

City Research Online

City St George's, University of London

Citation: Jones, S. (1993). Three-Dimensional Interactive Connection Diagrams for Knowledge Engineering. (Unpublished Doctoral thesis, City, University of London)

This is the submitted version of the paper.

This version of the publication may differ from the final published version. To cite this item please consult the publisher's version.

Permanent repository link: https://openaccess.city.ac.uk/id/eprint/20156/

Copyright and Reuse: Copyright and Moral Rights remain with the author(s) and/or copyright holders. Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge, unless otherwise indicated, provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way. For full details of reuse please refer to City Research Online policy.

City Research Online: http://openaccess.city.ac.uk/ publications@citystgeorges.ac.uk

Three-Dimensional Interactive Connection Diagrams for Knowledge Engineering

Sara Jones
Department of Computer Science
City University

February 1993

This thesis is submitted as part of the requirements for a Ph.D. in Computer Science, in the Department of Computer Science of City University, London, England.

Contents

Acknowledgements xvii

Declaration xix

Abstract xxi

I Introduction 1

1 Introduction 3

1.1 Overview 3

1.2 Definitions 4

1.3 Context 4

1.4 Objectives 9

1.5 Thesis Outline 10

2 Graphical Support for Human-Computer Interaction 13

2.1 Introduction 13

2.2 Evolution of the Human-Computer Interface 13

2.3 Arguments for the Use of Graphical Representations 14

2.3.1 Information Bandwidth 14

2.3.2 Support for Mental Representations 15

2.3.3 Other Claims 15

2.4 Graphical User Interfaces: Some Examples 16

2.4.1 Virtual Reality 16

2.4.2 Pictures, Models and Metaphors 17

2.4.3 Representing the Abstract 20

2.5 Conclusions 22

2.6 Summary 22

3 Graphical Support for Knowledge Engineering **25**

3.1 Introduction 25

3.2 Knowledge Based System Human-Computer Interface Requirements . . . 25

3.2.1 The End User 26

3.2.2 The Domain Expert 26

3.2.3 The Knowledge Engineer 26

3.3 Arguments for the Use of Graphical Representations 27

3.4 Tools Providing Graphical Support for Knowledge Engineering 28

3.4.1 Knowledge Acquisition and Validation 28

3.4.2 Design and Development 29

3.4.3 Debugging and Verification 31

3.5 Paradigms for Graphical Knowledge Representation 32

3.5.1 Graphical Vocabulary 32

3.5.2 Animation 33

3.5.3 Spatial Coding 35

3.6 Conclusions 37

3.7 Summary 37

II Depth in 3-d ICDs **39**

4 Representation of Depth in 3-d ICDs **41**

4.1 Introduction 41

4.2 Cues Used in Human Depth Perception 41

4.2.1 Monocular Cues (No Movement) 42

4.2.2 Monocular Cues (Movement) 43

4.2.3 Binocular Cues 43

4.3 Techniques for the Display of 3-d Computer Generated Images 44

4.3.1 Real and Virtual 3-d Images 44

4.3.2 2-d Images with Illusory Depth 44

4.4 Choice of Techniques for the Display of 3-d ICDs 46

4.4.1 Requirements for Realism 46

4.4.2 Hardware Constraints 46

4.4.3 Depth Cues in JIN, JINGLE and ICDEDIT 47
4.5 Summary 48

5 Perception of Depth in 3-d ICDs: A Pilot Study **49**
5.1 Introduction 49
5.1.1 Hypotheses 50
5.1.2 Choice of Task 51
5.1.3 Choice of Stimuli 51
5.2 Method 51
5.2.1 Design 51
5.2.2 Apparatus 52
5.2.3 Procedure 52
5.2.4 Subjects 54
5.3 Results 54
5.3.1 Response Time Analysis 54
5.3.2 Analysis of Error Rates 54
5.3.3 Analysis of Questionnaire Data 55
5.3.4 Further Analysis of Response Times 58
5.4 Conclusions 58
5.5 Discussion 59
5.5.1 Support for Experimental Hypotheses 59
5.5.2 Implications for Further Experiments 60
5.6 Summary 60

6 Perception of Depth in 3-d ICDs: Further Experiments **61**
6.1 Introduction 61
6.1.1 Hypotheses 62
6.1.2 Choice of Task 63
6.1.3 Choice of Stimuli 63
6.2 Method 64
6.2.1 Design 64
6.2.2 Apparatus 64
6.2.3 Procedure 66
6.2.4 Subjects 69

6.3 Results 69
6.3.1 Methods of Analysis 69
6.3.2 Effect of Angle of Motion 70
6.3.3 Effect of Period of Motion 73
6.3.4 Effect of Stimulus Type 77
6.3.5 General Comments 78
6.3.6 Analysis of Difficult Pairs 81
6.4 Conclusions 84
6.5 Discussion 85
6.5.1 Comments on the Experimental Design 85
6.5.2 Implications for Future Implementations 86
6.5.3 Implications for the Layout of 3-d ICDs 86
6.5.4 Implications for Phase Two 87
6.6 Summary 88

III Utility and Usability of 3-d ICDS 89

7 Investigation of Utility and Usability 91
7.1 Introduction 91
7.2 Estimation of Utility 92
7.2.1 Definitions of Utility 92
7.2.2 Approaches to the Estimation of Utility 92
7.3 Evaluation of Usability 93
7.3.1 Definitions of Usability 93
7.3.2 Approaches to the Evaluation of Usability 94
7.4 Case Study Design 97
7.4.1 Ecological Validity 100
7.4.2 Procedure 101
7.4.3 Analysis of the Results 104
7.5 Summary 104

8 Utility of 3-d ICDs for Knowledge Engineering 113

8.1 Introduction . . . 113

8.2 User Skills and Experience . . . 113

8.2.1 Perceptual Capabilities . . . 114

8.2.2 Visualisation Skills . . . 114

8.2.3 Experience with Graphical Representations for Knowledge Engineering . . . 115

8.2.4 Experience with Knowledge-Based Systems . . . 115

8.3 Task Factors . . . 116

8.3.1 Knowledge Acquisition . . . 118

8.3.2 Knowledge Validation . . . 119

8.3.3 Design . . . 120

8.3.4 Development . . . 121

8.3.5 Debugging . . . 121

8.3.6 Verification . . . 122

8.3.7 Maintenance . . . 122

8.4 Environmental Factors . . . 124

8.4.1 Organisational Environment . . . 124

8.4.2 Physical Environment . . . 125

8.4.3 Computing Environment . . . 125

8.5 System Characteristics . . . 126

8.5.1 Knowledge Representation Style . . . 128

8.5.2 System Size . . . 129

8.5.3 System Complexity . . . 130

8.6 Recommendations for Further Development . . . 132

8.7 Summary . . . 132

9 Usability of Depth-Related Features of ICDEDIT 133

9.1 Introduction . . . 133

9.2 Effectiveness of Support for Perceiving Depth . . . 134

9.2.1 Hiding . . . 134

9.2.2 Perspective . . . 135

9.2.3 Rocking Motion . . . 135

9.2.4 Colour . . . 136

9.2.5 Summary and Recommendations 136
9.3 Effectiveness of Support for Manipulating 3-d ICDs 137
9.3.1 Strategies 137
9.3.2 Problems 138
9.3.3 Summary and Recommendations 138
9.4 Effectiveness of Support for Positioning Elements in 3-d ICDs 139
9.4.1 Strategies 139
9.4.2 Summary and Recommendations 140
9.5 User Satisfaction 140
9.5.1 Summary and Recommendations 141
9.6 Summary 141

IV Conclusion **143**

10 Further Investigation of the Utility of 3-d ICDs for Knowledge Structure Representation **145**
10.1 Introduction 145
10.2 Experimental Comparison of 3-d, 2-d and Textual Representations of Knowledge Structures 146
10.2.1 Method 147
10.2.2 Results 155
10.3 Discussion 156
10.3.1 Problems with the Use of Controlled Experiments in Design 156
10.3.2 An Alternative Approach to Assessing Utility 159
10.4 Summary 161

11 Formal Specification of 3-d ICD Languages for Knowledge Structure Representation **163**
11.1 Introduction 163
11.2 Definition of Visual Languages 164
11.3 Specification of 3-d ICD Languages Supported by ICDEDIT 165
11.3.1 Abstract Graphs 165
11.3.2 Graphical Primitives 166
11.3.3 Vocabularies 167
11.3.4 Diagrams 168

11.3.5 Visual Languages 169
11.3.6 Primitive Operations 169
11.4 Specification of a Particular 3-d ICD Language 173
11.4.1 Knowledge-Based System Profile 174
11.4.2 Representation Requirements 174
11.4.3 Representation Specification 175
11.5 Discussion 179
11.6 Summary 181

12 Discussion **183**
12.1 Overview 183
12.2 Evaluation 184
12.2.1 Investigation of Depth Cuing in 3-d ICDs 186
12.2.2 Estimation of the Utility of 3-d ICDs for Knowledge Engineering . 187
12.2.3 Evaluation of the Usability of Features of ICDEDIT 189
12.2.4 Formal Specification of 3-d ICD Languages 190
12.3 Further Development of Tools 191
12.3.1 Presentation of 3-d ICDs 191
12.3.2 Layout of 3-d ICDs 192
12.3.3 Further Functionality 193
12.4 Conclusions 193
12.5 Summary 194

Appendices **195**

A Pilot Experiment: Materials and Results **197**
A.1 Stimuli 197
A.2 Protocol 203
A.3 Questionnaire 205
A.4 Results 207

B Further Experiments: Materials and Results **209**
B.1 Stimuli 209
B.2 Protocol 226

B.3 Questionnaire 227
B.4 Results 228
B.4.1 Effect of Angle of Motion 228
B.4.2 Effect of Period of Motion 235
B.4.3 Effect of Stimulus Type 243

C ICDEDIT: A Brief Description **259**
C.1 Introduction 259
C.2 The ICDEDIT 2¾-d Diagram Paradigm 261
C.3 The ICDEDIT User Action Paradigm 262

D Case Studies: Materials **275**
D.1 Stage 1 276
D.1.1 Interview Prompt Sheet 276
D.2 Stage 2 281
D.2.1 Interview Prompt Sheet 281
D.3 Stage 3 283
D.3.1 Interview Prompt Sheet 283
D.3.2 Generic Task Taxonomy 292
D.3.3 Graphical Representations 295

E Further Experiments for Investigating the Utility of 3-d ICDs for Knowledge Structure Representation **309**
E.1 Effects of User Characteristics 310
E.1.1 Visualisation Skills 310
E.1.2 Experience with Graphical Representations 310
E.1.3 Experience with Knowledge-Based Systems 310
E.2 Effects of Environmental Factors 311
E.2.1 Knowledge Representation Formalism 311
E.3 Effects of System Characteristics 311
E.3.1 System Size 311
E.3.2 Number of Different Relations 312
E.3.3 Number of Relationships per Entity 312

F Introduction to the Z Notation 313

F.1 Glossary of Symbols 315

G Formal Specifications of 3-d ICD Representations 317

G.1 Introduction 317

G.2 Case Study 1 318

G.2.1 Knowledge-Based System Profile 318

G.2.2 Representation Requirements 318

G.2.3 Representation Specification 319

G.3 Case Study 3 325

G.3.1 Knowledge-Based System Profile 325

G.3.2 Representation Requirements 325

G.3.3 Representation Specification 326

G.4 Case Study 4 331

G.4.1 Knowledge-Based System Profile 331

G.4.2 Representation Requirements 331

G.4.3 Representation Specification 332

G.5 Case Study 5 335

G.5.1 Knowledge-Based System Profile 335

G.5.2 Representation Requirements 335

G.5.3 Representation Specification 336

G.6 Case Study 6 339

G.6.1 Knowledge-Based System Profile 339

G.6.2 Representation Requirements 339

G.6.3 Representation Specification 340

Bibliography 345

List of Figures

1.1 Example of Graphical Knowledge Representation 5
1.2 3-d ICD Drawn Using JIN 6
1.3 3-d ICD Drawn Using JINGLE 6
1.4 3-d ICD Drawn Using ICDEDIT 7

2.1 A Virtual Reality Interface 17
2.2 Graphical Representation Used in a Tool for CAD 18
2.3 Graphical Representation Used in Steamer 19
2.4 Graphical Representation Used in Molecular Modelling 19
2.5 3-Dimensional Representation of Numerical Data 21
2.6 Graphical Representation Used in Pict 22

3.1 Concept Maps in GIS 30
3.2 LOOPS class browser 32
3.3 KEE Class Hierarchy 33
3.4 Iconic Representations Used in KEE 34
3.5 Iconic Representations Used in TMV 34
3.6 2-Dimensional Representation Used in TRI 35
3.7 3-Dimensional Representation Suggested for Use in the Casnet System . . 36

4.1 Rocking Motion in JIN 48

5.1 One of the Stimuli Used in the Pilot Study 53
5.2 Plot of Response Times in the Pilot Study 55
5.3 Plot of Ease Scores in the Pilot Study 56

6.1 One of the Type I Stimuli (Layered) 67
6.2 One of the Type II Stimuli (Hierarchic) 67

6.3 One of the Type III Stimuli (Random) 68
6.4 One of the Type IV Stimuli (Regular) 68
6.5 Graphs showing the effects of angle of motion on total numbers of errors and on median ratings of ease and confidence for all subjects 71
6.6 Graphs showing the effects of period of motion on mean logged response times, total numbers of errors and median ratings of ease for all subjects . 74
6.7 Graphs showing the effects of stimulus type on total numbers of errors and median ratings of confidence and complexity for all subjects 79
6.8 Graphs showing the relationship of ratings of perceived complexity to total numbers of errors and ratings of confidence for all subjects 80
6.9 Examples of Masking 83
6.10 Examples of Perspective Cues from Links 83

7.1 Training Diagram 105
7.2 Key Describing the Visual Language Used in Case Study 2 107
7.3 Dummy 3-d ICD Representation Used in Case Study 2 109
7.4 Actual 3-d ICD Representation Used in Case Study 2 111

10.1 Summary of Experiment Design 148
10.2 Examples of 3-d Stimuli 151
10.3 Examples of 2-d Stimuli 152
10.4 Examples of Textual Stimuli 153
10.5 Examples of Questions 154
10.6 Predicted Results 157
10.7 Error Rate Tables 158

12.1 Chart of the Parse Space Produced by Lucy 185
12.2 Cone Tree Used in the Information Visualiser 185

A.1 Pilot Experiment: Stimulus 1 198
A.2 Pilot Experiment: Stimulus 2 199
A.3 Pilot Experiment: Stimulus 3 200
A.4 Pilot Experiment: Stimulus 4 201
A.5 Pilot Experiment: Stimulus 5 202

B.1 Type I Stimuli: Layered (1) 210
B.2 Type I Stimuli: Layered (2) 211

B.3 Type I Stimuli: Layered (3) 212
B.4 Type I Stimuli: Layered (4) 213
B.5 Type II Stimuli: Hierarchic (1) 214
B.6 Type II Stimuli: Hierarchic (2) 215
B.7 Type II Stimuli: Hierarchic (3) 216
B.8 Type II Stimuli: Hierarchic (4) 217
B.9 Type III Stimuli: Random (1) 218
B.10 Type III Stimuli: Random (2) 219
B.11 Type III Stimuli: Random (3) 220
B.12 Type III Stimuli: Random (4) 221
B.13 Type IV Stimuli: Regular (1) 222
B.14 Type IV Stimuli: Regular (2) 223
B.15 Type IV Stimuli: Regular (3) 224
B.16 Type IV Stimuli: Regular (4) 225

C.1 A Small ICDEDIT Diagram 260
C.2 ICDEDIT in Use 265
C.3 Node Selected 267
C.4 Link Selected 267
C.5 Selected Node Border Edit Menu Raised 269
C.6 Selected Link Arrow Head Edit Menu Raised 269
C.7 View Control Panel Raised 271
C.8 Colour Edit Panel Raised 271
C.9 Frontal View of a 3-d ICD 273
C.10 Side View of the Same 3-d ICD 273

D.1 ICDEDIT Vocabulary 282
D.2 Key Describing the Visual Language Used in Case Study 1 297
D.3 Actual 3-d ICD Representation Used in Case Study 1 297
D.4 Key Describing the Visual Language Used in Case Study 2 299
D.5 Actual 3-d ICD Representation Used in Case Study 2 299
D.6 Key Describing the Visual Language Used in Case Study 3 301
D.7 Actual 3-d ICD Representation Used in Case Study 3 301
D.8 Key Describing the Visual Language Used in Case Study 4 303

D.9 Actual 3-d ICD Representation Used in Case Study 4 303
D.10 Key Describing the Visual Language Used in Case Study 5 305
D.11 Actual 3-d ICD Representation Used in Case Study 5 305
D.12 Key Describing the Visual Language Used in Case Study 6 307
D.13 Actual 3-d ICD Representation Used in Case Study 6 307

List of Tables

6.1 Summary of the Effects of Angle of Motion 70

6.2 Summary of the Effects of Period of Motion 75

6.3 Summary of the Effects of Stimulus Type 78

6.4 Numbers of Pairs Causing Particular Numbers of Errors 82

7.1 Details of Case Studies 1 - 3 98

7.2 Details of Case Studies 4 - 6 99

8.1 Effects of User Skills and Experience on Utility of 3-d ICDs for Knowledge Engineering 117

8.2 Utility of 3-d ICDs for Various Knowledge Engineering Tasks 123

8.3 Effects of Environmental Factors on Utility of 3-d ICDs for Knowledge Engineering 127

8.4 Utility of 3-d ICDs in the Representation of Systems of Different Kinds . 131

B.1 Mean Logged Response Times by Angle of Motion: Experiment 1 230

B.2 Total Errors by Angle of Motion: Experiment 1 231

B.3 Median Ease Ratings by Angle of Motion: Experiment 1 232

B.4 Median Confidence Ratings by Angle of Motion: Experiment 1 233

B.5 Median Comfort Ratings by Angle of Motion: Experiment 1 234

B.6 Mean Logged Response Times by Period of Motion: Experiment 2 237

B.7 Total Errors by Period of Motion: Experiment 2 238

B.8 Median Ease Ratings by Period of Motion: Experiment 2 239

B.9 Median Confidence Ratings by Period of Motion: Experiment 2 240

B.10 Median Comfort Ratings by Period of Motion: Experiment 2 241

B.11 Mean Logged Response Times by Stimulus Type: Experiment 1 246

B.12 Mean Logged Response Times by Stimulus Type: Experiment 2 247

B.13 Total Errors by Stimulus Type: Experiment 1 248

B.14 Total Errors by Stimulus Type: Experiment 2 249
B.15 Median Ease Ratings by Stimulus Type: Experiment 1 250
B.16 Median Ease Ratings by Stimulus Type: Experiment 2 251
B.17 Median Confidence Ratings by Stimulus Type: Experiment 1 252
B.18 Median Confidence Ratings by Stimulus Type: Experiment 2 253
B.19 Median Comfort Ratings by Stimulus Type: Experiment 1 254
B.20 Median Comfort Ratings by Stimulus Type: Experiment 2 255
B.21 Median Complexity Ratings by Stimulus Type: Experiment 1 256
B.22 Median Complexity Ratings by Stimulus Type: Experiment 2 257

Acknowledgements

I hereby gratefully acknowledge the funding from the Science and Engineering Research Council and British Telecommunications which supported the bulk of this work.

Particular thanks are also due to a number of individuals: to David Bolton for his enthusiastic reading of an earlier draft of this thesis; to Jun Cui for his work in developing the software used in the studies reported; to David Dodson who supervised the ICDEDIT project and the writing of an earlier draft of this thesis; to James Hampton for his advice regarding the design of some of the experiments; to Chris Marshall and Nigel Mitchem who helped me with many technical difficulties; to Alistair Sutcliffe for his guidance at various stages; to David Till for discussions regarding the work using Z; to Darren Van Laar for numerous helpful suggestions; and to Luke Whitaker for assistance with much of the statistical analysis. I am also grateful to those who acted as subjects in the studies reported below.

I would like to offer heartfelt thanks to all those who provided support, encouragement, understanding and tolerance without whom I could not have completed this work.

Finally, I would like to thank my parents, Vivienne and Denis Jones for the love and support they have always given me, and for their continuing vitality and enthusiasm.

Declaration

I grant powers of discretion to the University Librarian to allow this thesis to be copied in whole or in part without further reference to me. This permission covers only single copies made for study purposes, subject to normal conditions of acknowledgement.

Abstract

This thesis describes research into human factors aspects of the use of 3-dimensional node and link diagrams, called Interactive Connection Diagrams (ICDs), in the human-computer interface of tools for knowledge engineering. This research was carried out in two main stages: the first concentrated on perceptual aspects of 3-d ICDs, and the second on more general aspects of their use in realistic situations. A final section looked briefly at the possibility of formally specifying 3-d ICD representations.

The main aim of the first stage was to investigate whether users were able to make effective judgements about the relative depths of components in 3-d ICDs. Controlled experiments were carried out to determine the extent to which such judgements were supported by the use of a particular approach to creating the illusion of depth. The results of these experiments showed that users were able to make reasonably effective judgements about the relative depths of components in 3-d ICDs. 3-d ICDs produced using the approach of interest were therefore argued to be suitable for use in the second stage of the study.

In the second stage, case studies were used to investigate the utility in more realistic knowledge engineering situations of tools supporting 3-d ICDs, and the usability of depth-related features of a prototype tool which permits 3-d ICDs to be viewed and edited. On the basis of the findings of these studies it is claimed that tools supporting 3-d ICDs will, in some situations, be more useful than those which employ only more conventional 2-d versions. It was found that depth-related features of the prototype tool were usable but should be improved upon in future implementations.

The third and final section of work involved a preliminary investigation into the formal specification of the 3-d ICD representations of the kind used in the second set of studies. A scheme for specifying the range of 3-d ICD languages currently supported by the prototype tool was developed, and each of the particular 3-d ICD languages used in the case studies were specified.

Implications of the results of this work are discussed and a number of suggestions regarding directions for future work are made. The overall conclusion is that 3-d ICDs have considerable potential as a medium in which to represent knowledge structures for use in knowledge engineering.

