. Panda, J. M. Kukla, and D. A. Roberts. *Procedural Shape Generation for Multidimensional Data Visualization*, Data Visualization'99, Proceedings of the Joint Eurographics-IEEE TCVG Symposium on Visualization, Springer-Verlag, New York, pages 3-12, May 1999.
- [ES90] P. Eades and K. Sugiyama. *How to Draw a Directed Graph*, Journal of Information Processing, **13**(4):424-434, 1990.
- [ESS92] S.G. Eick, J.L. Steffen, and E.E Sumner Jr.. *Seesoft - a tool for visualizing line oriented software statistics*, IEEE Transactions on Software Engineering, IEEE Press, **11**(18):957-968, 1992.

- [FAP90] W. T. Freeman, E. H. Adelson, and A. P. Pentland. *Shape-from-shading Analysis with Shadelets and Bumplets*, Association for Research in Vision and Ophthalmology, (ARVO), **31**:410, 1990.
- [FF95] E. T. Freeman and S. J. Fertig. *Lifestreams: Organizing your electronic life*, In AAAI Fall Symposium: AI Applications in Knowledge Navigation and Retrieval, Cambridge, MA, November 1995.
- [FNF91] K. D. Forbus, P. Nielsen, and B. Faltings. *Qualitative spatial reasoning: The clock project*, Artificial Intelligence, **51**:417-471, 1991.
- [Fowler96] M. Fowler. *UML Distilled: Applying the Standard Object Modeling Language*, Addison-Wesley, 1996.
- [FPF88] K. Fairchild, S. Poltrock and G. Furnas. *SemNet: Three-dimensional graphic representations of large knowledge bases*, in "Cognitive Science and its Applications for Human-Computer Interaction", R. Guindon, ed., Lawrence Erlbaum Associates, Publishers, pages 201-233, 1988.
- [Funt80] B. V. Funt. *Problem-Solving with Diagrammatic Representations*, Artificial Intelligence **13**:201-230, 1980.
- [Furnas86] G. W. Furnas. *Generalized fisheye views*, Human Factors in Computing Systems CHI '86 Conference Proceedings, Boston, pages 16-23, April 13-17, 1986
- [GE95] Gershon, N. and Eick, S. G., 1995, Visualizaton's new tack: Making sense of information, IEEE Spectrum, Nov, 38-56.
- [GE97] N. Gershon and S.G. Eick. *Information Visualizaton*, IEEE Computer Graphics and Applications **17**(4):29-31, 1997.
- [GL95] G. Grinstein and H. Levkowitz. *Perceptual Issues in Visualisation*, Springer-Verlag, Berlin, Germany, 1995.
- [GNC95] J.I. Glasgow, N. Hari Narayanan, and B. Chandrasekaran. *Diagrammatic Reasoning: Cognitive and Computational Perspectives*, MIT Press, Cambridge, MA, 1995.
- [Goldstine72] H.H. Goldstine. *The Computer from Pascal to von Neumann*, Princeton University Press, Princeton, NJ, 1972.
- [Gombrich65] E. H. Gombrich. *Art and Illusion: A study in the psychology of pictorial representation*, Princeton University Press, Princeton, NJ, 1965.
- [GP92] J.I. Glasgow and D. Papadias. *Computational Imagery*, Cognitive Science, **17**(3):355-394, 1992.