Part I

Introduction

This part of the thesis contains introductory material. The first chapter presents an overview of the aims and objectives of the work described in later chapters and sets out some important definitions. Chapter 2 describes how techniques for the provision of graphical support for human-computer interaction have evolved and gives some examples of the ways in which such support is currently provided. Chapter 3 focuses specifically on the provision of graphical support for knowledge engineering and presents a review of the kinds of support currently provided for this task.

Chapter 1

Introduction

1.1 Overview

This thesis describes research into human factors aspects of the use of 3-dimensional node and link diagrams in the human-computer interface of tools for knowledge engineering. Such diagrams are here referred to as 3-dimensional Interactive Connection Diagrams, or 3-d ICDs [39]. A number of empirical studies were conducted in order to investigate the use of 3-d ICDs produced by tools embodying a particular approach to creating the illusion of depth. This approach is described in outline in section 1.3.

The first of the studies investigated perceptual aspects of 3-d ICDs. Three controlled experiments were used to investigate the effectiveness of the approach to creating the illusion of depth used in the tools of interest. On the basis of the results of these experiments it was concluded that:

- users are able to make reasonably effective judgements about the relative depths of components in 3-d ICDs

and that 3-d ICDs produced using this approach constituted a reasonable medium for use in further investigations into meaningful depth-based spatial coding in graphical representations of knowledge structures for use in knowledge engineering.

The use of 3-d ICDs in more realistic knowledge engineering situations was investigated in a second set of six case studies. These studies investigated the utility in knowledge engineering of interface technologies based on the use of 3-d ICDs, and the usability of depth-related features of a particular prototype tool which permits 3-d ICDs to be viewed and edited. On the basis of the findings of these studies it is claimed that:

- interface technologies based on the use of 3-d ICDs will, in some knowledge engineering situations, be more useful than those which employ only more conventional 2-d versions;
- depth-related features of the prototype tool are usable but may be improved in future implementations.

Finally, some work was done on investigating the possibilities of formalising specifications of the 3-d representations used in the second set of studies. As a result of this work, it is claimed that:

- 3-d ICD representations of knowledge structures for use in knowledge engineering can be formally specified using the Z notation.

1.2 Definitions

Three-dimensional Interactive Connection Diagrams may be defined as follows [38].

A *connection diagram* (CD) is a diagram or graphical representation in which nodes are used to denote important concepts or entities and links may be used to represent any connections between them.

An *interactive connection diagram* (ICD) is a connection diagram which supports interaction of some kind: for example, human-computer interaction.

A *3-dimensional interactive connection diagram* (3-d ICD) is an ICD whose nodes have their positions defined in terms of 3 co-ordinates. Figuratively speaking, a 3-d ICD may therefore look more like a graphical representation of a structure in space than a drawing on a page.

1.3 Context

Two-dimensional node and link or connection diagrams are widely used in the human-computer interface. Three-dimensional versions are much less common, though interest has recently been growing and a number of researchers have alluded to the potential benefits of using 3-d ICDs [128, 52, 116].

Benefits cited for the use of 3-dimensional layouts rather than 2-d include:

- greater freedom of layout and expression [128]
- greater visibility of information for a given size of screen or window [116]
- the availability of an extra dimension for spatial coding in representations of multi-dimensional information [52]

Work described in this thesis investigates the use of 3-d ICDs in the domain of knowledge-based systems: specifically, in the process of knowledge engineering. Knowledge engineering is the process of designing, developing, implementing and maintaining knowledge-based systems. Two-dimensional node and link diagrams are already widely used in providing graphical representations of abstract components of systems or knowledge structures for use by knowledge engineers (see chapter 3). They are, however, often inadequate for the representation of the large, complicated and inherently multi-dimensional knowledge structures which lie at the heart of modern knowledge-based systems. It was therefore felt that the benefits of using 3-d ICDs (rather than 2-d) might be considerable in this field.

The work described in this thesis was carried out at the same time as other work at City University aimed at developing prototype tools to allow users to view and edit 3-d ICDs. All of the tools developed at City have used the same approach to creating the illusion of depth. This approach was intended for use in workstations with 2-d graphics hardware and is described in detail elsewhere [39, 31]. Briefly, it involves a

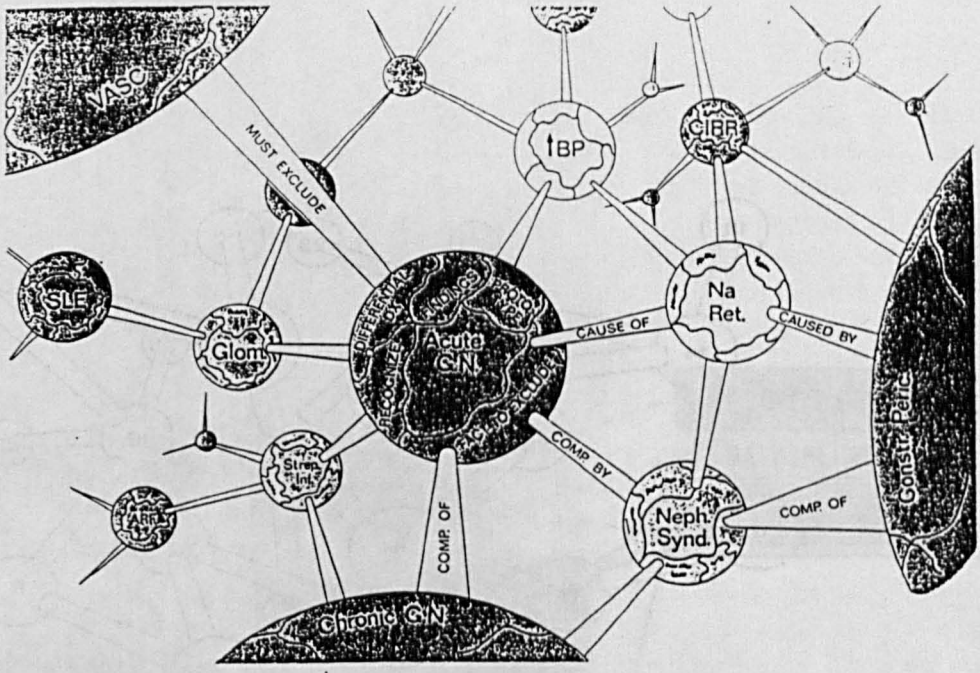


Figure 1.1: Example of Graphical Knowledge Representation

simplification of the 3-d display problem in which nodes (and links) are displayed using purely 2-d methods and are drawn in back-to-front order to create the effect of occlusion. A crude simulation of rocking motion is also employed with the aim of enhancing the illusion of depth. This produces what has been called a '2¾-d' effect [36] in which three-dimensional arrangements of nodes and links can be rotated and transformed while individual nodes always face the front. The speed of display update achieved using this relatively simple approach was found to be acceptable though the process was not perceptually instantaneous.

As stated above, each of the tools developed ran on a general purpose workstation with 2-d graphical hardware. JIN [31, 39], the earliest tool, ran on a Whitechapel workstation with a monochrome monitor and displayed a view of a 3-d structure of spherical nodes and narrow cylindrical links specified in a pre-defined datafile. A precedent for the use of such diagrams can be found in a paper by Pauker *et al.* [105] (see figure 1.1). An example of the kind of diagram which could be produced using JIN is shown in figure 1.2.

The second tool, JINGLE [31], ran on the same sort of workstation and again displayed a view of a 3-d node and link structure which had been previously defined. From the user's point of view, the main differences between JIN and JINGLE were as follows. Instead of using cylinders and spheres, diagrams in JINGLE were made up of single line links and flat rectangular nodes which could contain text. A number of the properties of nodes and links could be defined using menus, and components of JINGLE diagrams could be edited and positioned by direct manipulation. An example of the kind of diagram supported by JINGLE is shown in figure 1.3.

The third tool, ICDEDIT [37] is essentially a port of JINGLE's graphics facilities to a colour Sun 3/60 workstation. An example of the kind of diagram supported by ICDEDIT is shown in figure 1.4. Some effort was spent on development of ICDEDIT's user interface and more of the properties of nodes and links could be defined by form-filling and direct manipulation. Further information regarding ICDEDIT and the kind of diagrams it may be used to create can be found in appendix C.

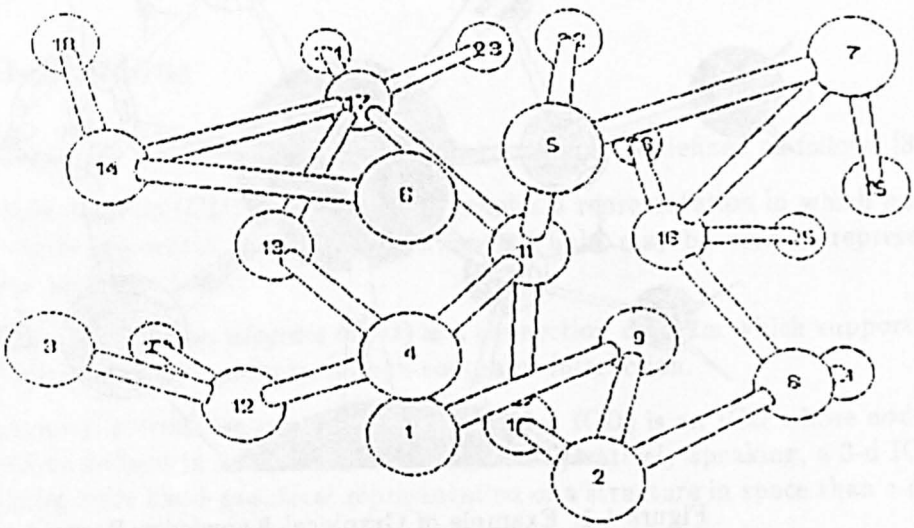


Figure 1.2: 3-d ICD Drawn Using JIN

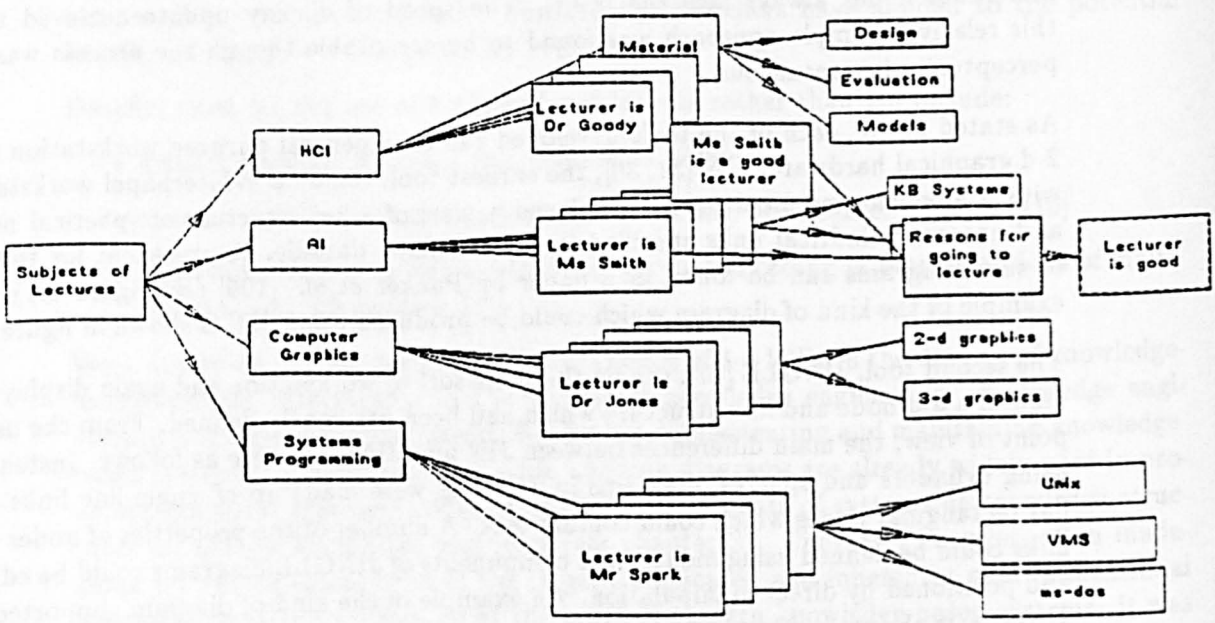


Figure 1.3: 3-d ICD Drawn Using JINGLE

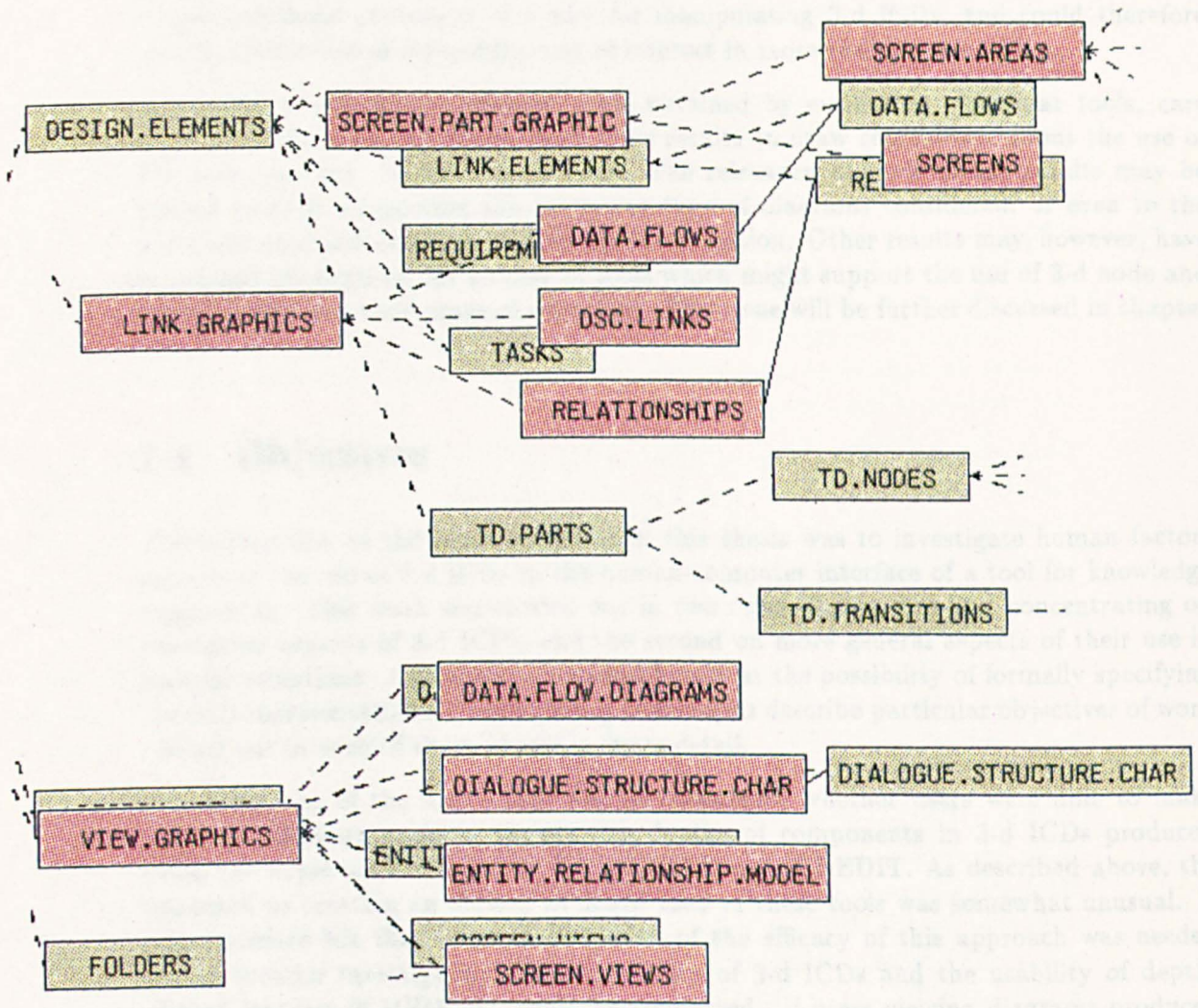


Figure 1.4: 3-d ICD Drawn Using ICDEDIT

As mentioned above, an empirical approach was adopted in carrying out most of the work described in this thesis. The nature of this work therefore reflects, to at least some extent, the nature of the tools available at the time it was conducted. At the time when the early work was carried out, only the first of the prototypes (JIN) had been completed and was available for use: JIN had only minimal interactive functionality and only perceptual aspects of the representations supported (which could be studied in the absence of such functionality) were considered at this stage. Later work used ICDEDIT, a more advanced prototype of a tool for manipulating 3-d ICDs, and could therefore investigate the use of representations of interest in more realistic situations.

Since most of the results reported were obtained by evaluating particular tools, care must be taken in extrapolating from these results to draw conclusions about the use of 3-d node and link diagrams in general. The relevance of some of the results may be limited to tools supporting the particular type of diagrams considered, or even to the particular tool and diagrams used in the investigation. Other results may, however, have important implications for a range of tools which might support the use of 3-d node and link diagrams in a wide range of situations. This issue will be further discussed in chapter 12.

1.4 Objectives

The overall aim of the work described in this thesis was to investigate human factors aspects of the use of 3-d ICDs in the human-computer interface of a tool for knowledge engineering. This work was carried out in two main phases, the first concentrating on perceptual aspects of 3-d ICDs, and the second on more general aspects of their use in realistic situations. A final section looked briefly at the possibility of formally specifying 3-d ICD representations. The following paragraphs describe particular objectives of work carried out in each of these phases in more detail.

The main aim of the first phase was to investigate whether users were able to make effective judgements about the relative depths of components in 3-d ICDs produced using the approach embodied in JIN, JINGLE and ICDEDIT. As described above, the approach to creating an illusion of depth used in these tools was somewhat unusual. It was therefore felt that some investigation of the efficacy of this approach was needed before broader investigations into the utility of 3-d ICDs and the usability of depth-related features of ICDEDIT could be conducted. If users viewing diagrams produced using this approach were unable to see them as three-dimensional in the way intended, then an investigation of the utility and usability of 3-d ICDs *qua* 3-dimensional diagrams would not have been possible.

The aim of the second phase was twofold. The first objective was to investigate the utility in knowledge engineering of tools incorporating 3-d ICDs of the kind supported by ICDEDIT. It was envisaged that these findings could be used in determining whether further development of the existing prototype was justified and, if so, at which aspects of knowledge engineering activity potential future implementations of a tool incorporating 3-d ICDs would most usefully be directed. The second objective was to investigate the usability of features which permit 3-d ICDs to be manipulated in ICDEDIT to help in developing further versions of this tool.

The final section of work involved a preliminary investigation into the formal specifica-

tion of 3-d node and link diagrams of the kind produced by ICDEDIT. This work was done in order to be able to assess whether formal specifications of 3-d node and link representations would be useful in the development of future tools supporting the use of such diagrams.

1.5 Thesis Outline

The thesis begins by presenting introductory and background material about the broad context within which the work described was carried out. The work itself is then described and its results are reported. As explained above, the bulk of the work was carried out in two phases. The first (described mainly in chapters 4, 5 and 6) dealt with controlled experimental investigations of the perceptual properties of 3-d ICDs supported by JIN. The second (described in chapters 7, 8 and 9) adopted a case study approach to the investigation of the utility and usability of 3-d ICDs of the kind supported by ICDEDIT in the context of knowledge engineering. The thesis ends with three chapters which lay the foundations for further work on the use of 3-d ICDs in the human-computer interface of tools for knowledge engineering. The rest of this section provides a brief overview of the contents of individual chapters.

Chapter 2 sets the scene by considering the way in which human-computer interfaces have evolved, often to incorporate a greater emphasis on the use of graphical representations of one form or another. Various forms of graphical representation are considered and a summary of the arguments commonly made in favour of their use is presented. Chapter 3 contains a survey of the literature concerning the use of node and link diagrams in the representation of knowledge structures for knowledge-based systems. Interfaces for various types of user are considered, but the main emphasis is on those for knowledge engineers, that is, for the people who must design, develop and maintain such knowledge-based systems. This brief review is also intended to set the scene for the work reported in the following chapters.

Chapters 4, 5 and 6 describe experiments which investigated whether users were able to make effective judgements about relative depths of components in 3-d ICDs produced using the approach embodied in JIN, JINGLE and ICDEDIT. Chapter 4 describes cues ordinarily used in depth perception and the corresponding mechanisms commonly used to create an impression of depth in so-called 3-d computer-generated images. The cues used in JIN, JINGLE and ICDEDIT are then described in more detail. Chapter 5 describes a pilot experiment carried out to investigate the effectiveness of these cues, and in particular of the use of rocking motion. Chapter 6 reports on two more detailed experiments on the effects of varying the angle and period of the rocking motion and the type of 3-d ICD.

Chapter 7 describes the methods used to conduct the investigation into the utility in knowledge engineering of 3-d ICDs and the usability of features of ICDEDIT. Six case studies were carried out with the aim of looking at real users working with 3-d representations of real systems and performing tasks which were as realistic as possible within the constraints imposed by the prototypical nature of the tool. In each study, knowledge engineers were interviewed, and 3-d ICD representations of knowledge-based systems with which they had recently been working were developed for use in the subsequent phase of usability evaluation. Chapter 8 discusses issues relating to the utility of 3-d ICDs and chapter 9 reports on observations relating to the usability of depth-related features of ICDEDIT.

The final chapters draw on work described earlier in the thesis to provide a basis for further work in this area. Chapter 10 describes some empirical studies which could be used to further investigate some of the issues discussed in chapter 8. Chapter 11 presents work on the formal specification of 3-d node and link diagrams of the kind used in the case studies and includes a specification of one of those representations as an example of the application of this work. (Specifications of each of the other representations from the case studies are included in appendix G). Finally, work described in the rest of the thesis is discussed and evaluated in chapter 12 where further suggestions regarding ways in which this work could in future be extended are also given.

Chapter 2

Graphical Support for Human-Computer Interaction

2.1 Introduction

This chapter aims to set the scene for the research described in the body of the thesis. A brief history of the development of graphical user interfaces is given. Arguments made in support of the use of such interfaces are presented and the applicability of these arguments to the case of interfaces based on the use of 3-dimensional representations is discussed. Some examples of the way graphical representations are currently used in various areas of application are given.

2.2 Evolution of the Human-Computer Interface

In the early days, when processing power was at a premium, commands issued by the user needed to be structured in such a way that their interpretation could be achieved using a minimum of computer resources. This meant typing commands to a dumb terminal in a language not too far removed from assembly code. As processing power became more readily available, it was, however, possible to shift the emphasis away from the needs of the computer and towards those of the human operator. By the late 1980's, the limiting factor on interactive (human-computer) system efficiency was often the cognitive capacity of the human rather than the performance of the computer [93].

At the same time as processors were becoming more powerful, rapid developments in I/O technology were also taking place. Terminals were no longer completely dumb, and display screens were larger and more able to handle detailed non-textual information due to higher resolution. This, coupled with the development of new I/O devices such as trackerballs, joysticks and mice lead to the introduction of new techniques for interaction such as direct manipulation [125, 126].

The emergence of this enabling technology lead to the development of graphical interfaces to systems in many different application areas as will be seen below. Research on the way in which such interfaces could most effectively be used was begun [58, 59]. The development of WIMP (Window, Icon, Menu, Pointer) and direct manipulation interfaces which relied on an increased use of graphical representations was a further step forward.

The next major development which seems set to take the market by storm is that of 'virtual reality' interfaces which use more advanced graphical representations as a basis on which to create the illusion of a 'virtual' or 'artificial' reality for the user of a computer system (see section 2.4.1). Negroponte predicts that our personal computers may in future support animated holographic images of agents or actors who wait on a stage ready to do our bidding [99]. Whatever the form of the human- computer interface in years to come, graphical representations of some form are, it seems, here to stay.

The following section presents an overview of the arguments which have been made in support of the use of graphical representations in the human-computer interface.

2.3 Arguments for the Use of Graphical Representations

A number of studies have been conducted to show that performance of problem-solving tasks can be significantly affected by the way in which the relevant information is presented, independent of problem complexity [18, 158, 123]. In these studies, it is very often the case that more graphical forms of representation lead to superior task performance. Some studies have also attempted to demonstrate beneficial effects of presenting information in a 3-d form rather than relying on simple 2-d representations [4] though the results of these studies have not been widely generalisable. This section will present some of the general points which are often raised in favour of the use of graphics in the human-computer interface and will briefly explore the possible extension of these arguments (often raised in connection with 2-d representations) to the context of 3-dimensional schemes.

2.3.1 Information Bandwidth

One of the claims most commonly made in favour of graphical representations is that communication involving graphical information can take place over a much higher bandwidth than is possible using sequential text-based representations alone. This has several implications. It means that users should find it much easier to obtain some form of 'gestalt' or overview of the information from graphical representations than from text. Scanning for salient information should therefore also be easier. It also means that a given piece of information may often be expressed in a more compact form using some kind of graphical notation.

Communication based on the use of 3-dimensional representations can take place over an even higher bandwidth than that using 2-d representations. Robertson *et al.* [116] explain how more information can be displayed on a given screen using 3-d representations than is possible with 2-d. If spatial information is used as the basis for graphical coding of information of interest, then the availability of three dimensions rather than two will, of course, permit the coding of at least one further variable, thus again increasing the amount of information present in a single image. This might indeed mean that users are able to obtain a gestalt of larger chunks of information than would be possible with 2-d. Scanning for salient information may, however, be harder using a 3-dimensional representation than it would be with a 2-d version, partly because there may be more non-salient information to be filtered out in a 3-d display, and partly because some of

the information (for example that shown 'further back' in the display) may sometimes be hidden from view.

2.3.2 Support for Mental Representations

Another set of related claims concerns the way in which graphical representations of information may map onto the user's own internal or mental representations of the same information. The strong claim here is that external graphical representations may map directly onto internal mental images with which people can reason directly using well-advanced visual recognition and pattern-matching capabilities. For example, Rumelhart *et al.* [120] suggest that since relations among real world, physical objects are often understood in visual or spatial terms, it should be easier to derive a mental model of an unknown system structure from a graphical representation than from a textual one. Larkin and Simon [85] also speculate on the possibility of such a mapping and suggest that these kinds of mental images, derived from external graphical representations should provide a computationally efficient means of problem-solving because of the ease with which information stored in such a form might be indexed.

The status of these claims is by no means clear. The debate as to the nature of mental imagery continues [3, 109, 55, 56] and it is certain that there are individual differences in the extent to which people tend to 'visualise' information. The type of information, or task which the user wishes to perform with it, are also likely to have an influence on whether users will tend to think in graphical terms or whether the 'natural idiom' will be graphical [136]. It does, however, seem likely that for some users and some kinds of information, a graphical representation might prove to be invaluable in providing exactly the kind of 'cognitively transparent representation' which is needed for effective comprehension and efficient solving of problems [20].

The extension of these arguments to the context of 3-dimensional representations is also by no means straightforward. Since real world, physical objects are seen by most people to be 3-dimensional, it might be thought that the relationships between them would be understood in terms of three dimensions. But just as it is not clear whether, or under what circumstances, people tend to visualise information, it is also not clear under what circumstances people might prefer to visualise information in terms of three dimensions rather than two. Whether or not such 3-dimensional representations would 'naturally' be used need not, however, be seen to be very important. If 3-d representations are developed on a strong conceptual basis (for example by using two or three of the available dimensions to order the representation with respect to salient concepts), they are likely to be easily comprehensible and may themselves form a basis on which 3-dimensional mental representations can be developed. This in turn will assist the comprehension of further similar 3-d graphical representations.

2.3.3 Other Claims

Whatever the status of this last set of claims, we can end on two more which may be stated with a greater degree of certainty. Firstly, there is a great deal of evidence to support the claim that graphical information is much more easily remembered than textual or verbal information. A greater proportion of pictorial information is retained, with recall being significantly better for pictures than for words, and the capacity for recognition of

pictures being almost limitless when measured under appropriate conditions [135]. This might be equally true of both 2-d and 3-d representations.

Lastly, but by no means least importantly, numerous studies have reported how computer users simply seem to enjoy interacting with their machines by using graphical representations much more than they do using text alone. Increasingly sophisticated 3-d representations are likely to be even more fun to work with than the old 2-d versions.

2.4 Graphical User Interfaces: Some Examples

This section presents some examples of particular ways in which graphical representations of various kinds have been used in various areas of application.

With the advent of the enabling technologies described above, the graphical bandwagon really began to roll. Software vendors were suddenly anxious to claim that their packages too supported Graphical User Interfaces or GUIs. This meant, in many cases, no more than that lines or boxes were drawn around chunks of text. Although this kind of facility can be of some use (for example in drawing attention to a particular piece of text by surrounding it with a brightly coloured or flashing box), it allows little more than would be possible with a purely text-based system. This kind of facility will therefore be discussed no further. Neither will the provision of windows, menus and graphical pointers by virtue of which many other software systems are said to possess GUIs. The emphasis will instead be on application areas in which the information of primary interest (that is, information about the application domain, rather than that about how the application itself works) has been displayed in a graphical form.

'Graphical' here can be taken to mean 'not purely textual', although many of the representations discussed will include some textual elements. (According to Fitter and Green [57], usable notations always contain both symbolic (textual) and perceptual (graphical) elements.) The order in which various areas of application are considered reflects the degree of realism or faithfulness of representation in the graphical images they use. The most realistic are presented first.

2.4.1 Virtual Reality

Virtual reality systems achieve what is currently the closest we can get to a realistic representation of objects in the domain of interest. By wearing a head-mounted monitor (with stereoscopic display), earphones and a dataglove, users can gain a strong impression of interacting directly with the virtual world created in software (see figure 2.1). They can not only see the objects in that world, but hear and feel them too, as the headphones and dataglove provide for audio and tactile feedback.

This technology is relatively new and is as yet not in widespread use. It is therefore difficult to assess how useful it might eventually prove to be. Similar systems have been used at NASA in training astronauts to cope with unusual environments, and also in training people to work in dangerous environments such as failed nuclear reactors which cannot otherwise be easily simulated [142].

Mercurio and Erikson [94] describe experiments with a virtual reality system in the field of scientific visualisation. The task here was to examine a 3-d graphical representation of the

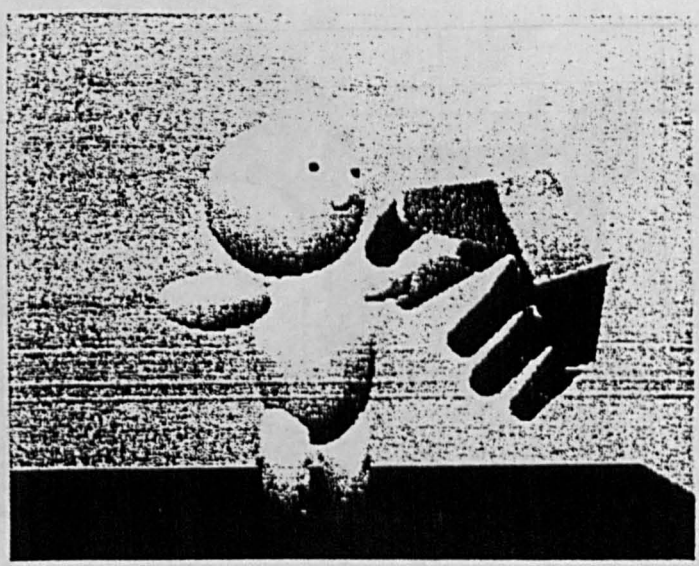


Figure 2.1: A Virtual Reality Interface

human brain with the aim simply of evaluating the possibilities for interaction provided by such a system. They report that their experience of virtual reality in this context was 'both disappointing and exciting': disappointing in the sense that the technology is not yet sufficiently developed, the hardware not yet sufficiently powerful and the mechanisms and metaphor for interaction not sufficiently well understood, but exciting in that there is apparently an enormous potential for displaying a wide variety of different types of data in such a way as to make it easy and useful to interact with.

Virtual reality systems allow users to draw heavily on their understanding of the real world in their interactions with data displayed. This should enhance their ability to appreciate, understand, reason about and remember information presented in such a way. They are also fun to use, as witness the increasing inclusion of elements of virtual reality in today's arcade games.

2.4.2 Pictures, Models and Metaphors

Applications in a number of areas use pictures or models of objects in the domain of interest and support a variety of metaphors which the user must learn in order to interact with them.

CAD packages initially used mainly 2-d projections so that designing an object using a computerised tool was very similar to drawing a plan or blueprint on paper, except for relatively trivial differences concerning modifiability and ease of re-use. It is now more common to use 3-d graphical representations (see, for example, figure 2.2) with more powerful systems allowing the users to manipulate the models, using a joystick or trackerball, or even a keyboard, in a way analogous to that in which a concrete scale model might be turned over in space. Still more powerful systems allow users to animate their models so that, for example, parts of an engine can be seen working together in much the same way as they would do in real life.

Computer-based training systems also commonly use graphical representations of the

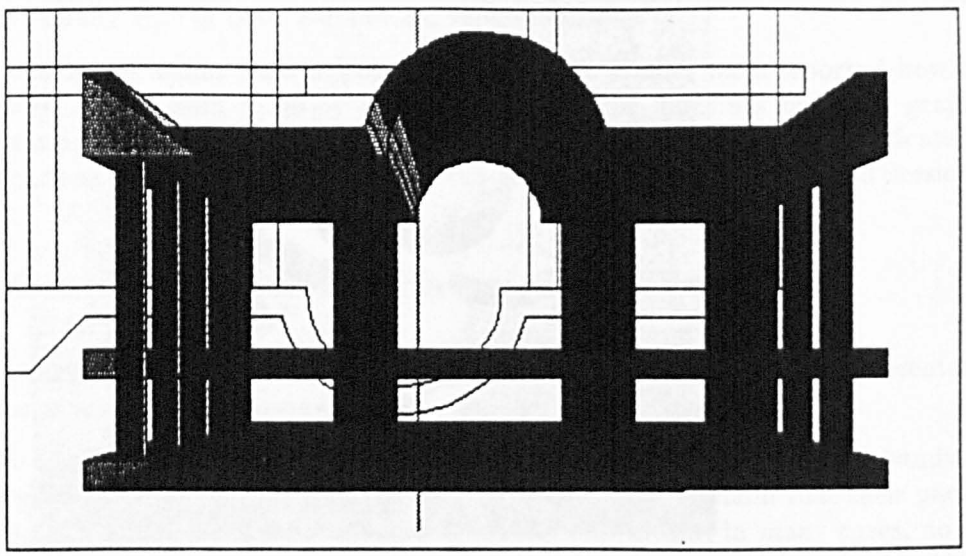


Figure 2.2: Graphical Representation Used in a Tool for CAD

objects or systems with which students or operators are being trained to work. Early systems again used 2-d, often iconic representations. For example Steamer [137] used 2-d schematic diagrams of a steam propulsion plant consisting of graphical representations of pipes, gauges, valves and pumps which could be animated to illustrate the behaviour of the plant under certain conditions of interest (see figure 2.3). The user could control operation of the simulated plant by clicking with a mouse on various objects in the display. Systems such as ThingLab [11] and LabVIEW [148] used similar schematic and iconic representations of electrical circuits, mechanical constructions and laboratory instruments which could be manipulated by the user via the mouse. More recent training systems often use much more realistic graphical representations. For example, the periscope simulator built by Marconi for the Dutch Navy [129] represents realistic 3-d images of ships and aircraft in surroundings which can be varied to simulate different times of day and different sea conditions.

Another class of tools which rely heavily on the use of sophisticated graphical representations are those for scientific visualisation. Molecular modelling systems have for some time used 3-d representations of molecular structures (see figure 2.4) in order to investigate and experiment with new arrangements [114, 53].

There has also been a proliferation of medical imaging systems [143, 1, 61] which allow the user to non-invasively view the interior of a human body thanks to the computerised display of 3-d data derived from various scanning techniques.

The kinds of system described in this section all use sophisticated graphical representations of real world objects of interest to the user. Representations used here are never as realistic as those supported by virtual reality environments: however accurate the representation may be, the user is always aware to some extent of looking at a picture on a screen rather than a real object. However, the use of detailed images still has the advantages described above concerning high bandwidth communication of information in a form readily understood due to its similarities with other forms of information about the real world which we regularly process. The challenge in this case is to find a metaphor

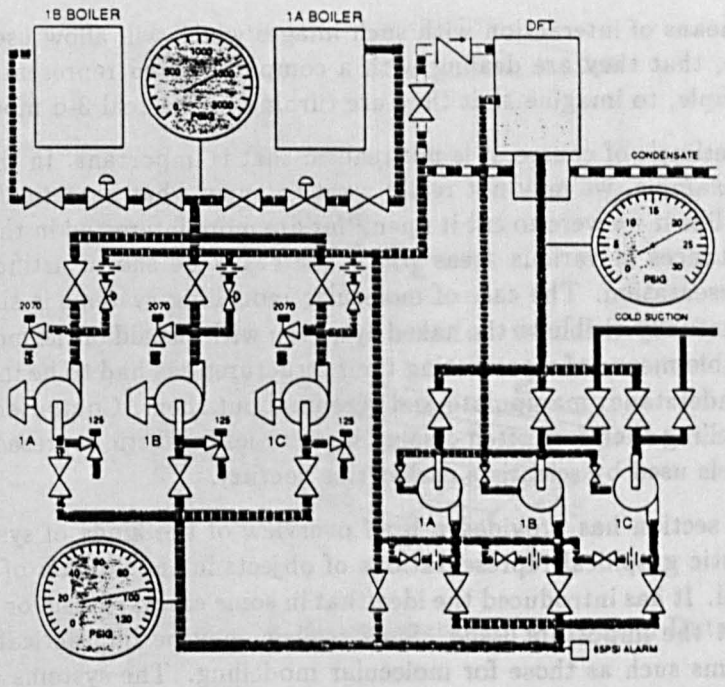


Figure 2.3: Graphical Representation Used in Steamer

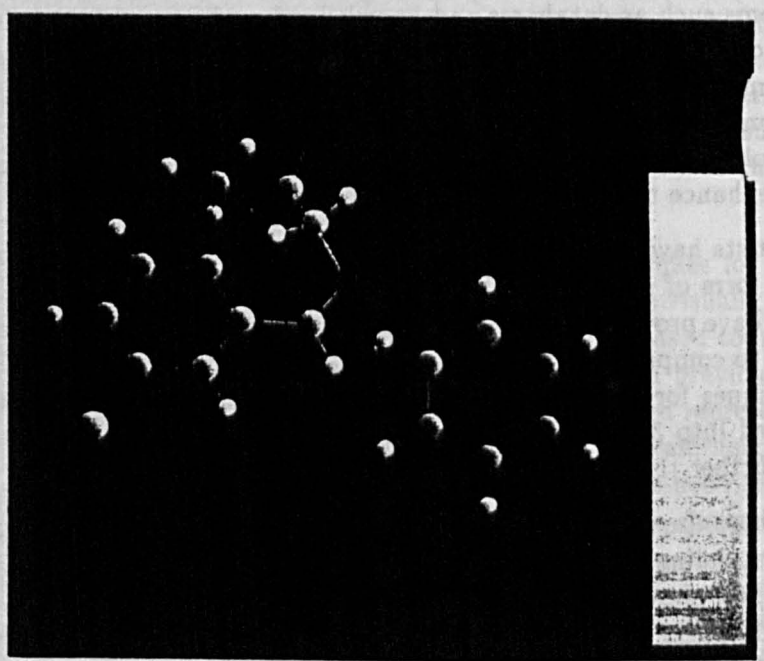


Figure 2.4: Graphical Representation Used in Molecular Modelling

or means of interaction with such images which will allow users to forget, as far as possible, that they are dealing with a computer-based representation, and allow them, for example, to imagine that they are turning over a real 3-d model in their hands.

Sometimes, of course, it is not realism that is important. In the case of medical imaging, for example, we may not really want to know what a particular part of the body would look like if we were to cut it open, but are more interested in the concentrations of various substances in various areas which can easily be shown artificially in a computer-based representation. The case of molecular modelling systems is similar. Since molecules are not actually visible to the naked eye, even with the aid of the most powerful microscopes, a suitable means of representing their structures has had to be invented in order to allow us to understand, manipulate and reason about them. Computer-based tools for molecular modelling therefore often display more or less realistic representations of the kinds of 3-d models used by scientists earlier this century.

This section has provided a brief overview of the kinds of system in which slightly less realistic graphical representations of objects in the domain of interest can be extremely useful. It has introduced the idea that in some cases, realism or accuracy of representation is not the important issue. Such realism may be impracticable or even undesirable in systems such as those for molecular modelling. The systems described in the following section are used in domains in which the objects or entities of interest are abstract, and for which there can be no such thing as a realistic representation. They raise a further, slightly different set of issues.

2.4.3 Representing the Abstract

Systems such as databases and knowledge based systems deal with abstract entities like data or knowledge structures. Programs are made up of abstract constructs and software systems consist of abstract components. Tools for the design and development of these systems must therefore find a way of representing this abstract information which is helpful to the user in supporting the necessary design and development tasks, and may also enhance understanding by making the abstract concrete.

Scientists have long been familiar with the idea of representing abstract data in the concrete form of graphs, pie charts, bar charts and histograms. These concrete representations have proved useful in conveying an appropriate understanding of the data, allowing users to compare, contrast, reason and infer in useful ways. Traditional 2-d paper-based techniques for representation have now been taken up by computer-based tools and developed into 3-d interactive versions (see figure 2.5) which can enhance understanding still further [119].

Are there then some concrete forms of representation which might similarly enhance our understanding of abstract software systems? As in the case of abstract scientific data described above, a user can have no a priori expectations about the way such systems should appear in physical terms. The aim, then, in developing a concrete representation should be simply to find one which is 'cognitively transparent' [20], that is, one which reveals or emphasises what the user needs to see or know in order to develop a productive mental model of the system. Of course, diagrammatic representations of software systems and programs have long been used in design, development and teaching. It therefore seems reasonable to assume that such representations have, to some extent at least, been seen to be useful. It is probably also reasonable to assume that most of today's software

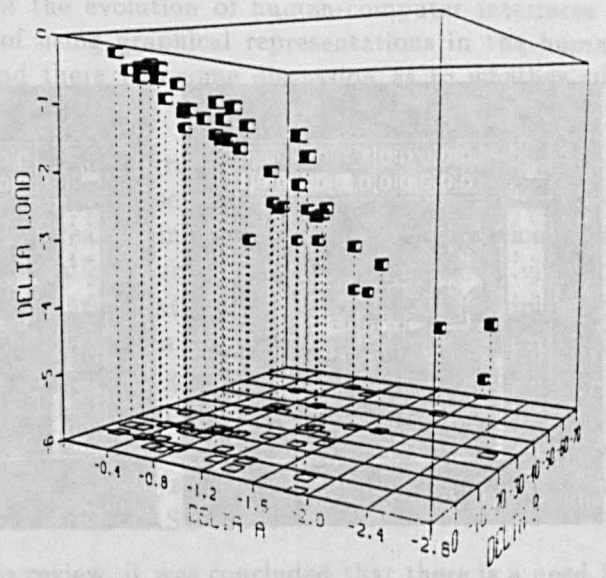


Figure 2.5: 3-Dimensional Representation of Numerical Data

practitioners will have had some experience of such diagrammatic representations, and may as a result have some beliefs or expectations about what a concrete representation of an abstract software system should be like.

Many computer-based tools for the design and development of software systems of various kinds have capitalised on these ideas. Chang [25], Ambler and Burnett [2] and Myers [97] all review a wide variety of visual programming environments, many of which draw on traditional ideas of flow charting (see, for example, figure 2.6). Yasdi [159], Bryce and Hull [21], Goldman *et al.* [65] and Czejdo *et al.* [33] all describe computer-based tools for developing and querying databases which use 2-d graphical representations similar to those developed for paper-based methods. Many CASE tools also now support exactly the techniques for diagrammatic representation of system specifications that were learnt using paper and ink.