- [GS86] C. Gane and T. Sarson. *Structured Systems Analysis: Tools and Techniques*, Prentice-Hall, Englewood Cliffs, NJ, 1979.
- [GW96] Grinstein, G. and Ward, M. O., Introduction to Visualization, Vis '96 Tutorial #2.
- [Halverston92] J. Halverston. *The first pictures, perceptual foundations of Paleolithic art*, Perception, **21**:389-404, 1992.
- [Harel88] D. Harel. *On Visual Formalisms*, Communications of the ACM **31**(5):514-530, May 1988.
- [HDWB95] B. J. Hendley, N. S. Drew, A. M. Wood, and R. Beale. *Narcissus: Visualising Information*, In Proceedings of InfoVis'95, Atlanta, GA, IEEE Computer Society, pages 90-96, 1995.
- [HE98] C. G. Healey and J. T. Enns, *Building Perceptual Textures to Visualize Multidimensional Datasets*, In Proceedings of IEEE Visualization '98, North Carolina, pages 111-118, 1998.
- [HHN00] S. Havre, B. Hetzler, and L. Nowell. *ThemeRiver: Visualizing Theme Changes over Time*, In Proceedings of InfoVis 2000, Salt Lake City, UTAH, 2000.
- [Hibbard99] B. Hibbard, *Top Ten Visualization Problems*, ACM SIGGRAPH, **33**(2), May 1999.
- [HK01] A. Hanazawa and H. Komatsu. *Influence of the Direction of Elemental Luminance Gradients on the Responses of V4 Cells to Textured Surfaces*, The Journal of Neuroscience, **21**(12):4490-4497, June 2001.
- [HMM00] I. Herman, G. Melançon, M. S. Marshall, *Graph Visualization and Navigation in Information Visualization*, IEEE Transactions on Visualization and Computer Graphics, **6**(1):24-43, 2000.
- [Horstmann99] C.S. Horstmann, *Computing Concepts with Java 2 Essentials*, John Wiley & Sons, 1999.
- [Huff54] D. Huff. *How to Lie with Statistics*, New York, W. W. Norton & Company, 1954.
- [Intraub97] H. Intraub. *The representation of Visual Scenes*, Trends in the Cognitive Sciences, **1**:217-221, 1997.

- [JB96] L. Jin and D. C. Banks. *Visualizing a Tennis Match*, In Proceedings of the 1996 IEEE Symposium on Information Visualization (InfoVis '96), San Francisco, CA, pages 108-114.
- [JOC00] Y. Jiang, I. R. Olson, and M. M. Chun. *Organization of visual short-term memory*, Journal of Experimental Psychology: Learning, Memory, and Cognition, **26**:683-702, 2000.
- [JS91] B. Johnson and B. Shneiderman. *Tree-Maps: A Space-Filling Approach to the Visualization of Hierarchical Information Structures*, In Proceedings of 1991 IEEE Symposium on Visualization, IEEE Computer Society, San Diego, CA, pages 284-291, 1991.
- [KA00] J. Korn and J. Abello. *Visualizing Massive Multi-Digraphs*, In Proceedings of 2000 IEEE Symposium on Information Visualization (InfoVis2000), Salt Lake City, UTAH, pages 39-48, 2000.
- [Kanizsa79] G. Kanizsa. *Organization in vision: Essays on gestalt perception*, New York, Praeger, 1979.
- [Koffka35] K. Koffka. *Principles of Gestalt psychology 5th ed.*, Routledge & Kegan Paul Ltd., London, UK, 1935.
- [Kosslyn93] S. M. Kosslyn. *Elements of Graph Design*, Freeman, W. H. & Company, New York, NY, 1993.
- [Kosslyn94] S. M. Kosslyn. *Image and Brain: The Resolution of the Imagery Debate*, MIT Press, Cambridge, MA, 1994.
- [LH88] M. Livingstone and D. Hubel. *Segregation of form, colour, movement and depth: anatomy, physiology and perception*, Science, **240**:740-749, 1988.
- [LH92] H. Lefkowitz and G. T. Herman. *Color Scales for Image Data*, IEEE Computer Graphics and Applications, **12**(1):72-80, January 1992.
- [Lippe95] P. von der Lippe. *Perceptual Articulation of Musical Gestalts*, Univ. of Oslo, Norway, 1995.
- [LJ80] G. Lakoff and M. Johnson. *Metaphors We Live By*, University of Chicago Press, Chicago, IL, 1980.
- [LM96] J. M. Lee and J. MacLachian. *The effects of 3D imagery on managerial data interpretation*, MIS Quaterly, 257-269, September 1986.