Of course it is useful to draw on existing and well-proven techniques for diagrammatic representation of these abstract systems, and many authors of the visually oriented tools described above claim huge improvements in usability over text-based counterparts. But we should not be bound by such traditions. As we have seen in previous sections, today's technology allows a wide variety of exciting and sophisticated representations to be accessed and exploited with relative ease. As new kinds of programming languages are developed and software systems grow in size and complexity, the need for investigation of novel representations which can accommodate and harness such changes becomes ever more pressing.

There have been some interesting and innovative developments in this field over the last few years. Attempts have been made to find new and useful graphical representations for object-oriented languages, concurrent programming languages [118, 115] and fifth generation languages such as Prolog [34, 16, 47]. Ramanathan and Hartung [110] suggest an iconic representation for viewing databases which might enable users to cope with their rapidly increasing complexity, and Fairchild *et al.* [51] have developed the Tourist system to help teams of software designers cope with increasingly large and complex

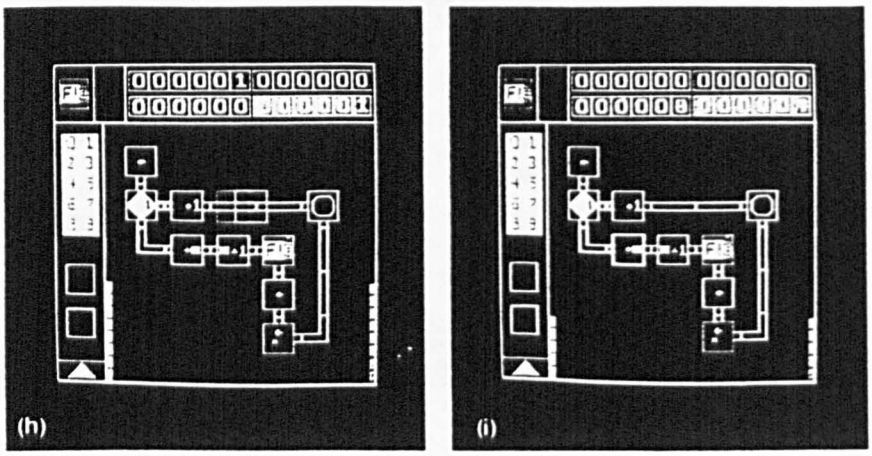


Figure 2.6: Graphical Representation Used in Pict

systems. The available technology is, however, still often under-exploited.

2.5 Conclusions

Graphical representations of one form or another are extremely common in the human-computer interfaces of today's computer systems. There has recently been a rapid growth of interest in the use of interfaces employing such representations for applications in many areas running on many different hardware platforms. But despite the large number of developments described, little work has apparently yet been done on determining which of the proposed new forms of representation actually provide real assistance to users of the software in question.

Graphics is not a panacea as many writers have cautioned [57, 25, 2, 106]. Before introducing a new form of interface or a new representation into a particular area of application, attempts should always be made to determine whether it will be of any real assistance to those working in that area. Basic research into the use of particular kinds of graphical representation in particular application areas is therefore needed in order to harness the current growth of interest.

Work described in this thesis aims to address this need for basic research by investigating the use of a particular form of graphical representation – that of 3-d ICDs – in the human-computer interface of tools used in a particular application area – that of knowledge engineering.

2.6 Summary

This chapter presented a review of literature relevant to the provision of graphical support for human-computer interaction.

A brief overview of the evolution of human-computer interfaces was presented. The potential benefits of using graphical representations in the human-computer interface were introduced and there was some discussion as to whether such benefits might be obtained from the use of 3-d as well as 2-d representations.

Some examples of the kinds of graphical representations used in various application areas were given. Examples were described in terms of their position on a continuum which reflected the degree of realism in the graphical representations used. At one end of the continuum were applications such as those making use of virtual reality techniques of which realism is an essential feature. Further along were applications using less realistic models or pictures of objects in the domain of interest such as CAD tools, computer-based training systems and programs for molecular modelling and medical imaging. At the opposite end of the spectrum were systems using graphical representations of abstract entities or relations in which considerations of realism or faithfulness of representation are simply not appropriate. Systems in this last category include visual programming environments, DBMS's and CASE tools.

On the basis of this review, it was concluded that there is a need for basic research into the use of particular kinds of graphical representation in particular areas of application.

Chapter 3

Graphical Support for Knowledge Engineering

3.1 Introduction

This chapter aims to further set the scene for work described in later chapters by presenting material relevant to the particular application area of interest - that of knowledge engineering. It begins by distinguishing the needs of the knowledge engineer from those of the knowledge-based system end user and those of the domain expert. Arguments for the use of graphical representations in the human-computer interface presented in chapter 2 are then re-examined in the context of the needs of the knowledge engineer. The final sections describe the way in which graphical support for knowledge engineering is currently provided, firstly in terms of the stages in the knowledge-based system development lifecycle at which support is provided by various tools, and secondly in terms of the particular forms of graphical representation used.

3.2 Knowledge Based System Human-Computer Interface Requirements

In designing the user interface of a knowledge-based system, the needs of three distinct groups of users must be considered. These three groups of users can be characterised as end users (those who will eventually use the completed system), knowledge engineers (those who will design, develop and maintain the system) and domain experts (who are the source of the knowledge embodied in the system).

Of course, the needs of these different groups of users often overlap to some extent. Individual users may often play more than one role with respect to a particular system. The domain expert may, for example, also be an end user as in the case of expert decision support systems. Users from different groups may also need to interact with the system via a common interface on certain occasions. The knowledge engineer and the domain expert may well need to discuss the validation of knowledge gathered using a single, jointly understood representation of that knowledge.

Despite these complications, consideration of the different roles users may play can often

be useful in shaping our thinking regarding the kinds of support which might be provided in the human-computer interface to a knowledge based system. The following paragraphs therefore present an introduction to the needs of users in each of the three roles identified.

3.2.1 The End User

The knowledge-based system end user is the intended user of the finished system. In the simplest case, the end user is interested only in answers. Information relating to the structures and mechanisms of the knowledge-based system is of no interest and should in this case not be provided: the role of the interface should be rather to provide the user with a natural means of expressing a query and a suitable answer to that query. In other cases, however, the intended users of the system might themselves be domain experts such as doctors, who wish to use the system as an aid to making complex decisions or diagnoses. In these cases, a system's users might want more than simple answers. In order to trust the system's decisions, they may wish to be able to check that such decisions have been reached in a reasonable way and that the knowledge on which they were based is correct. Under these circumstances, it becomes appropriate to provide the end user with a means of browsing over the system's representations of knowledge and obtaining 'how' or 'why' explanations which show how particular conclusions have been reached.

3.2.2 The Domain Expert

The distinctive role of the domain expert is that of providing the knowledge on which a system is based. The expert's main contact with the system has traditionally been during validation and verification in which it must be checked firstly that the appropriate knowledge has been collected and encoded into the system, and secondly that the system as a whole behaves in an appropriate way. In order to be certain that important knowledge has not been omitted or mis-represented, it is important that the system's knowledge be represented in a way that is easy and natural for the expert to understand. For validation, this may mean using a representation which is slightly different to some of the low-level 'coding oriented' forms which might be preferred by the knowledge engineer (though as we will see below, the roles and needs of domain experts and knowledge engineers are increasingly being seen to converge at this point). For verification, representations similar to those used in browsing and explanation facilities for the end user may be more appropriate.

3.2.3 The Knowledge Engineer

The work described in this thesis focuses on the needs of the knowledge engineer. Its aim is to investigate the use of a particular form of graphical representation in supporting what Eisenstadt *et al.* have called 'visual knowledge engineering' [48],

The knowledge engineer is the person who is responsible for designing, developing, debugging and maintaining a knowledge-based system. Many of these activities involve problems which are similar to those in software engineering. In both cases, the engineer needs powerful abstract representations of the relevant constructs and appropriate tools with which to manipulate them. In contrast with the naive end user, the knowledge

engineer is interested almost exclusively in abstract representations of the system's internal structures and mechanisms, and should therefore be provided with appropriate and efficient ways of viewing or browsing and editing such representations.

3.3 Arguments for the Use of Graphical Representations

In the following paragraphs, arguments presented in section 2.3 are re-examined in the context of knowledge engineering.

Arguments concerning increased enjoyment of interaction and memorability of graphical representations seem as likely to hold for the field of knowledge engineering as for any other, though it should be noted that the kinds of abstract graphical representations used here are not likely to be quite as memorable as pictures of more familiar real world objects.

The fact that graphical representations support high bandwidth communication is also likely to be useful here. Knowledge-based systems are often extremely complex. Graphical representations provide a compact and immediate way of expressing multi- dimensional information both about the system as a whole, and about individual components. The fact that overall gestalts can easily be grasped is likely to be useful in debugging [16], as is the potential to browse and scan with ease.

The question of whether the kinds of graphical representation commonly used in tools for knowledge engineering constitute a 'natural representation' for the structures involved is more difficult. Since there is no physical counterpart to a rule network or class taxonomy, it is difficult to say with any certainty what the knowledge engineer's 'natural idiom' [136] for consideration of such structures might be. Of course, individual knowledge engineers are likely to differ in the extent to which they prefer to visualise abstract knowledge structures. Differences are also likely to arise as a result of variations in knowledge engineering practice. As the field of knowledge-based systems is still relatively young, there are, as yet, few generally accepted methods for knowledge engineering of the kind which exist for software engineering, and individual practitioners are likely to want to be able to think and work in different ways. The possibility of providing universally comprehensible and satisfactory forms of graphical representation for the support of knowledge engineering might therefore seem somewhat remote.

The situation is not, however, all that bleak. Certain structures which are common in knowledge based systems, such as hierarchies or taxonomies, do occur elsewhere, and can be expressed in a (graphical) manner which is almost universally comprehensible. As described in section 2.3.2, there may be a number of advantages to the use of a graphical representation whether or not it is the one 'naturally' used by current practitioners. For example, new users may be more able to grasp and comprehend the meaning of a graphical representation than that of a textual equivalent on the first encounter. In any case, Kidd and Cooper [79] and Tsuji and Shortliffe [144] are optimistic that the kinds of graphical representations commonly used in tools for knowledge engineering are sufficiently similar to the users' own conceptual representations to be useful.

3.4 Tools Providing Graphical Support for Knowledge Engineering

A number of tools and environments claiming to provide support for various aspects of knowledge-based system development have appeared during the last few years. Some have taken a very focused approach in attempting to provide direct support for a small range of relatively well-defined tasks performed at particular points in the knowledge-based system development lifecycle. Examples of systems of this kind include specialised tools for knowledge acquisition such as AQUINAS [10] and systems for tracing rule executions such as ORBS [54], GEETREE [86] and Arboretum [95]. Other developers have adopted a much broader perspective with, for example, SemNet [52] acting as a general purpose vehicle for research into the possibilities of providing support of various kinds. Finally, some work has been done on providing integrated support for a number of knowledge engineering activities. VEGAN, KET and CGT [49] provide support for various approaches to knowledge acquisition and encoding and commercial tools for knowledge engineering such as KEE and NEXPERT provide for the development of various kinds of knowledge-based system which can then be seamlessly meshed together. Perhaps the most ambitious approach has been that taken by the developers of KEATS [48] the Knowledge Engineer's Assistant, whose various components are intended to provide integrated support right through the knowledge engineering lifecycle.

3.4.1 Knowledge Acquisition and Validation

The knowledge-based system lifecycle begins with knowledge acquisition. This phase traditionally consists of one or more iterations around a cycle of knowledge elicitation, initial problem conceptualisation or knowledge formalisation and knowledge validation. Both knowledge engineer and domain expert must be involved in the processes of knowledge elicitation and validation, and must be able to communicate accurately and efficiently regarding this joint activity. This raises a number of problems [10, 102]. The domain expert and the knowledge engineer have normally been trained in different disciplines and do not, at the outset, share any common understanding of the domain of interest. This in itself can make communication difficult. Furthermore, experts may often not be consciously aware of their own knowledge so that it is difficult to verbalise or describe it in precise terms.

Various methods and techniques for knowledge elicitation have been proposed [29] and found to be effective in the gathering and structuring of expert knowledge of various kinds. An important requirement at this stage is that the representations of knowledge used in discussions between domain experts and knowledge engineers must be easily understood by both parties. They must be sufficiently strong and versatile to provide a rigorous expression of many kinds of knowledge, but sufficiently natural and easy to use that both knowledge engineers and domain experts can work happily with them. Graphical representations based on the ideas of semantic nets or conceptual graphs [132] have been found to be particularly successful in this respect. For example, Nosek and Roth [102] report experimental findings which demonstrate the superiority of graphical semantic net representations over predicate logic in comprehension and conceptualisation tasks carried out with student 'experts'.

A number of tools provide graphical interfaces which support the perceived need for in-

creased participation of experts in the knowledge design process. NEXTRA, the knowledge acquisition component of the commercial system NEXPERT OBJECT provides a number of tools, many of which use graphical representations such as tables or maps with which the expert can interact directly, NEXPERT OBJECT code being automatically produced as a result [100]. GIS, the Graphical Interface System component of KEATS allows the knowledge engineer to construct 'concept maps' (see figure 3.1), which can easily be validated by domain experts, on the basis of online interview transcript annotations. COOL [30] supports the construction of conceptual graphs by knowledge engineers, and NEED [28] is an interface to a diagnostic expert system which allows knowledge engineers to develop graphical representations of deep causal knowledge for subsequent checking by the domain expert.

The developers of VEGAN, KET and CGT intend to take the involvement of domain experts in knowledge acquisition one step further. Their aim has been to build a suite of tools 'to aid experts, initially in collaboration with knowledge engineers, in creating their own knowledge bases' [77]. All three of these tools are intelligent graphical editors which allow users to input knowledge of different kinds directly into a central knowledge base. VEGAN allows users to build domain models in terms of associative or semantic nets; KET allows entry of procedural knowledge in the form of production rules using a combination of decision trees and associative networks; and CGT allows entry of knowledge in a conceptual graph form which complements the semantic nets supported by VEGAN. These tools have met with some success. Users have apparently enjoyed interacting with graphical components - often to the exclusion of other components without graphical interfaces. Work is apparently continuing with the aim of determining how an appropriate balance between the three tools might be struck.

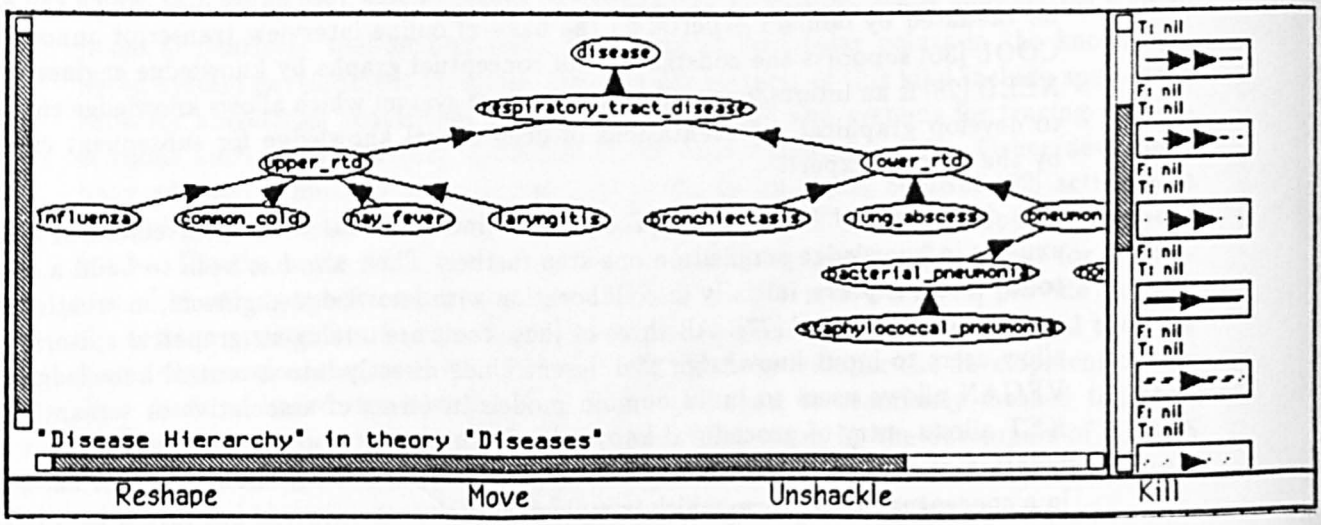
The tools described above are intended to provide domain-independent support mainly for knowledge engineers (though in one case for teams of knowledge engineers and experts) in gathering and structuring knowledge. Before moving on, it is worth noting that some success in using graphical techniques to support direct knowledge entry by domain experts has already been achieved in more restricted problem domains. For example, DETEKTR [60] allowed experts to enter knowledge into expert systems for finding faults in electronic devices simply by direct manipulation of a graphical representation of the appropriate devices. ESSA [70] supports expert input of diagnostic rules in a decision tree form which can then be translated into an appropriate language for use by the EXPERT shell. Finally, OPAL [96] is an intelligent graphical editor which allows experts to enter knowledge about chemotherapy protocols directly into the ONCOCIN knowledge base.

3.4.2 Design and Development

From the above discussion, we may now move on to the consideration of graphical support for the encoding or formalisation of knowledge by knowledge engineers.

Some of the tools described above (eg VEGAN, CGT and NEED) perform automatic translation of graphical representations specified by the user into particular forms of knowledge representation (such as frames or production rules or even Prolog) and in doing so perform limited amounts of syntactic checking. They can therefore be said to support knowledge encoding to some extent at least.

Other tools support the translation of graphical representations into a range of formalisms. These tools (such as KEE [124], NEXPERT [100] and KEATS [48]) aim to



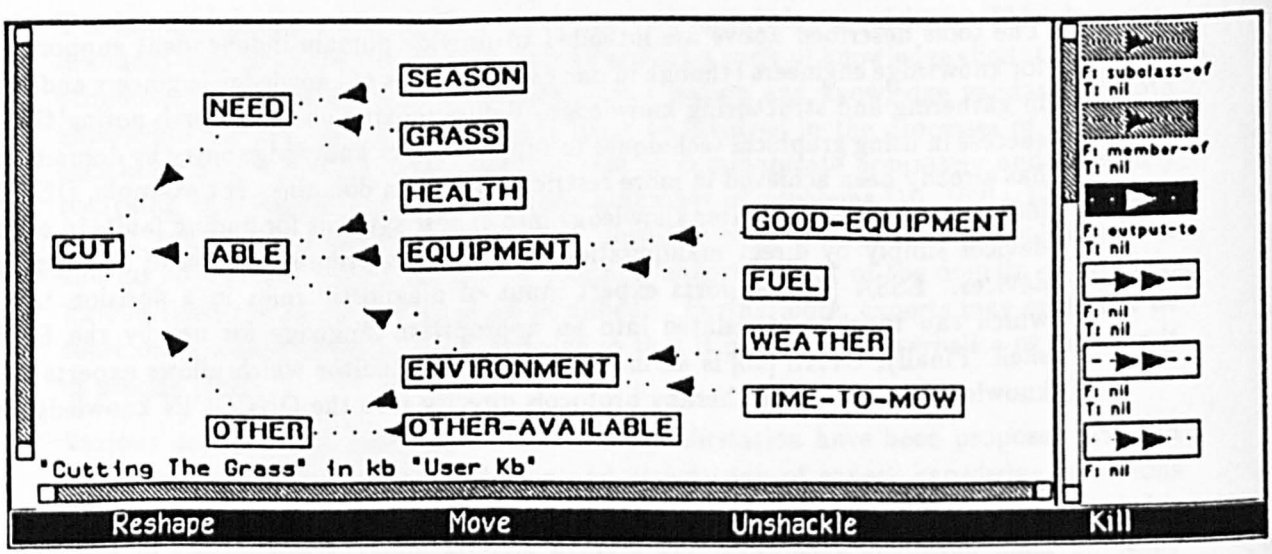


Figure 3.1: Concept Maps in GIS

guide the knowledge engineer in appropriate directions by allowing detailed problem conceptualisation and even producing skeletal code on the basis of associated graphical representations. The knowledge engineer may, however, often be left with considerable work to do on aspects of design, that is, on choosing appropriate mechanisms and techniques for formal representation and problem solving. Detailed programming is rarely done using visual programming tools as existing graphical representations of knowledge formalisms tend to lack the power and efficiency necessary to support this kind of task.

3.4.3 Debugging and Verification

Various tools provide support for the debugging and verification of knowledge-based systems. One form of verification which should be performed involves checking with the domain expert that the system behaves as expected. In this case, it is often most appropriate to allow the expert to view the system through the intended end user interface. Tools such as KEE [81] and LOOPS [63] support the construction of sophisticated graphical end-user interfaces with guages, meters and sliders which are often very appropriate for use in verification. Knowledge engineers may also use these interfaces for similar purposes.

A number of tools provide the facility for knowledge engineers to view a graphical trace of rule execution. Rule tracers such as ORBS [54], GEETREE [86] and Arboretum [95] typically use graphical representations of AND/OR trees for this purpose and KEE [81], NEXPERT [100] and Guidon-Watch [113] all support dynamic viewing of execution on the basis of an animated representation of the rule network. Finally, KEATS's TRI [48] provides a novel representation of forward-chaining rule execution involving a table of icons used to represent firings of particular rules in successive cycles of the inference engine.

Other tools provide more specific support for a variety of debugging activities. Most of the tools described above provide at least primitive support for viewing or browsing over graphical representations of large sections of the knowledge base. Some, such as VEGAN and KET provide various means of filtering the information displayed so that only nodes or links of particular kinds are shown. This reduces the visual complexity of the display which might otherwise be unmanageable in the case of large systems. Tools from the Open University provide other means of coping with complexity: for example, Brayshaw and Eisenstadt's Transparent Prolog Machine [16] supports four distinct methods involving variations in granularity of objects displayed or scale of view, compression of information into iconic form and abstraction away from detailed programming constructs towards a representation which is closer to the programmer's own conception of the system.

Finally, several of the tools described provide graphical support for obtaining 'how' and 'why' information and playing through 'what if' scenarios. VEGAN includes the facility to highlight particular paths through the network of rules which allows the knowledge engineer to determine 'why' a particular question was posed and was planned to have a similar facility to illustrate 'how' a particular conclusion was reached [77]. KEATS's Truth Maintenance Viewer provides the user with a graphical representation of the current contents of the belief set which allows knowledge engineers to determine the answers to 'how' and 'why' questions and to pose 'what if' queries by toggling individual nodes in the TMS tree into or out of the belief set in order to observe the effects.

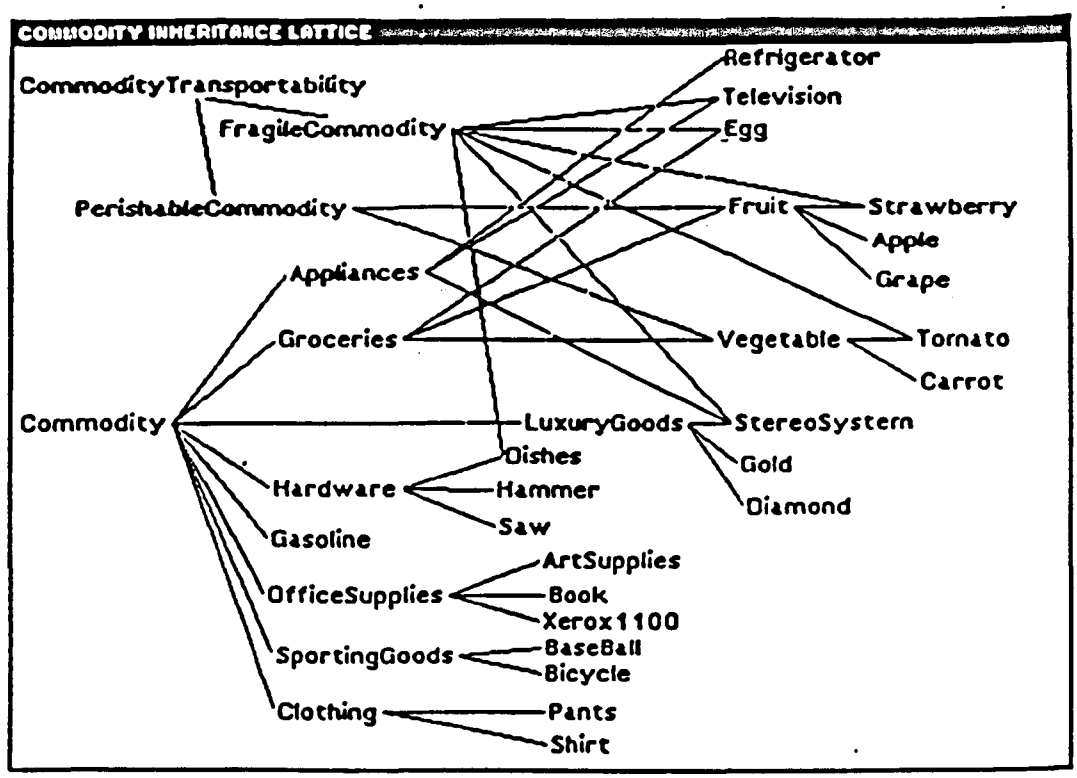


Figure 3.2: LOOPS class browser

3.5 Paradigms for Graphical Knowledge Representation

The various forms of graphical support for knowledge engineering described above are in general provided by some kind of graphical editor. In all cases except one (that of the Transparent Rule Interpreter [40]), the graphical formalisms or paradigms supported by such editors are based on the use of nodes (represented as points or boxes of various shapes and sizes) and links (or lines), usually with textual labels to identify individual elements. Typical examples are shown in figures 3.2 and 3.3. Node and link diagrams have been found to be highly appropriate for use in the expression of knowledge structures such as semantic nets, object or class hierarchies, and various frame-based networks, though representation of rule-based systems using such a notation has been said to require a little more ingenuity [52].

This section presents a brief description of important features of the graphical representations used in the tools for knowledge engineering described above. Representations will be considered firstly in terms of the graphical vocabulary on which they are based, secondly in terms of their use of animation, and finally in terms of their use of spatial coding or positional information.

3.5.1 Graphical Vocabulary

The term 'graphical vocabulary' here refers to the range of graphical symbols or primitives available for use in the construction of a graphical representation of a particular structure.

A fairly small vocabulary consisting of different sizes, shapes or colours of nodes and links is typically provided. Many of the systems described support some kind of graphical coding to allow viewers to make an immediate visual discrimination between nodes and

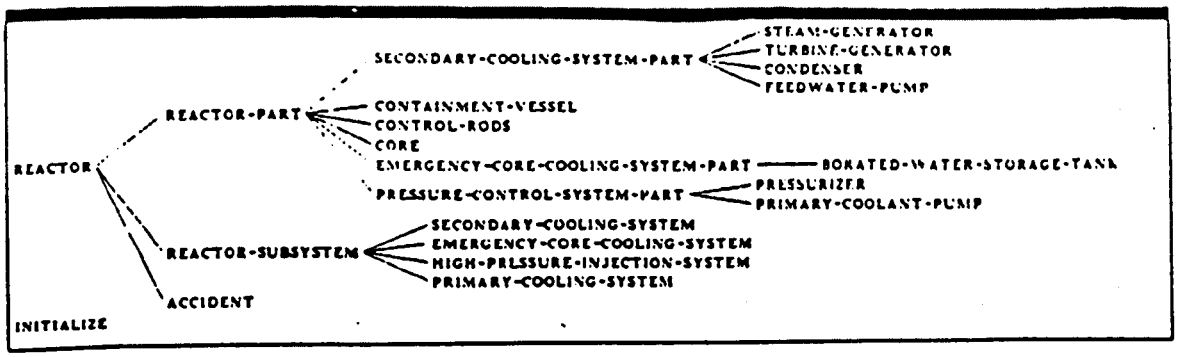


Figure 3.3: KEE Class Hierarchy

links representing different kinds of entities or relationships. For example, SemNet [52] uses links and nodes of different colours with bands of various thicknesses around nodes to describe the amount of information they contain. NEED [28] provides the user with five different shapes of node (rectangles, ellipses, double-edged ellipses, hexagons and rhomboids to represent actions, states, initial causes, diagnostic hypotheses and findings respectively) and six different types of link. GIS (part of KEATS [48]) provides the user with up to eight user-definable link types distinguished by line style (whether solid, dotted or dashed etc) and number of arrow heads.

Some systems also provide simple icons for the purpose of graphical coding. For example, Guidon-Watch [113] uses boxing and flashing of nodes labelled with text written in various font styles to illustrate the system's dynamic search strategy, as well as providing '+' and '-' symbols to indicate certainty factors. KEE [81] and NEXPERT [100] use simple icons (tick and cross in KEE - tick, query and spot in NEXPERT) as well as highlighted node boxes and different font styles to indicate the current state of the rule network during execution (see, for example, figure 3.4). Finally, TMV (also part of KEATS) uses graphical coding to illustrate the current state of beliefs monitored by KEATS's TMS showing nodes which are 'in' as white and nodes which are 'out' as black with nodes representing premises being shown with a small cross in the centre (see figure 3.5).

3.5.2 Animation

Some tools use animation to illustrate the changing states of the knowledge base during execution. For example, Guidon-Watch [113] displays dynamic information on top of a static graphical representation of a disease taxonomy in order to describe the system's search for a diagnosis. KEE [81] and NEXPERT [100] also provide the user with a dynamic view of rule execution using animation of the rule network representation. All three graphical components of KEATS [48] (GIS, TRI and TMV) can be made to dynamically reflect changes in the knowledge base giving an animated view of active values in a semantic net or structured situation model (GIS) as well as information about the current state of rule execution (from TRI) and the belief set (from TMV). Finally SemNet [52] experimented with the use of animation to provide a dynamic simulation of morphological analysis [127] or, more generally, to show the progress of any knowledge

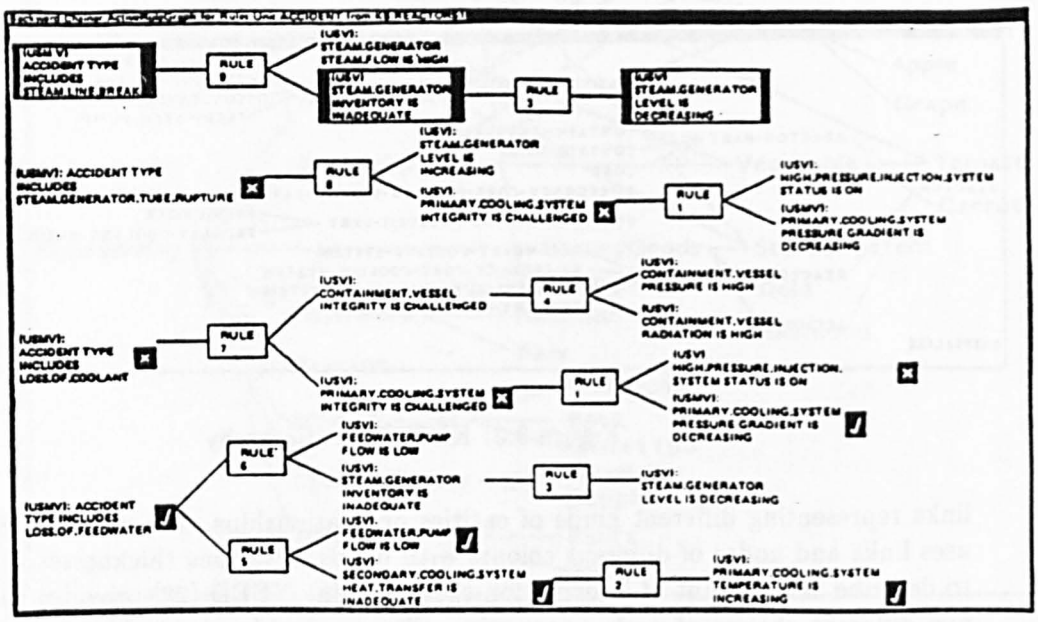


Figure 3.4: Iconic Representations Used in KEE

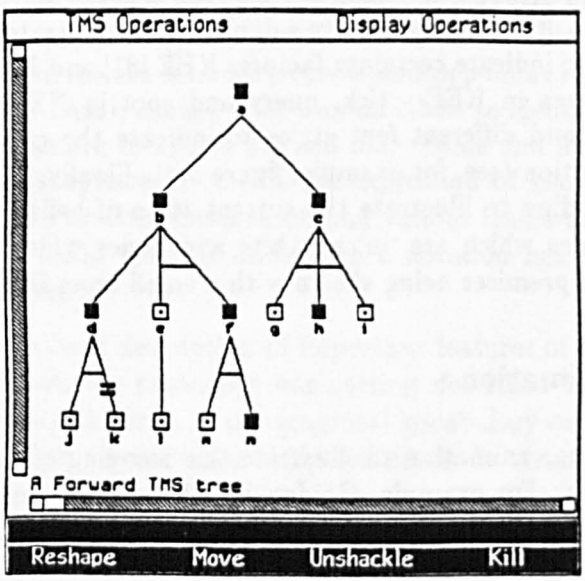


Figure 3.5: Iconic Representations Used in TMV

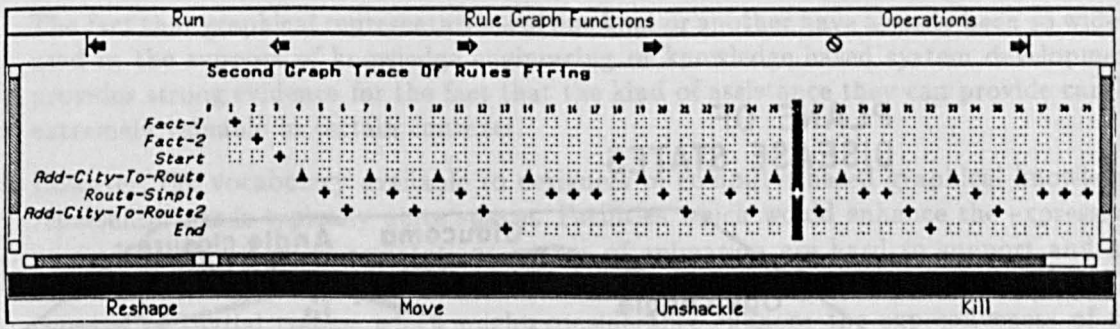


Figure 3.6: 2-Dimensional Representation Used in TRI

base execution by depicting a 'sprite' travelling along arcs between nodes as processing moves from one node to the next.

Facilities for animation of graphical representations are not, at present, commonly provided in tools for knowledge engineering. It is, of course, harder to provide support for animation than it is to provide a simple static vocabulary, and considerable processing power is often needed to support the use of dynamic images. There has so far been little research on the utility in knowledge engineering of animated graphical representations.

3.5.3 Spatial Coding

The potential for spatial coding or the portrayal of information about an object in terms of its position relative to other objects is also rarely exploited in any fixed way. Of the systems described, some simply allow complete freedom of layout, thereby leaving spatial coding of any kind to the user. Others provide layout algorithms which produce neat and aesthetic arrangements according to various criteria.

In representational schemes developed by users or explicitly supported by the provision of layout algorithms, a maximum of one dimension of spatial information is normally used, and this mainly in the redundant recoding of information about an element's position in a hierarchical structure which is already available from the display of graphical links. One notable exception here is TRI [48] whose tabular form exploits two dimensions (see figure 3.6). Some suggestions have also been made for the use of all 3 dimensions. A 3-dimensional representation consisting of hierarchies drawn on three separate planes has been suggested as an appropriate way of envisioning knowledge in the Casnet expert system [46]. A further suggestion made by Fairchild *et al.* [52] in a paper describing the SemNet project is that up to three classifications of sets of entities in a knowledge base may be used as the basis of 'mapping functions' which would define the positions of graphical representations of those entities in three-space.

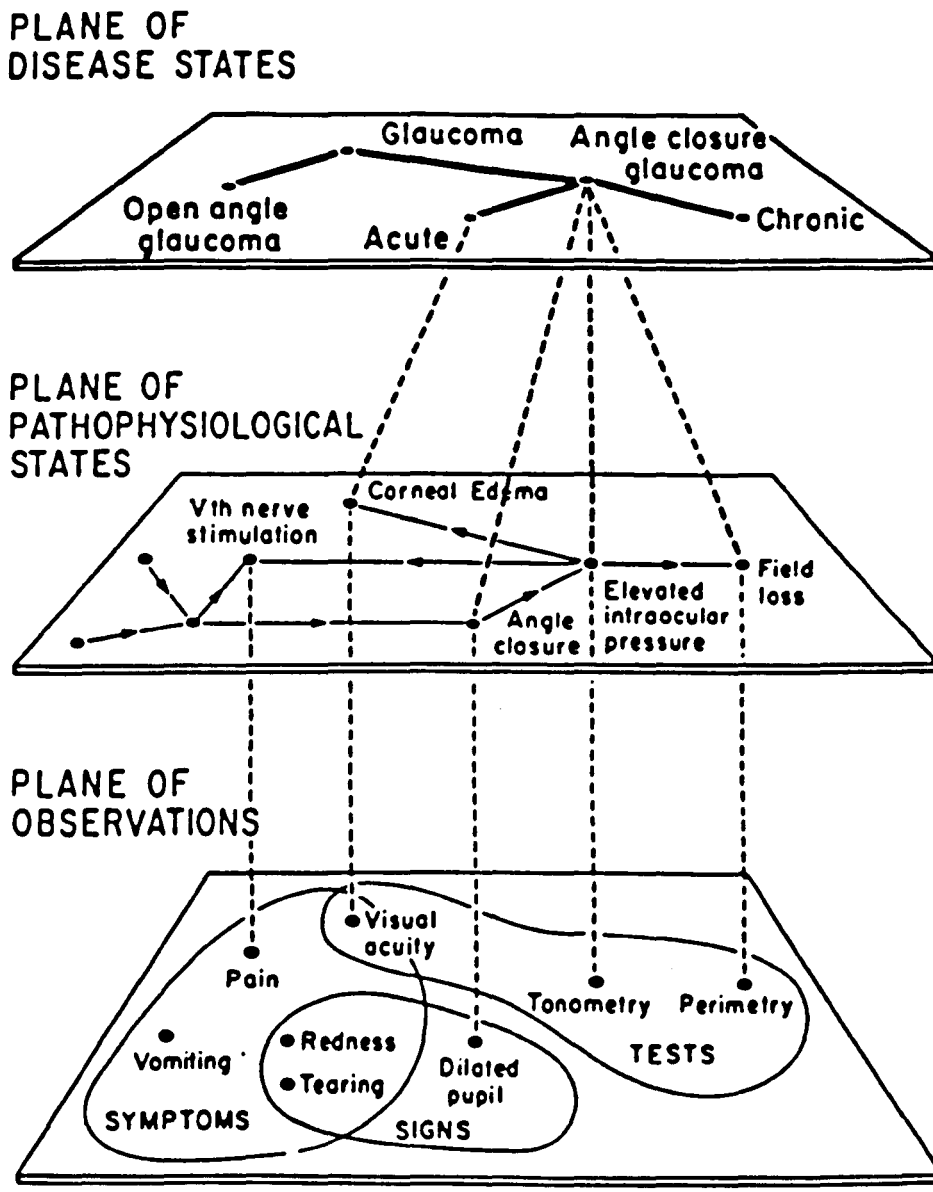


Figure 3.7: 3-Dimensional Representation Suggested for Use in the Casnet System

3.6 Conclusions

The fact that graphical representations of one kind or another have already been so widely used in the support of knowledge engineering or knowledge-based system development provides strong evidence for the fact that the kind of assistance they can provide can be extremely valuable in certain contexts.

However, the vocabulary available to designers of computer-based graphical knowledge representations is typically quite sparse. Facilities which would enhance the expressiveness of simple static representations by use of animation are hard to support and are therefore not commonly provided in tools for knowledge engineering. Facilities for the support of spatial coding which might considerably enhance the expressiveness of the simple representations commonly provided are relatively easy to provide but are also rarely exploited.

Since human information processing mechanisms are so easily able to understand and use spatial information, it seems that there may be considerable potential for spatially enriching the human-computer interface of tools for knowledge engineering at very little cost, simply by using such information in a meaningful way.

Work described in this thesis is aimed at investigating the ways in which the use of spatial information may be incorporated into graphical interfaces for knowledge engineering.

3.7 Summary

A review of literature relating specifically to the use of graphical interfaces for knowledge-based systems was presented. The needs of three groups of users characterised as end users, domain experts and knowledge engineers were briefly discussed.

Attention was focused on the needs of the knowledge engineer and arguments for the use of graphical representations introduced in chapter 2 were re-considered in this context.

A review of tools providing graphical support for various phases of the knowledge-based system development lifecycle followed. Stages considered were knowledge acquisition, validation, design, development, debugging and verification.

The final section described the graphical representations used in tools for knowledge engineering in terms of their use of graphical vocabulary, animation and spatial coding techniques.

It was noted that while the available vocabulary is typically quite sparse and animation can be difficult and expensive to support, possibilities for spatial coding are often under-exploited.

Part II

Depth in 3-d ICDs

The chapters in this part of the thesis describe and investigate the way in which depth is represented in 3-d ICDs. Chapter 4 provides an introduction to cues used in human depth perception and the way in which they are typically exploited in producing computer-generated representations of 3-d scenes and objects.

Chapter 5 describes a pilot experiment which investigated the use of rocking motion as a cue to depth in 3-d ICDs. Chapter 6 describes two further experiments carried out in order to investigate the effects of varying important parameters of the rocking motion on the effectiveness with which depth in 3-d ICDs could be perceived.

Chapter 4

Representation of Depth in 3-d ICDs

4.1 Introduction

Western artists have been familiar with methods of representing depth in scenes depicted on a flat canvas since the 15th century. More recently, cinematographers have had to explore methods suitable for use in films to be shown on flat screens. More recently still, workers in the field of computer graphics have been exploring a number of techniques for simulating depth in computer generated images, again to be displayed on a flat, 2-dimensional screen.

This chapter begins by explaining what cues to depth are thought to be used in human perception. Techniques used in computer graphics which exploit these cues in order to create the illusion of depth in computer generated images are then briefly reviewed. This sets the scene for a discussion of cues chosen for use in JIN, JINGLE and ICDEDIT, the tools developed at City for viewing and editing 3-d ICDs (see chapter 1).

4.2 Cues Used in Human Depth Perception

A number of theories regarding the nature of the cognitive mechanisms which underlie the perception of depth have been put forward. Of these, two are particularly well-known. Firstly, Gibson [62] has proposed that information about spatial relations can be directly derived from known structural and transformational invariants, such as texture and smoothness of motion. Marr [92] on the other hand proposed that depth and motion perception are based on inferences made about a series of frozen '2-and-a-half-d sketches'.

Whatever the mental processes involved in perception, there are certain visual cues which are known to be used in the extraction of information about depth from the environment. This section provides a brief introduction to the most commonly cited of these cues.

4.2.1 Monocular Cues (No Movement)

Monocular depth cues are those which can be appreciated using only one eye. Of these, some can be used only when there is motion, either of the eye or of the object under view, whereas others can be used when both eye and object are static. This section considers the latter.

4.2.1.1 Size

The size of an object *relative* to that of other objects can be used in making judgements about which of the objects is closer to the viewer. Thus, given an array of similar but differently-sized object images, objects with smaller images will be seen as being further away. *Familiar size* can also be a helpful cue: for example, if the image of a car appears to be the same size as that of a house, then it is reasonable to assume that the house is further away.

4.2.1.2 Occlusion

If the image of one object overlaps or partly obscures that of another, the object which is partially hidden will tend to be seen as the further away.

4.2.1.3 Height in Visual Field

Objects with images closer to the visual horizon will tend to be seen as more distant then objects whose images are further above or below the horizon.