- [Lowe94] R.K. Lowe. *Selectivity in diagrams: Reading beyond the lines*, Educational Psychology, **14**:467-491, 1994.
- [LR94] J. Lamping and R. Rao. *Laying out and Visualizing Large Trees Using a Hyperbolic Space*, In Proceedings of UIST 1994, ACM, Marina del Rey, CA, pages 13-14, 1994.
- [LS87] J. H. Larkin and H. A. Simon. *Why a diagram is (sometimes) worth ten thousand words*, Cognitive Science **11**(1): 65-99, 1987.
- [LV97] S. J. Luck and E.K. Vogel. *The capacity of visual working memory for features and conjunctions*, Nature, **390**:279-281, 1997.
- [MacEachren95] A. M. MacEachren. *How Maps Work, Representation, Visualization & Design*, Guilford Press, New York, NY, 1995.
- [MacKinlay86] J. Mackinlay. *Automating the Design of Graphical Presentations of Relational Information*, ACM Transactions on Graphics, **5**(2):110-141, April 1986.
- [Marr82] D. Marr. *Vision: A computational investigation into the human representation and processing of visual information*, Freeman, San Fransisco, CA, 1982.
- [Marshall99] Marshall C., *Enterprise Modeling with UML: Designing Successful Software through Business Analysis*, Addison-Wesley, 1999.
- [MN78] D. Marr and H.K. Nishihara. *Representation and recognition of the spatial organization of three-dimensional shapes*, Proceedings of the Royal Society of London, B, **200**:269-294, 1978.
- [Munzer97] T. Munzner. *H3: Laying Out Large Directed Graphs in 3D Hyperbolic Space*, In the Proceedings of the 1997 IEEE Symposium on Information Visualization, Phoenix, AZ, pages 2-10, October 20-21 1997.
- [NPMK97] D. A. Nation, C. Plaisant, G. Marchionini and A. Komlodi. *Visualizing websites using a hierarchical table of contents browser: WebTOC*, In Proceedings of the 3rd Conference on Human Factors and the Web, US WEST, Denver, CO, 1997.
- [Palmer94] S.E. Palmer and I. Rock. *Rethinking perceptual organization: the role of uniform connectedness*, Psychonomic Bulletin and Review **1**:29-55, 1994.
- [Petre95] M. Petre. *Why looking isn't always seeing: readership skills and graphical programming*, Communications of the ACM, **38**(6), pages 33-44, June 1995.
- [PH89] C. J. Price and G.W. Humphreys. *The effects of surface detail on object categorization and naming*, Quarterly Journal of Experimental Psychology, **41A**:797-828, 1989.

[PL92] A. G. Priest and R. O. Lindsay. *New light on novice-expert differences in problem solving*, British Journal of Psychology, 1992.

[Ploix96] D. Ploix. *Building program metaphors*, In M. Ireland (Ed.), Proceedings of the First Psychology of Programming Interest Group Postgraduate Student Workshop, pages 125-129, 1996.

[Piaget52] J. Piaget, *The Child's Conception of Number*, W. W. Norton & Company, Inc., NY, 1952.

[Ramachandran88] V. S. Ramachandran. *Perception of shape from shading*, Nature, **331**:163-166, 1988.

[RBPEL91] J. Rambaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen. *Object Oriented Modeling and Design*, Prentice Hall, 1991.

[RCM93] G. G. Robertson, S. K. Card, and J. D. Mackinlay. *Information Visualization Using 3D Interactive Animation*, Communications of the ACM, **36**(4):57-71, April 1993.

[RES99] R. Rohrer, D. Ebert, and J. Sibert. *A shape-based visual interface for text retrieval*, IEEE Computer Graphics and Applications, **19**(5):40-46, Sep./Oct. 1999.

[Resnikoff89] H. L. Resnikoff. *The illusion of reality*, Springer-Verlag, New York, NY, 1989.

[RL98] B. Rogowitz and L. Treinish. *Data Visualization: The End of the Rainbow*, IEEE Spectrum, **35**(12):52-59, December 1998.

[RLK92] B. Rogowitz, D. T. Ling, and W. A. Kellogg. *Task Dependence, Veridicality, and Pre-Attentive Vision: Taking Advantage of Perceptually-Rich Computer Environments*, In the Proceedings of the SPIE Symposium, 1666, Human Vision, Visual Processing and Digital Display III, pages 504-513, February 1992.

[RMC91a] G. G. Robertson, J. D. Mackinlay, and S. K. Card. *Cone trees: Animated 3D visualizations of hierarchical information*, ACM Conference on Human Factors in Computing Systems (CHI '91), ACM, pages 189-194. 1991.

[RMC91b] G. G. Robertson, J. D. Mackinlay, and S. K. Card. *Information Visualization Using 3D Interactive Animation*, In CHI'91 Video Proceedings, ACM, 1991.

[ROC97] R.A. Rensink, J.K. O'Regan, and J.J. Clark. *To See or Not to See: The Need for Attention to Perceive Changes in Scenes*, Psychological Science, **8**:368-373, 1997.

[Rose99] S. Rose. *The sunflower visual metaphor, a new paradigm for dimensional compression*, In Proceedings of IEEE Information Visualization'99, October 1999.

[RT96] B. E. Rogowitz and L. A. Treinish. *How Not to Lie with Visualization*, Computers In Physics, **10**(3):268-273, May/June 1996.
<http://www.research.ibm.com/dx/proceedings/pravda/truevis.htm>.

[SA83] R. Spence and M. Apperley. *The Bi-focal Display*, Video, Imperial College Television Studio, London, 1983.

[SAAF73] M. E. Senko, E. B. Altman, M. M. Astrahan, and P. L. Fehder. *Data Structures and Accessing in Data-Base Systems*, Evolution of Information Systems, IBM Systems Journal **12**(1):30-44, 1973.

[SB94] M. Sarkar and M. H. Brown. *Graphical fisheye views*, Communications of the ACM **37**(12):73-84, December 1994.

[SBK99] O. Saal, E. Blake, and A. Krzesinski. *Visualisation of ATM Network Connectivity and Topology*, SATNAC99 Proceedings, University of Cape Town Press, 1999.

[SC93] D.L. Schacter and L.A. Cooper. *Implicit and explicit memory for novel visual objects: Structure and function*, Journal of Experimental Psychology: Learning, Memory, & Cognition, **19**:988-1003, 1993.

[Scanlan89] D.A. Scanlan. *Structured flowcharts out perform pseudocode: an experimental comparison*, IEEE Software, **6**(5):28-36, September 1989.

[Shneiderman98] B. Shneiderman. *Designing the User Interface 3rd ed.*, Addison-Wesley, Reading, MA, 1998.

[SHBER99] C. D. Shaw, J. A. Hall, C. Blahut, D. S. Ebert, and D. A. Roberts. *Using Shape to Visualize Multivariate Data*, Workshop on New Paradigms in Information Visualization and Manipulation, pages 17-20, 1999.

[Skupin00] A. Skupin. *From Metaphor to Method: Cartographic Perspectives on Information Visualization*, In the Proceedings of IEEE Symposium on Information Visualization 2000 (InfoVis2000), Salt Lake City, UTAH, 2000.

[SmartMoney] SmartMoney.com

[SMMH77] B. Shneiderman, M. Richard, D. McKay, and P. Heller. *Experimental Investigations of the Utility of Flowcharts in Programming*, Communications of the ACM, **20**(6):373-381, June 1977.

[SP96] J.Y. Sun and P. Perona. *Preattentive perception of elementary three-dimensional shapes*, Vision Research, **36**(16):2515-2529, August 1996.