4.2.1.4 Linear Perspective

The apparent convergence of straight lines in an image towards a point on the visual horizon provides an important cue to depth. For example, the image of a straight road stretching ahead of the viewer will occupy an increasingly small part of the retina as the viewer looks toward the horizon, and this decrease in size of retinal image is again interpreted as increase in depth or distance away of the object from the viewer.

4.2.1.5 Texture Gradient

An increase in the density of an object's surface texture will tend to be interpreted as an increase in distance of the surface from the viewer. Thus, in viewing a tiled floor, parts of an image in which tiles are packed more closely together will be seen as representing parts of the floor which are further away.

4.2.1.6 Atmospheric Perspective

Objects whose images are blurred or bluish tend to be thought of as further away than objects whose images are clear and colourful.

4.2.1.7 Shading

Variations in light and shade over the surface of an object can be interpreted as giving information about which parts stick out and which cave in. Shadows can also be used in understanding relative distances of objects from the light source.

4.2.1.8 Accommodation

In order to focus on objects at different distances, the muscles of the eye are used to change the shape of the lens. Thus, if the image of an object is focussed when the lens is thin, the object will tend to be perceived as being relatively distant on the basis of information provided by the muscles. Accommodation is thought to be a relatively weak cue to depth. Note that it is also a muscular cue and therefore of a slightly different nature to the previously described pictorial cues.

4.2.2 Monocular Cues (Movement)

Two of the most important cues to distance and depth are those which involve motion, either of the object or of the viewer. These two cues are explained below.

4.2.2.1 Motion Parallax

As the viewer moves, objects at different distances appear to move in different directions and at different speeds. Motion parallax is an extremely useful source of spatial information and can be used reliably, even in the absence of other cues.

4.2.2.2 Kinetic Depth Effect

The kinetic depth effect involves motion of the object rather than the viewer. It has been shown [150] that objects such as solid blocks, wire figures and straight rods which might look flat when stationary appear to be three-dimensional once they have been seen in motion. The kinetic depth effect is also extremely powerful and can be used quite reliably in the absence of other cues of the kinds described above.

4.2.3 Binocular Cues

Although monocular cues are in fact sufficient for the perception of depth, binocular cues form an important complementary source of information. The two main kinds of binocular cue are introduced below.

4.2.3.1 Binocular Parallax

Since the viewer's two eyes are separated in space, each sees a slightly different image of the objects within the visual field. Information about relative distances and depths can again be gathered from the difference in relative positions of the objects as seen from each of the eyes.

4.2.3.2 Convergence

In order to look at a nearby object, the eyes must converge, thereby giving a further muscular cue to the relative distance of the object from the viewer. Convergence though, like accommodation, is thought to be a relatively weak cue and operates mainly for objects close to the viewer.

4.3 Techniques for the Display of 3-d Computer Generated Images

4.3.1 Real and Virtual 3-d Images

4.3.1.1 Real 3-d Images

One possibility for the presentation of information about 3-d objects involves the use of 'real' 3-d images. There are a number of ways of generating such images using, for example, holographic techniques [72], vibrating varifocal mirrors [71] or multi-planar displays. These methods are, however, still uncommon and involve the use of specialised and expensive technology.

4.3.1.2 Virtual 3-d Images: Stereoscopy

A more common technique involves exploiting binocular parallax effects (see above) by presenting slightly different views of an object to each of an observer's eyes, thus creating a 'virtual 3-d image'. This technique, called stereoscopy, has been used to a limited extent for over 150 years. Early devices for stereoscopic viewing of 3-d images such as the Wheatstone and Brewster stereoscopes (developed in 1838 and 1849 respectively) used arrangements of mirrors and prisms to present specially prepared photographs. More recent versions involve the use of mechanical or, more recently still, liquid crystal shutters to rapidly alternate between left and right eye images which must then be viewed using special spectacles. A comprehensive review of stereoscopic techniques is given in Hodges and McAllister [72].

4.3.2 2-d Images with Illusory Depth

A much more common technique for the display of so-called 3-d images than those described above involves the use of sophisticated computer graphics to simulate the effects involved in monocular depth cuing as described above. Many different effects are used and there are many different methods and algorithms for achieving each of the effects. Research into the most efficient ways of producing realistic 3-d images is on-going in the field of computer graphics. Below is a list of the effects most commonly used, and the general principles behind each.

4.3.2.1 Perspective Projections

Perspective projections of objects and scenes can be generated on the basis of 3-d co-ordinates and allow observers to use cues such as relative size, height in visual field, linear perspective and even, to a certain extent, texture gradient. If objects have only a limited depth variation, the foreshortening provided by a 'realistic' perspective projection may not, on its own, provide an adequate cue to depth. The cue can, however, be strengthened by exaggerating the amount of perspective in the image in the manner of a wide-angled camera lens.

4.3.2.2 Hidden Line and Surface Removal

The removal of lines and surfaces which are to be thought of as being 'hidden' behind other objects or surfaces in the scene provides a powerful cue to depth on the basis of the occlusion effects described above. Methods for the removal of hidden lines and surfaces can, however, be quite computationally expensive.

4.3.2.3 Intensity Cuing

A further simple method for indicating depth in an image is to vary the intensity with which lines or surfaces are displayed according to their distance from the viewing position. Lines closer to the front are brighter or wider and the colour of surfaces nearer the front is made more intense than that of those behind. These methods rely on the atmospheric perspective effects described above and are relatively easy to implement in hardware. It has been suggested [71] that they work best with simple images. But with high resolution colour displays providing a wide range of intensities and colours, complex structures such as molecular models can be displayed very effectively using this technique.

4.3.2.4 Shading

Closely related to intensity cuing is the use of varying surface textures and lighting models which mimic the effects of various kinds of lighting in the real world. There are many techniques for modelling the effects of light, shade and surface texture, and new ones are appearing all the time. Most are computationally expensive, though efforts are being made to increase their efficiency [122] and an increasing proportion of the operations involved is often implemented in hardware.

4.3.2.5 Kinetic Depth Effect

Simulated motion of the object displayed allows the observer to use the kinetic depth effect in making relative depth judgements about it. Motion may be either system- or user-controlled. Early molecular modelling systems used small, system-controlled vibrations of structures displayed [114]. Later versions have given control of motion to the user, who can manipulate structures in virtual space using special graphics keypads, mice and sometimes joysticks [53]. The effect of apparent motion is achieved using animation. This involves the rapid generation of a series of transformed images and is again best achieved using specialised processing hardware.

4.4 Choice of Techniques for the Display of 3-d ICDs

This section discusses the choice of techniques for the display of 3-d ICDs in JIN, JINGLE and ICDEDIT. (Note that these choices were made by other members of the team working on the development of these tools as described in chapter 1. The author was not involved in making any such choices.)

4.4.1 Requirements for Realism

As suggested in chapter 2, the degree of realism required in the display of 3-d graphical images depends on the application and the tasks they are intended to support. At one end of the spectrum, graphical images used in CAD or in simulation applications (such as flight simulators for use in pilot training) may need to be extremely realistic in order to allow users to clearly imagine the real objects they represent. The need for realism in systems such as those for molecular modelling is not so great, as molecular structures are in any case impossible to see in any normal way: the need here is simply for accurate portrayal of the relative positioning of the structure's elements in 3-d space (see figure 2.4). At the opposite end of the spectrum are systems such as that described by Grotch [68] for the display of scientific data as a three-dimensional scatter plot. Attention to realism is simply not appropriate in such cases and highly metaphorical techniques such as the use of base-plane gridding and shadow projections can therefore be used (see figure 2.5).

Attention to realism or faithfulness of representation had also been felt to be inappropriate in the display of 3-d ICDs. Since ICDs were to be used mainly in the exposition of abstract, conceptual structures with no real-world physical structure or appearance, it was felt that metaphorical techniques could be used for the representation of properties or relative positions of elements within the structure.

4.4.2 Hardware Constraints

It had been decided early on in the project that it should be possible for 3-d ICDs to be generated and displayed using widely available general purpose workstation technology. It had been decided that there should be no reliance on sophisticated machinery for the production of stereographic images or holograms and efforts were directed toward producing strong monocular cues to depth which will be effective for images displayed on the flat screen of an ordinary monitor. The advantage of this was felt to be the increased portability of any software used in supporting such images.

Fortunately, Braunstein *et al.* [15] have shown that in situations such as that in which a supposedly 3-d image is viewed on a flat screen, where monocular and binocular cues to depth conflict, the monocular cues are often used in preference to the binocular. It should therefore be possible to provide monocular cues which suggest the existence of differences in depth in a displayed image and are strong enough to overcome the binocular cues telling us that we are actually looking at a flat screen.

Finally, it had been decided that the processing power used in the generation of 3-d ICDs should be kept to a minimum, as images would have to be generated and updated

at reasonable speeds using ordinary non-specialised processors which would also be under demand from other system components. This meant choosing simple but powerful techniques of the kind discussed above.

4.4.3 Depth Cues in JIN, JINGLE and ICDEDIT

Given the constraints and requirements described above, it is now possible to appreciate why some of the techniques described in section 4.3 were chosen for implementation in JIN, JINGLE and ICDEDIT while others were not.

One of the cues chosen for implementation in JIN, JINGLE and ICDEDIT was perspective. Perspective is almost universally used in 3-d computer graphics. The City tools provide the user with a standard perspective view, generated on the basis of a typical viewing point in front of the screen. They also provide the user with the capability to neutralise the effects of perspective by requesting a planar projection, or to increase the amount of perspective shown for a 'wide-angled lens' effect which can help clear up any ambiguities in networks with small variations in depth.

Hiding is also relatively simple to implement and cheap in terms of processing power. Hiding in the City tools is accomplished by ordering diagram elements in depth and drawing them in order from the furthest to the nearest so that representations of nearer objects simply overwrite those of objects which are further away.

Shading and intensity cues are not used as these require far more bits per pixel than were available in the hardware used.

The remaining cue to depth implemented in the tools at City involves a simulated rocking motion which makes use of the kinetic depth effect. The Whitechapel workstation on which JIN and JINGLE were implemented allows the user to create and store up to three separate images per window panel and to swap these rapidly to and from the screen. It is therefore possible to simulate motion by means of animation, swapping quickly between three slightly different perspective views. Figure 4.1 illustrates the swapping cycle used. A cycle begins with the presentation of a neutral view (marked in the figure as 'O') lasting for one unit of time; this is followed by one of the perspective views ('P') which is presented for two units of time; the view then returns to neutral ('O') for one unit before being replaced by the other perspective view ('N'). The cycle is repeated for as long as rocking motion is desired. This pattern was devised with the intention of simulating, as far as possible, a sinusoidal motion so that the network of nodes and links would appear to rock to and fro about a given axis [31]. JIN and JINGLE provide the user with the possibility of adjusting the angle (or degree), period (or speed) and axis of motion. The Sun 3/60 used for ICDEDIT provides rather different facilities to that of the Whitechapel. It has 8 bits per pixel, or 8 bit planes, but nothing corresponding directly to the Whitechapel's facility for multiple swappable images. The three swapped images are therefore each held in 2 bit planes. (Note that as a consequence of this, only 4 different colours are available for use at any one time.)

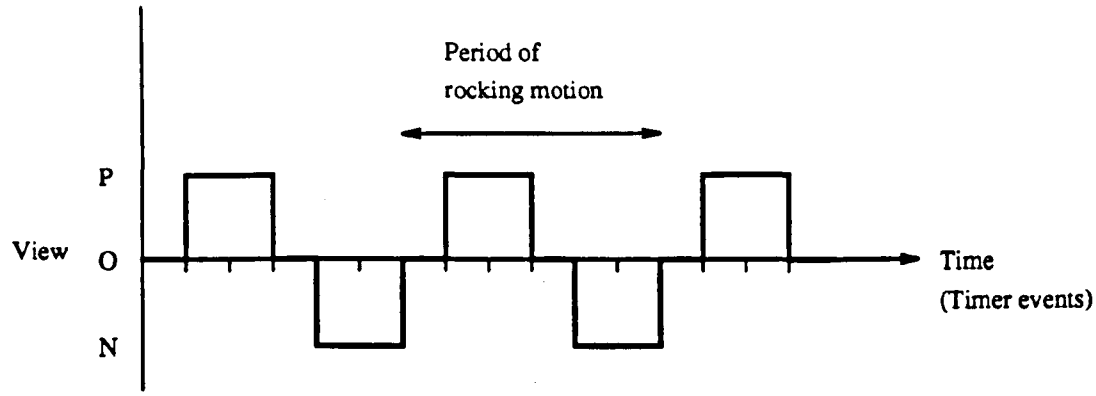


Figure 4.1: Rocking Motion in JIN

4.5 Summary

This chapter briefly described the cues used in human depth perception, and the way these cues are exploited in computer-generated representations of 3-d objects and scenes.

Depth cues implemented in tools developed at City for the display and manipulation of 3-d ICDs were described. Cues chosen for use in these tools were hiding, perspective and simulated rocking motion.

Chapter 5

Perception of Depth in 3-d ICDs: A Pilot Study

5.1 Introduction

This chapter describes a pilot experiment carried out in order to investigate whether users were able to make effective judgements about relative depths of components in 3-d ICDs presented using JIN, and in particular to determine whether the use of a relatively unusual cue to depth (that of simulated rocking motion) would enhance the effectiveness with which such judgements could be made.

A great deal of work on the effectiveness of various cues to depth has already been carried out in the field of perceptual experimental psychology. As suggested in the previous chapter, there are several theories as to how information about three-dimensional structures can be recovered from the two-dimensional images on the retina. Gibson [62] has proposed that information about spatial relations can be directly derived from known structural and transformational invariants, such as texture and smoothness of motion. Marr [92] has argued that information in the retinal image leads to the formation of a '2-and-a-half-d sketch' (defined in viewer-centred terms) which can then be used to create a three-dimensional model in terms of object-centred co-ordinates. Braunstein *et al.* [15] have argued for the existence of a second process by which object-centred shape information can be recovered directly from retinal information by using what has been called the 'kinetic depth effect' or 'structure from motion'. Prazdny [108] also argues for the existence of a process of this kind and suggests that it might involve the use of long range motion detection capabilities.

It is this work on what has been called the kinetic depth effect which is most relevant here. One of the first studies to be performed in the area was that of Wallach and O'Connell [150] who defined the kinetic depth effect as 'the perception of 3-dimensional structures from 2-dimensional motions'. Many studies have since been conducted to investigate the phenomenon in greater detail. Rogers and Graham [117] found that the motion parallax resulting from 2-d motions of either an object or the observer could be used to form a reliable, consistent and unambiguous impression of relative depth, even in the absence of other cues. Lappin *et al.* [84] found that the presentation of just two frames or images of a polar projection of a dot-covered sphere were sufficient for the detection of 3-d structure. Braunstein and Andersen [13] found that the simple use of velocity gradients to simulate

the motion of two intersecting planes was also enough to allow accurate judgements about relative depth to be made. Braunstein *et al.* [14], however, found that the use of rotation alone with parallel projections of 3-d objects provided only ambiguous information about depth, before the introduction of occlusion as a further cue which then enabled accurate depth judgements to be made.

From the results of these studies, it can be seen that humans are very adept at making judgements about three-dimensional structure using only the sparsest of cues. There are, however, situations in which such cues are ambiguous or conflict.

The investigation described below attempts to determine whether the cues to depth provided in JIN for the representation of 3-d ICDs are successful in supporting the reliable perception of relative depths and whether the monocular depth cues of perspective, occlusion and relative motion can overcome conflicting binocular cues as described in section 4.4.2 above. The emphasis is on determining the effectiveness of rocking motion, that is, of the simulated kinetic depth effect. The main aim is to discover whether rocking motion as implemented in JIN is a useful cue to depth over and above the cues of perspective and hiding which are more commonly provided.

The experiment used in conducting this investigation will now be described in detail.

5.1.1 Hypotheses

The major experimental hypothesis was as follows:

H_1: The use of simulated rocking motion will enhance the effectiveness with which judgements about the relative depths of components within 3-d ICDs of the kind supported in JIN can be made

It was judged that effectiveness could, in this context, reasonably be defined in terms of speed, accuracy and subjective feelings of ease and confidence. The main hypothesis could thus be broken down into four more readily testable parts as follows:

H_{1a}: The use of simulated rocking motion will significantly increase the speed with which judgements about the relative depths of components in 3-d ICDs of the kind supported by JIN can be made.

H_{1b}: The use of simulated rocking motion will significantly increase the accuracy with which judgements about the relative depths of components in 3-d ICDs of the kind supported by JIN can be made.

H_{1c}: The use of simulated rocking motion will significantly increase the ease with which judgements about the relative depths of components in 3-d ICDs of the kind supported by JIN can be made.

H_{1d}: The use of simulated rocking motion will significantly increase the confidence with which judgements about the relative depths of components in 3-d ICDs of the kind supported by JIN can be made.

Note that this experiment considers only judgements made in relative, object-centred terms. In the kind of use envisaged for 3-d ICDs, such as that of representing knowledge structures for knowledge-based systems, judgements about 'actual' or viewer-centred depth are not so likely to be needed. Positional information is in this case most likely to be used to represent relative values of significant entity attributes, or to show some

ordering on the entities represented. In this situation, viewers need only to judge which components of a diagram or structure seem further away than others, and need not have an accurate idea of how far away the structure as a whole is intended to seem.

5.1.2 Choice of Task

It was decided that these hypotheses could best be tested by looking at performance of a forced-choice depth discrimination task in which subjects were asked to decide which of two highlighted components in a 3-d ICD appeared to be closer to the front.

Performance of this task was timed, and response times were collected and analysed in order to assess support for hypothesis H_{1a}. Support for hypothesis H_{1b} was assessed on the basis of error rates for the experiment as a whole. Data from questionnaires completed by subjects at the end of the experiment was used to assess support for hypotheses H_{1c} and H_{1d}.

5.1.3 Choice of Stimuli

Most of the studies listed above used variants on the random dot technique similar to that described by Julesz [75]. This involves the simulation of solid surfaces using random dot coverage and allows the controlled investigation of individual depth cues. This technique is, however, not appropriate to an investigation of the affects of simulated rocking motion in node and link diagrams in which stimuli of a different kind are required.

As described above, the purpose of this investigation was to determine the effectiveness of depth cues implemented in JIN with a view to using 3-d ICDs as the basis for an interface for knowledge engineering. It was therefore thought more appropriate to use stimuli representative of those which might be used in real applications in the area of interest. The basic structure chosen was intended to imitate the layered structure suggested for use in the Casnet expert system (see figure 3.7). More details of the exact composition of stimuli are given below.

5.2 Method

5.2.1 Design

A repeated measures design was used so that each subject performed the experimental task with both moving and still stimuli. Each subject performed a total of 5 blocks of 24 trials.

For each subject, the first block of trials was used for practice. Each of the remaining blocks of experimental trials was split into two halves: the first half of the block consisting of trials under one experimental condition and the second half under the other. The first two trials of each half block were also used as practice trials and data from these was discarded. The order in which experimental conditions were presented was counter-balanced across subjects.

Note that the use of highlighting was controlled in a pseudo-random manner so that for each subject, nodes highlighted in one way were in front on half the trials and nodes highlighted in the other way were in front on the other half of the trials.

5.2.2 Apparatus

5.2.2.1 Hardware

The experiment was carried out on 4 Whitechapel workstations running JIN (see above). Monochrome monitors were used and user input was made via the keyboard. The 'b' and 'n' keys on the keyboard were labelled black and grey to correspond to the shading of nodes used in the experimental task (see below). The allocation of the shaded labels to the 'b' and 'n' keys was counterbalanced across subjects.

5.2.2.2 Stimuli

Five different ICDs were used for the five different blocks of trials. One of these is shown in figure 5.1, the rest can be found in appendix A. All ICDs were, roughly speaking, of the same structural type and each consisted of 30 nodes and 45 links. Each network was made up of 3 layers, each containing 10 nodes.

Stimuli were constructed by first placing nodes randomly within layers and then doing a small amount of manual editing to produce the kind of good layout which could be created automatically by a more advanced tool. Links between nodes (both within and between layers) were also added manually.

5.2.2.3 Motion Type

The same type of motion was used for all trials. ICDs were rocked at a rate of 1.25 cycles per second with an amplitude of 2 degrees about the x axis (corresponding to a maximum displacement of approximately 55 pixels across approximately 3.5 degrees of visual angle). This type of motion was chosen on the basis of informal feedback from various system users as being one which was reasonably comfortable to work with.

5.2.3 Procedure

Subjects were told that they would see a number of ICDs which they were asked to interpret as 3-d networks of nodes and links. They were told that pairs of nodes within the network would be shaded, one grey and one black, and that their task would be to indicate which of the two shaded nodes appeared to them to be closer to the front of the network by pressing the correspondingly labelled key on the keyboard.

Subjects were asked to perform the first block of practice trials as quickly and accurately as possible. In cases where they found it impossible to tell which of the two nodes was nearer the front, they were asked to guess. Subjects were then given the opportunity to ask any questions about the experimental task. Once the experimental task requirements were clear, they were asked to begin the first block of experimental trials. They were reminded that they should work as quickly and accurately as possible.

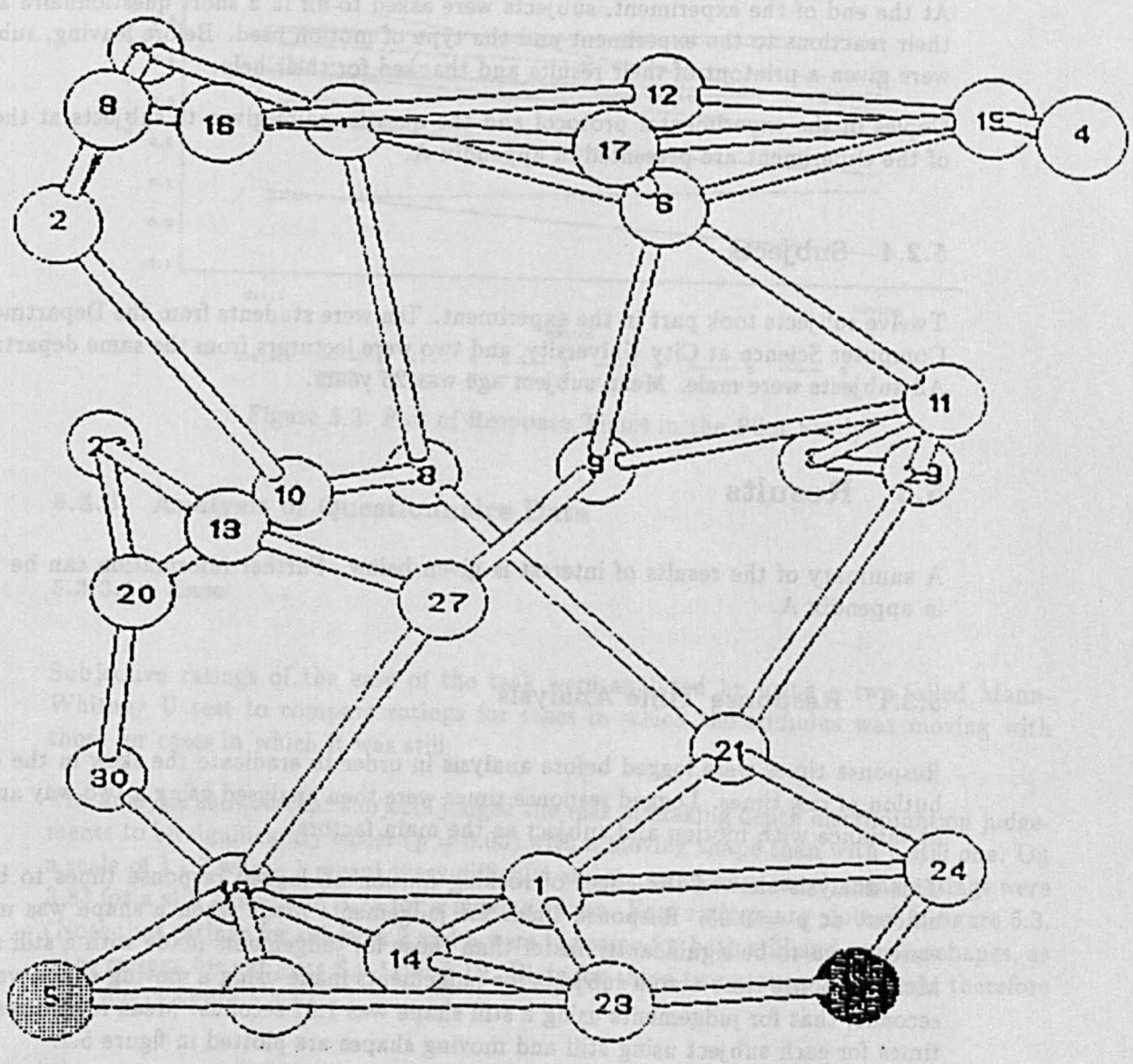


Figure 5.1: One of the Stimuli Used in the Pilot Study

There was a period of approximately 2 seconds between trials and a buzzer was sounded 1 second before every new pair of nodes was shaded. Subjects were allowed breaks between blocks of experimental trials and were able to initialise new blocks by pressing the space bar as and when they were ready to continue.

At the end of the experiment, subjects were asked to fill in a short questionnaire about their reactions to the experiment and the type of motion used. Before leaving, subjects were given a printout of their results and thanked for their help.

Copies of the experimental protocol and the questionnaire given to subjects at the end of the experiment are presented in appendix A.

5.2.4 Subjects

Twelve subjects took part in the experiment. Ten were students from the Department of Computer Science at City University, and two were lecturers from the same department. All subjects were male. Mean subject age was 26 years.

5.3 Results

A summary of the results of interest is given below. Further information can be found in appendix A.

5.3.1 Response Time Analysis

Response times were logged before analysis in order to eradicate the skew in the distribution of raw times. Logged response times were then analysed using a two-way analysis of variance with motion and subject as the main factors.

This analysis showed the effect of rocking motion on logged response times to be significant at $p = 0.05$. Response times for judgements made when a shape was moving were shown to be significantly faster than those for judgements made with a still shape. Mean response time across subjects for judgements made using a moving shape was 1.68 seconds; that for judgements using a still shape was 1.97 seconds. Mean logged response times for each subject using still and moving shapes are plotted in figure 5.2.

The subject factor was also significant, indicating that some subjects responded consistently faster than others using both moving and still shapes.

5.3.2 Analysis of Error Rates

Error rates for judgements made when the stimulus was moving and when it was still were compared using a two-tailed matched samples t test with $p = 0.05$.

Error rates were not found to be significantly affected by the motion of the shape. Mean percentages of incorrect judgements across subjects were 11.7 % for a moving shape and 12.9 % for a still shape.

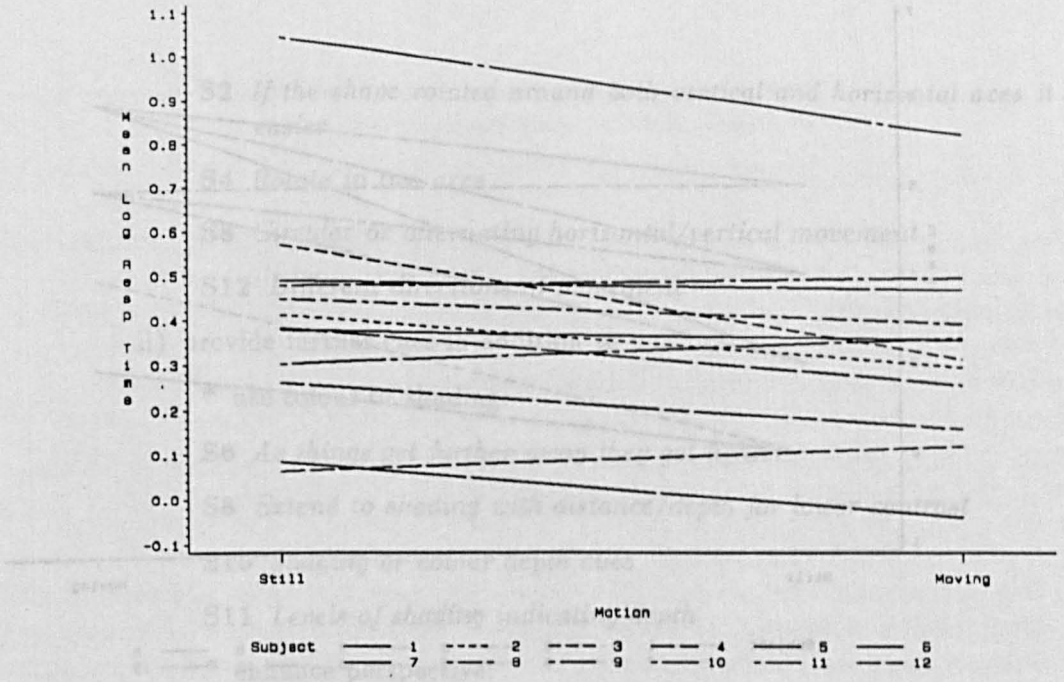


Figure 5.2: Plot of Response Times in the Pilot Study

5.3.3 Analysis of Questionnaire Data

5.3.3.1 Ease

Subjective ratings of the ease of the task were analysed by using a two-tailed Mann-Whitney U test to compare ratings for cases in which the stimulus was moving with those for cases in which it was still.

The analysis showed that subjects judged the task of making depth discrimination judgements to be significantly easier ($p = 0.05$) with a moving shape than with a still one. On a scale of 1 - 7 where 1 meant 'very difficult' and 7 meant 'very easy', mean ratings were 3.83 for a still shape and 5.25 for a moving shape. Ease ratings are plotted in figure 5.3. (Note that ratings for subjects 2 and 5 were the same for both still and moving shapes, as were ratings for subjects 8,9,11 and 12. Plots for these two groups of subjects therefore coincide in figure 5.3.)

5.3.3.2 Confidence

Subjective ratings of the confidence in task performance were analysed by using a two-tailed Mann-Whitney U test to compare ratings for cases in which the stimulus was moving with those for cases in which it was still.

This analysis showed no significant difference in confidence between judgements made using moving and still shapes. On a scale of 1 - 7 where 1 meant 'very unconfident' and 7 meant 'very confident', mean ratings were 3.75 for a still shape and 4.83 for a moving shape.

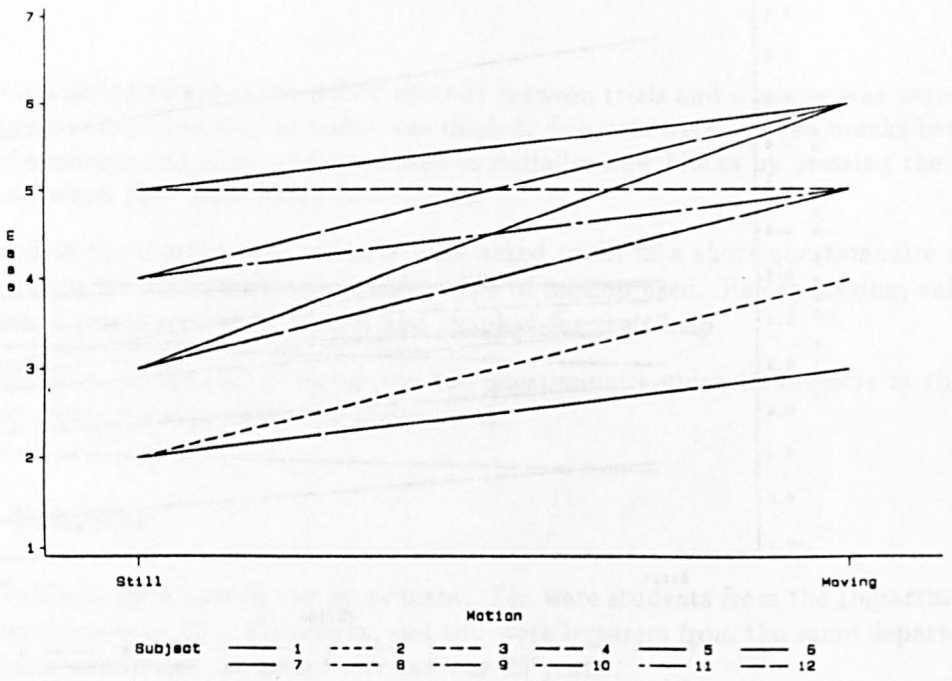


Figure 5.3: Plot of Ease Scores in the Pilot Study

5.3.3.3 Comfort

On a scale of 1 - 7 where 1 meant 'very uncomfortable' and 7 meant 'very comfortable' with the motion of the moving shapes, the median and modal rating was 5. Most subjects felt at least fairly comfortable with the type of motion used.

5.3.3.4 Preferred Angle of Motion

On a scale of 1 - 7 where 1 meant that a subject would have liked the shapes to move 'much less distance', 4 meant 'same distance' and 7 meant 'much greater distance', the median and modal rating was 5. Most subjects were reasonably happy with the angle of motion used but would have liked it to be a little greater than the 2 degrees amplitude used if anything.

5.3.3.5 Preferred Period of Motion

On a scale of 1 - 7 where 1 meant that a subject would have liked the shapes to move 'much slower', 4 meant 'same speed' and 7 meant 'much faster', the median and modal rating was 4. Most subjects seem to have been happy with the period of motion used.

5.3.3.6 Comments

Comments made by subjects in response to the two general questions at the end of the questionnaire were as follows.

Q8 Can you suggest any other changes which would make it easier to work with the moving shape?

i) provide different type(s) of movement:

S2 *If the shape rotated around both vertical and horizontal axes it would be easier*

S4 *Rotate in two axes*

S8 *Circular or alternating horizontal/vertical movement*

S12 *Different directions of movement*

ii) provide further cues in addition to motion:

* use colour or shading:

S6 *As things get further away they get lighter*

S8 *Extend to shading with distance/depth for lower contrast*

S10 *Shading or colour depth cues*

S11 *Levels of shading indicating depth*

* enhance perspective:

S3 *Distorted/increased depth change ie size of spheres might be (dramatically) reduced 'into' screen*

S8 *Could exaggerate perspective*

S11 *More perspective*

* provide stereo:

S10 *Stereo*

* use more realistic images:

S7 *Greater 3-d realism - especially in not having lengths and sizes rounded to integer pixel units*

* provide more context:

S6 *Some form of background (floor, walls etc)*

Q9 Is there anything in particular that strikes you about the difference between working with a still shape and working with a moving shape?

i) overall structure and relative depth differences are easier to grasp from a moving shape:

S10 *the still shape requires more concentration compared to the moving shape which is more intuitive*

S11 *The moving shape showed hidden [structures] and I was able to picture the depth of the structure more easily*

S12 *More depth can be seen when the shape is moving - easier to identify which is closer*

ii) movement is most useful in the case of nodes closely spaced in the $x - y$ plane:

S3 *It is more difficult to tell depth when two spheres are close on the screen with the still image*

S4 *The moving shape was easier with nodes which were close but just as hard when nodes were far away from one another*

iii) movement distracts:

S6 *Hard to choose between nodes lying near the vertical/horizontal 'eye-level' when shape is moving and nodes cross this plane*

5.3.4 Further Analysis of Response Times

Data about response times collected in the experiment described above were also used in informal investigations into the kinds of stimuli for which the effects of rocking motion seemed to be most marked.

In these investigations, subsets of the data corresponding to sets of trials in which the target pair of nodes exhibited certain characteristics of interest were extracted and analysed using separate analysis of variance tests. Subsets of data were collected for trials in which:

- target pairs were greater than a certain distance apart in depth
- target pairs were not directly linked
- target pairs were not indirectly linked (ie were not linked via a single other node)

Tests on subsets of data of the first kind tended to suggest that the strength of the effect of rocking motion was strongest for pairs which were around 35 pixels (approximately 1cm) apart in depth. The effect appeared to be weaker for pairs which were either much closer or much further apart in depth.

The strength of the effect of rocking motion for a subset of the data in which target pairs were not directly linked was similar to that for the data as a whole. In both cases, the use of rocking motion appeared to facilitate rapid task performance. However, the effect of rocking motion for a subset of the data in which target pairs were not directly linked was considerably weaker. This suggested that the use of rocking motion might be most effective in cases in which pairs of nodes whose depths are to be distinguished are indirectly linked.

It should at this point be remembered that no real weight can be attached to the above suggestions owing to the informal basis on which they have been made. A further series of experiments would need to be conducted to investigate such suggestions in a properly controlled manner before any conclusions such as the above could be safely drawn.

5.4 Conclusions

The results of this pilot experiment provide statistically significant evidence that the use of simulated rocking motion can increase both the speed and perceived ease with which subjects are able to make judgements about the relative depths of nodes in a 3-d ICD of the kind supported in JIN. Neither the accuracy of such judgements, nor the confidence with which they were made were, however, found to be greatly affected.

5.5 Discussion

5.5.1 Support for Experimental Hypotheses

It is possible that the effect of motion on response times was caused at least partly by an added sense of urgency felt when the stimulus was moving. This would mean that the benefit of rocking motion might appear unrealistically high. On the other hand though, the fact that JIN was an early prototype, whose implementation of simulated rocking motion was somewhat rough and ready (notably in its use of positional aliasing which sometimes caused motion to appear rather irregular), means that the observed effects of rocking motion may have been smaller than would have been possible with a more sophisticated implementation. In any case, it should be borne in mind that the actual figures or numbers of seconds saved by the use of rocking motion are not themselves important. Running the experiment with a slightly different version of JIN, or on a different type of workstation, or even with a different set of stimuli would probably have produced a different margin between the mean response times for still and moving shapes. What is important is that the difference in response times may indicate a difference in the cognitive effort required to make judgements about the relative depths of components in still and moving stimuli. The results of the experiment suggest that the effort required to make such judgements is lower for structures undergoing simulated rocking motion than it is for still ones. It is therefore suggested that facilities supporting the use of such motion should continue to be provided in future versions of tools for viewing or editing 3-d ICDs.

The fact that error rates were not significantly affected probably reflects the relatively high accuracy of judgements made by all subjects under both conditions (see appendix A for a table of error rates), and the fact that the number of data points available for use in the statistical test was relatively small (12 values for each of the two conditions of interest). The question of error rates will be further discussed at the end of the following chapter.

It should be remarked that the data on ease and confidence gathered from the questionnaires may have been biased, as subjects may have responded in a way which they felt might please the experimenter. However, the experimenter had made every effort to minimise this effect during the experiment, and the fact that one measure (that of ease) supports the use of motion, while the other (that of confidence) does not suggests that subjects felt able to respond honestly.

Comments made in response to questions 8 and 9 provide valuable further information on the effectiveness of simulated rocking motion as a cue to depth in 3-d ICDs. Subjects made a number of interesting suggestions regarding different types of motion and different cues which might be supported in future tools (see section 6.3.5). Responses to question 9, regarding the differences between making judgements about moving and still stimuli suggested that motion made it easier for some subjects to perceive both the overall structure of the diagram, and the relative depths of components within it, this latter effect being most marked for nodes which were close together in space. On the negative side, however, one subject found that the use of motion distracted attention from the task of making judgements about depth, thereby making it more difficult. This serves as an important reminder that the subjective effects of motion are likely to be felt differently by different observers and that future tools should always allow users to view static diagrams, as well as moving ones.

5.5.2 Implications for Further Experiments

This experiment provided a number of indications that the use of simulated rocking motion was likely to enhance the perception of depth in 3-d ICDs of the kind supported in JIN in at least some cases. It was therefore felt that it would be appropriate to conduct further experiments to investigate whether different kinds of motion would have different effects, and whether the effects of motion would vary for different types of diagram. JIN permitted the variation of angle, period and axis of motion as described in chapter 4 and it was decided that the next experiments to be conducted should investigate the use of different angles and periods of motion respectively.

It seemed appropriate that these further experiments should use a similar procedure to that employed for the pilot study. Certain modifications were, however, needed. Firstly, it was decided that more complex stimuli composed of a greater number of nodes and links should be used in order to make the experimental task a little more difficult. It was hoped that this might have the effect of increasing the variance in accuracy under different conditions.

Secondly, it was observed that the use of black and grey target nodes led to undesirable effects in the pilot with subjects tending to respond faster when the black node was nearer the front than when the grey node was nearer. It was noted that differences in depth cuing inherent in the markings used to discriminate between the two target nodes should be minimised in future experiments.

Since subjects seemed happy with the period of motion used in the pilot, it was decided that the range of motion periods to be investigated in a further experiment should be centred around that used above. Since the modal response to a question regarding the angle of motion suggested that subjects might have preferred stimuli to move through a slightly greater angle, it was decided that the range of angles of motion should be centred around a value slightly higher than that used for the pilot.

Experiments carried out to investigate the effects of these ranges of periods and angles of motion are described in the following chapter.

5.6 Summary

This chapter described a pilot experiment conducted in order to determine whether the kind of simulated rocking motion implemented in JIN would enhance the effectiveness with which judgements about the relative depths of components in a 3-d ICD could be made.

The results of this experiment suggested that subjects tended to make faster judgements about relative depth, and to make them more easily with moving diagrams than with still ones.

It was therefore concluded that rocking motion (as implemented in JIN) does indeed enhance the perception of depth in a particular kind of 3-d ICD. The implications of these results for the experiments reported in the following chapter were briefly discussed.

Chapter 6

Perception of Depth in 3-d ICDs: Further Experiments

6.1 Introduction

This chapter describes two experiments carried out in order to investigate the effects of varying important parameters of the rocking motion implemented in JIN on the effectiveness with which depth in 3-d ICDs could be perceived. Parameters investigated are the angle (amount) and period (speed) of rocking motion.

As stated above, the main aim of the first phase of the work described in this thesis was to investigate whether users were able make effective judgements about relative depths of components in 3-d ICDs produced using the approach embodied in JIN, JINGLE and ICDEDIT. This was done in order to determine whether such 3-d ICDs constituted a reasonable medium on the basis of which to conduct further investigations into the meaningful use of depth-based spatial coding in graphical representations of knowledge structures for use in knowledge engineering.

The results of the pilot experiment indicate that judgements about the relative depths of components in 3-d ICDs of a particular kind can be made reasonably quickly and reliably, and that subjects find it quite easy to make such judgements and are reasonably confident about them. Such judgements can apparently be made more quickly and easily when the stimulus is undergoing simulated motion of a particular kind. The experiments described in this chapter will investigate whether effective judgements can be made about the relative depths of components in a variety of different kinds of 3-d ICDs and whether the effectiveness of these judgements is likely to vary depending on the kind of motion used.

A considerable amount of work has already been done on the question of how the perception of motion can be induced by using animation of a static image, or by the sequential presentation of stationary stimuli. This work has suggested that the successful representation of apparent motion (and therefore, by extension, the provision of effective support for making judgements about relative depths of components of the stimuli) depends on setting correct values for a number of parameters. The intensities and durations of the stimuli, as well as their temporal and spatial separations have all been found to be critical factors [9]. In 1915, Korte developed a set of laws describing the relations between

these factors which he claimed must hold in order to induce an impression of smooth and continuous movement [80]. A number of experiments which have since been performed have, however, cast doubt on these initial proposals. Various stimuli and methods of presentation have been investigated and various results obtained, but few support generalisation and the interactions between the various stimulus parameters are still poorly understood. These difficulties with generalisation have been caused, at least in part, by the fact that there seem to be large differences between subjects in their responses to stimuli of this kind [9].

JIN allows the three parameters of angle (corresponding to spatial separation of stimuli in the above description), period (corresponding to duration) and axis of motion to be set by the user. In the pilot experiment, all three parameters were set to values which seemed reasonable on the basis of the experimenter's previous experience with the tool and were maintained at a constant level throughout. However, given that the illusion of motion created in such a way is apparently quite susceptible to variations in at least two of these parameters, and that there are likely to be large individual differences in response to such variations, it seemed reasonable to conduct a number of further experiments of the kind described in the previous chapter. This chapter describes two experiments carried out in order to investigate the effects of variations of angle and period of rocking motion on the perception of depth in 3-d ICDs. The first experiment looks at variations in angle of motion with period and axis fixed. The second looks at variations in period of motion with axis fixed and angle set to a value determined by the results of the first experiment.