- [Spence00] R. Spence. *Information Visualization*, Addison Wesley, 2000.
- [SR96] M. Scaife and Y. Rogers. *External cognition: how do graphical representation work?*, International Journal of Human-Computer Studies, **45**:185-213, 1996.
- [Stanislas98] D. Stanislas. *The Number Sense: How the Mind Creates Mathematics*, Oxford University Press, NY, 1998.
- [TG88] A. Treisman and S. Gormican. *Feature analysis in early vision: evidence from search asymmetries*, Psychological Review, **95**:15-48, 1988.
- [Triesman80] A. Triesman. *Preattentive processing in vision*, Computer Vision, Graphics and Image Processing, **31**:156-177, 1980.
- [Treisman86] A. Treisman. *Properties, parts and objects*, In: Handbook of Perception, Volume II, pages 35.1-35.70, 1986.
- [Tuft83] E. Tufte. *The Visual Display of Quantitative Information*, Graphics Press, Cheshire, CT, 1983.
- [Tuft90] E. Tufte. *Envisioning Information*, Graphics Press, Cheshire, CT, 1990.
- [Tuft97] E. Tufte. *Visual Explanations*, Graphics Press, Cheshire, CT, 1997.
- [Ullman89] S. Ullman. *Aligning pictorial descriptions: An approach to object recognition*, Cognition, **32**:193-254, 1989.
- [UML97] UML Notation Guide, Version 1.1, Rational Software Corporation, 1997.
- [Ware99] C. Ware. *Information Visualization: Perception for Design*, Morgan Kaufman, December 1999.
- [WD83] C. J. Watson and R. W. Driver. *The influence of computer graphics on the recall of information*, MIS Quaterly, pages 45-53, March.1983.
- [WF96] C. Ware and G. Franck. *Evaluating stereo and motion cues for visualizing information nets in three dimensions*, ACM Transactions on Graphics, **9**(2):226-232, 1996.
- [WF94] C. Ware and G. Franck. *Representing Nodes and Arcs in 3D Networks*, In IEEE Conference on Visual Languages, pages 189-190, October 1994.
- [Wilkinson99] L. Wilkinson. *The Grammar of Graphics (Statistics and Computing)*, Springer Verlag, 1999.

- [Williams95] P. Williams. *Effects of Picture-Plans Orientation, Size, and Color Transformations on Object Decision Priming*, 3rd annual Pre-Psychonomis Workshop on Object Perception and Memory, Los Angeles, CA, November 1995.
- [Wills97] G. J. Wills. *NicheWorks - Interactive Visualization of Very Large Graphs*, Journal of Computational and Graphical Statistics, 8(2):190-212, June 1999.
- [WK92] C. Ware and W. Knight. *Orderable dimensions of visual texture for data display: Orientation, size, and contrast*, In Proceedings of SIGCHI '92, Monterey, CA, pages 203-209, 1992.
- [WK95] C. Ware and W. Knight. Using visual texture for information display, ACM Transactions on Graphics, 14(1):3-20, January 1995.
- [WNDS97] M. Woo, J. Neider, T. Davis, and D. Shreiner. *OpenGL® Programming Guide: The Official Guide to Learning OpenGL Version 1.2*, 1998.
- [WTPLPS95] J.A. Wise, J. J. Thomas, K. Pennock, D. Lantrip, M. Pottier and A.Schur. *Visualizing the non-visual: Spatial analysis and interaction with information from text documents*, In Proceedings of the Information Visualization Symposium 1995, IEEE Computer Society Press, pages 51-58.
- [WW99] J.J. van Wijk and H. van de Wetering. *Cushion Treemaps*, In the Proceedings of the IEEE Symposium on Information Visualization (InfoVis'99), IEEE Computer Society, pages 73-78, October 25-26 1999.
- [Zhang97] J. Zhang. *The nature of external representation in problem solving*, Cognitive Science, 21(2):179-217, 1997.