The two experiments used the same design, procedure and stimuli (with a few minor exceptions) and are therefore described together in the following sections.

6.1.1 Hypotheses

The major experimental hypotheses were as follows:

H_1/H_2: Variations in angle/period of motion will significantly influence the effectiveness with which judgements about relative depth in 3-d ICDs can be made.

Note that H_1 is intended to relate to the first experiment concerning variations in angle, and H_2 to the second concerning variations in period. Note also that as in the pilot experiment, we are only interested in structural information, or judgements made in relative, object-centred terms.

Defining effectiveness in terms of speed, accuracy and subjective feelings of ease and confidence we may once again break down each of these hypotheses into four testable parts:

H_{1a}/H_{2a}: Variations in angle/period of motion will significantly influence the speed with which judgements about relative depth in 3-d ICDs can be made.

H_{1b}/H_{2b}: Variations in angle/period of motion will significantly influence the accuracy with which judgements about relative depth in 3-d ICDs can be made.

H_{1c}/H_{2c}: Variations in angle/period of motion will significantly influence the subjective ease with which judgements about relative depth in 3-d ICDs can be made.

H_{1d}/H_{2d}: Variations in angle/period of motion will significantly influence the confidence with which judgements about relative depth in 3-d ICDs can be made.

It was also thought that subjects might feel more comfortable with some kinds of motion than others. This gives us:

H_{1e}/H_{2e}: Variations in angle/period of motion of a 3-d ICD will significantly affect the comfort of the observer.

Furthermore, in order that the findings of these experiments could usefully be generalised to representations of interest in knowledge engineering, it was decided that a number of different types of 3-d ICD should be used (see below). It was thought that differences in stimuli might also influence the factors of speed, accuracy, subjective ease, confidence and comfort and we may therefore postulate a further set of hypotheses H_{3a-3e} analogous to those described above.

The final hypothesis concerns the stimuli alone and suggests that the perceived complexity of different types might vary:

H_{3f}: Variations in stimulus type will have a significant effect on perceived complexity.

6.1.2 Choice of Task

The type of task used in both experiments was the same as that used in the pilot study. Again, all subjects were asked to decide which of two highlighted components in a 3-d ICD appeared closer to the front. Performance of this task was timed and response times were collected and analysed in order to assess support for hypotheses H_{1a-3a}. Support for hypotheses H_{1b-3b} was assessed on the basis of error rates, and questionnaire data collected during the experiment was used in the evaluation of support for hypotheses H_{1c-3c}, H_{1d-3d}, H_{1e-3e} and H_{3f}.

6.1.3 Choice of Stimuli

As mentioned above, a number of different types of 3-d ICD were used in order that results obtained might be applicable when considering a variety of different kinds of representation for use in an interface for knowledge engineering.

Four types of stimuli were used. Two were constructed on the basis of specific structures relevant to knowledge engineering, and two were used mainly for the purposes of control (though these too might be appropriate for the representation of some kinds of knowledge structures). Type I was based on the kind of layered structure used in the pilot study and suggested for the representation of the Casnet knowledge base. Type II stimuli were based on the idea of a directed graph structure (similar to a tree in which nodes could have multiple parents) since 2-d graphical representations of trees or directed graphs are very commonly used in current tools for knowledge engineering (see chapter 3). Type III stimuli consisted of random arrangements of nodes and links. These could be seen as an extension of the kind of 2-d representations of semantic nets also commonly used in current knowledge engineering tools. Finally, type IV stimuli were regular cuboid structures which were included in the hope that some light would be thrown on the value of using highly principled structures in conveying differences in the depths of individual components.

Further details of methods used in creating these stimuli and of their exact composition are given below (see section 6.2.2.2).

6.2 Method

6.2.1 Design

A repeated measures design was used so that each subject performed the experimental task with each combination of angle (or period) and stimulus type. Four types of stimuli and four levels of angle/period were used yielding a total of sixteen experimental conditions for each of the two experiments. There were also four stimuli of each type so that every pairing of a stimulus type with a different angle/period used a different ICD. All subjects were given the same set of stimulus-angle/period pairs in different pseudo-random orderings. Orderings were constructed such that no angle/period or stimulus type appeared twice in a row for any subject.

Each subject performed a total of 17 blocks of 22 trials. The first block was a practice block and used a unique stimulus-angle/period pair not encountered elsewhere in the experiment. All subjects were given the same stimulus-angle/period pair for the initial practice block. The following 16 blocks were the experimental blocks: one for each experimental condition. The first two trials of every block were also treated as practice trials leaving a total of 20 trials per subject per experimental condition for analysis.

6.2.2 Apparatus

6.2.2.1 Hardware

The experiments were again carried out on a Whitechapel workstation with a monochrome monitor running JIN. User input was again made via the keyboard with the 'b' and 'n' keys labelled, this time with horizontal and vertical stripes corresponding to the revised method of shading nodes used in the experimental task (see section 5.5.2). The allocation of shaded labels to the 'b' and 'n' keys was again counterbalanced across subjects.

6.2.2.2 Stimuli

Four types of stimuli were used, with four stimuli of each type making a total of 16 different ICDs. All stimuli consisted of 48 nodes and 72 links (making a node-link ratio of 1:1.5) and each fell within a volume defined approximately by the ranges $x = 0-800, y = 0-800, z = 0-800$. Nodes in the stimuli were numbered and pairs of nodes for use in individual trials were picked using a random number generator. Inspection of the results from the first experiment revealed that a small number of node pairs had caused subjects particular difficulties so that their relative depths had been judged wrongly by at least 9 out of 12 subjects (see section 6.3.6). These pairs were therefore deliberately omitted from the pool of pairs from which those for use in the second experiment were chosen.

The construction and composition of stimuli of each type will now be described in a little more detail. Note that the final arrangements of all stimuli (except regular or type IV stimuli) were checked and finally adjusted using manual editing to produce the kind of

good layout which could be created automatically by a more advanced tool. Note also that the resting position (*ie* the position in the absence of motion) of each of stimulus was defined in such a way that subjects were able to see a slightly angled view of layers in $x-z$ or $y-z$ planes rather than seeing them strictly sideways on. This meant that it was possible to gain a better impression of the relative positions of components within layers. Once again, it is to be expected that this kind of appropriate default positioning could eventually be achieved automatically by a more advanced tool. Figures 6.1 - 6.4 show examples of each type of stimulus shown in the resting positions used in the experiment. The complete set of stimuli is shown in appendix B. (Note that nodes in stimuli used in the experiment were plain, not numbered as shown.)

6.2.2.2.1 Type I: Layered Type I stimuli were made up of 3 evenly separated horizontal layers of nodes and links at $y = 100$, $y = 400$ and $y = 800$ respectively. Nodes were constrained to lie within 5 co-ordinates of the relevant level of y. The small amount of freedom this allowed was used to reduce overlapping between nodes. There were always 12 nodes in the bottom layer, 16 in the middle and 20 in the top, with nodes being connected in small groups or hierarchies by links within layers. There were also some links between layers, most of which were either on or near the vertical. This arrangement of nodes and links was based on the design suggested for the Casnet knowledge base as described in section 6.1.3.

Type I stimuli were created by randomly positioning the nodes in the top layer, then placing the nodes in lower layers in similar positions on the x and z dimensions. This technique was used in order that the majority of inter-layer links were near vertical as in figure 3.7. Links were then added manually and a certain amount of manual editing was used in order to obtain a good layout within the constraints described above. The resting position for type i stimuli was defined by a 15 degree rotation about the x axis.

6.2.2.2.2 Type II: Hierarchic Type II stimuli were made up of a root node at $x = 0$ and 4 vertical layers of nodes placed at $x = 100$, 200, 400 and 800, consisting of 3, 6, 13 and 25 nodes respectively. Once again, nodes were constrained to lie within 5 co-ordinates of the relevant level of x. Each node is connected to at least one parent with some nodes being allowed more than one parent in order to maintain a node-link ratio of 1:1.5.

Stimuli were created by randomly positioning nodes in the leaf layer, then placing nodes in earlier generations in similar positions on the y and z dimensions so that paths down the tree would be relatively untangled. Links were added manually and a certain amount of manual editing was again used to achieve a good layout within the above constraints. The resting position for type II stimuli was defined by a rotation of -15 degrees about the y axis.

6.2.2.2.3 Type III: Random Nodes and links in type III stimuli were placed at random, the only constraints being that the x, y and z co-ordinates of every node should lie within the range 0 - 800 (making the total volume of a network similar to that taken up by a stimulus of any other type). A certain amount of manual editing was again necessary in order to untangle some of the crowded areas of randomly generated networks that could not be correctly represented by the tool.

6.2.2.2.4 Type IV: Regular Type IV stimuli were all regular cuboid arrangements of evenly spaced nodes. Positioning of the nodes was achieved automatically with two basic arrangements being used, and two stimuli of each of these types. Two stimuli used 3 horizontal layers of 16 nodes each, and two used 4 layers of 12 nodes each. Links were added manually in regular arrangements. The resting position for two of the type IV stimuli was defined by a rotation of 10 degrees about the x axis and 10 degrees about the y axis. For the other two stimuli of type IV, the resting position was defined by a rotation of 10 degrees about the x axis and -10 degrees about the y axis.

6.2.2.3 Motion Type

The experiment investigating angle of motion used 4 levels of angle corresponding to amplitudes of oscillation of 0.5, 0.79, 1.26 and 2.0 degrees about the x axis. This sequence of values represents a geometric progression in a range chosen on the basis of comments made by subjects in the pilot experiment (see section 5.3.3.6). A constant period of 0.63 seconds (corresponding to a rate of approximately 1.6 cycles per second) was used with all levels of angle. This value was also picked on the basis of results from the pilot study.

The second experiment, which investigated period of motion, used 4 levels of period. These were 0.4, 0.63, 1.01 and 1.6 seconds (corresponding to 2.5, 1.6, 1 and 0.63 cycles per second respectively). This sequence of values is again a geometric progression. The bottom end of the range was decided on the basis of a hardware constraint. At faster rocking speeds (lower periods of oscillation), shading sometimes occurred only partially or incorrectly: 0.4 seconds was the lowest period of motion at which shading could be reliably performed. The upper end of the range is simply 4 times the lower, as is the case with the range of angles. Again, this range corresponded well with the apparent preferences of subjects in the pilot experiment. A constant angle of 1.26 degrees of oscillation about the x axis was used. This angle was chosen on the basis of the results of the first experiment.

6.2.3 Procedure

Subjects were first shown pictures on paper of the kinds of stimuli to be used. They were told that they would see a number of these shapes during the experiment and were asked to interpret them as 3-d networks of nodes and links. The experimenter explained that the shapes would move in order to try and help them pick out depth information.

The experimental task was explained as follows. Subjects were told that two of the nodes in the network would be shaded, one with horizontal shading and one with vertical, and that they would be asked to indicate which of the shaded nodes appeared to be closer to the front of the network (in the sense that it looked closer to the viewer) by pressing the appropriate one of the two correspondingly labelled keys on the keyboard.

It was explained that the experiment would consist of 17 blocks of 22 trials, and that the first block, and first two trials in every subsequent block could be used for practice as the results would not be included in subjects' scores. Subjects were told that a buzzer would sound before every trial and every block of trials. Trials within blocks proceeded automatically, but new blocks of trials had to be initialised by subjects pressing the space bar.

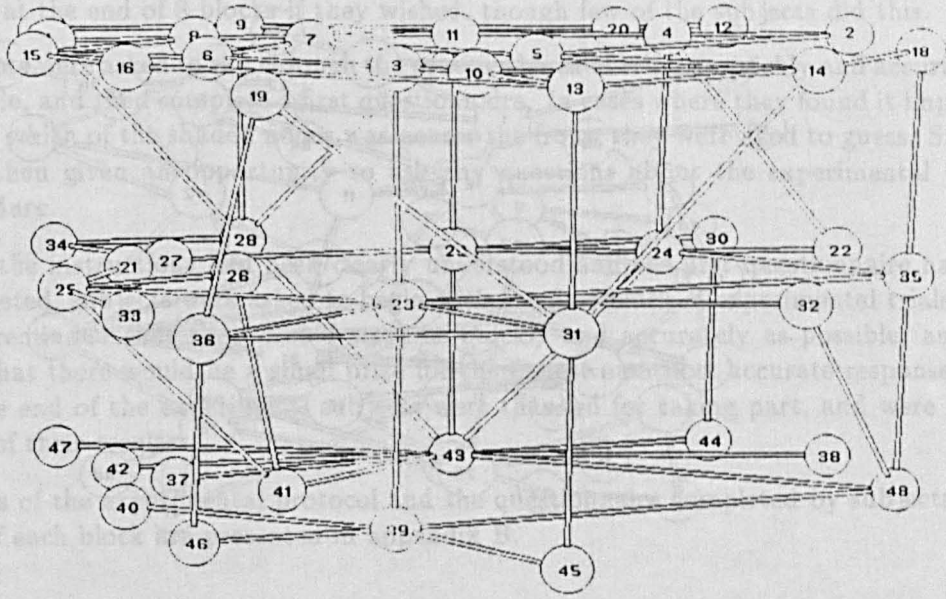


Figure 6.1: One of the Type I Stimuli (Layered)

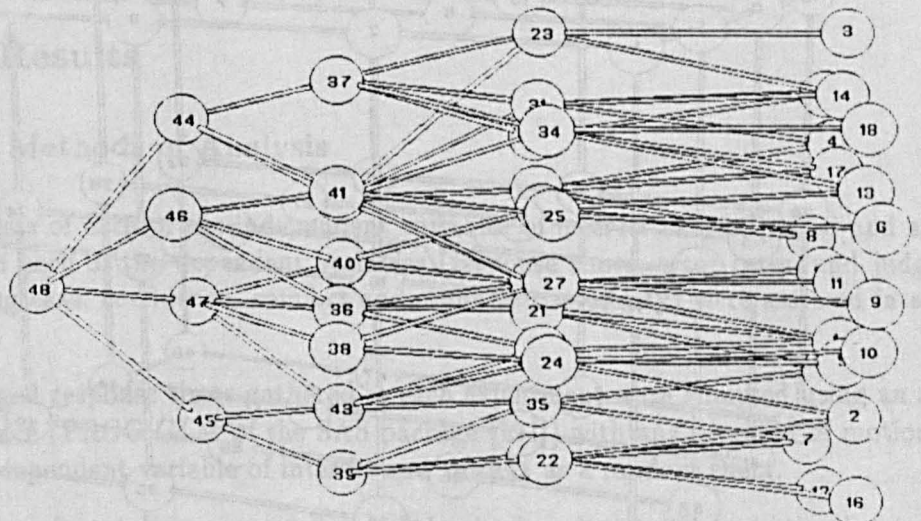


Figure 6.2: One of the Type II Stimuli (Hierarchic)

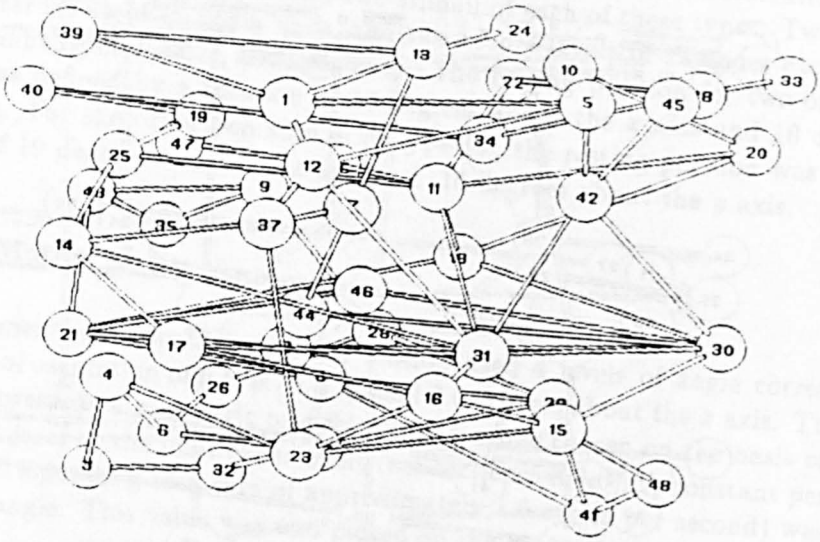


Figure 6.3: One of the Type III Stimuli (Random)

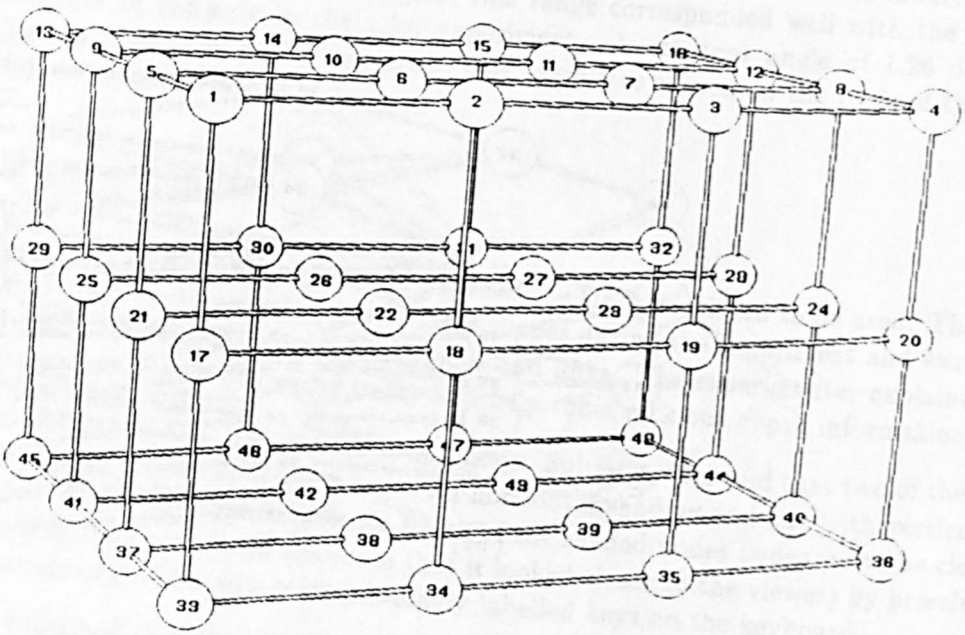


Figure 6.4: One of the Type IV Stimuli (Regular)

Subjects were asked to complete a short questionnaire of 5 questions at the end of every block. They were also told that they could take short breaks between blocks and a longer break at the end of 8 blocks if they wished, though few of the subjects did this.

Subjects were asked to run through the practice block of trials as quickly and accurately as possible, and then complete a first questionnaire. In cases where they found it impossible to tell which of the shaded nodes was nearer the front, they were aked to guess. Subjects were then given an opportunity to ask any questions about the experimental task or procedure.

Once the instructions had been clearly understood and the first questionnaire had been completed, subjects were asked to begin with the first block of experimental trials. They were reminded that they should work as quickly and accurately as possible, and were told that there would be a small prize for the fastest and most accurate response times. At the end of the experiment, subjects were thanked for taking part, and were given a copy of their results.

Copies of the experimental protocol and the questionnaire completed by subjects at the end of each block are presented in appendix B.

6.2.4 Subjects

Twelve subjects took part in each experiment. All subjects were students or lecturers from the Department of Computer Science at City University. For the experiment on angle, 10 of the subjects were male and 2 female, with an average age of 28.1 years. For the experiment on period of motion, 11 of the subjects were male and 1 female, with the average age being 28.9 years.

6.3 Results

6.3.1 Methods of Analysis

The effects of each of the independent variables of interest (angle, period and stimulus type) on each of the dependent variables (response times, error rates, and judgements regarding ease, confidence, comfort and stimulus complexity) were assessed in separate tests.

The logged response times gathered in each experiment were analysed using an analysis of variance (PROC GLM of the SAS package [121]) with angle/period of motion as the main independent variable of interest and subject as a random effect.

Differences in total error rates for each level of angle, period and stimulus type were analysed using a non-parametric Friedman two-way analysis of variance with subject as one factor and angle, period or stimulus type as the second. Note that the number of errors per condition was in general quite low: the overall percentage of errors was 12.6% for the first experiment and 8.9% for the second.

Differences in median ratings for ease of task, confidence in task performance, comfort with stimulus movement and stimulus complexity were also analysed using a Friedman

	Angle of Motion								Significance Level
	0.50		0.79		1.26		2.00		
Response Times	(4)	(35)	(2)	(29.5)	(1)	(24)	(3)	(31.5)	NS
Error Rates	1	23.5	3	27.5	4	42.5	2	26.5	0.05
Ease	2	29	3	33.5	1	17	4	40.5	0.005
Confidence	2	30.5	3	33	1	18.5	4	38	0.05
Comfort	(4)	(33.5)	(2)	(29.5)	(1)	(24)	(3)	(33)	NS

Figures on the left in each cell show rankings over all subjects. 1 denotes the best outcome (eg lowest errors or highest confidence) and 4 denotes the worst (eg highest errors or lowest confidence). Figures on the right show sums of subject by subject rankings. Figures in parentheses relate to non-significant effects.

Table 6.1: Summary of the Effects of Angle of Motion

two-way analysis of variance with subject as one factor and angle, period or stimulus type as the second.

The results of carrying out these analyses are presented in appendix B. The following paragraphs summarise important findings.

6.3.2 Effect of Angle of Motion

Table 6.1 summarises the results of analysis carried out in order to determine the effects of angle of motion on the dependent variables of interest. Graphs showing the effects of angle of motion on total numbers of errors and on ratings of ease and confidence are shown in figure 6.5.

6.3.2.1 Response Times

The analysis of the response time data from the first experiment showed the main effect of angle not to be significant, but the effects of the subject variable and the angle × subject interaction were significant at $p = 0.01$.

The fact that the effect of the angle variable was not significant implies that variations in angle of rocking motion did not have any consistent effect across subjects. However, the fact that the subject variable was significant indicates that there was a significant difference in performance between different subjects and the significance of the angle × subject interaction means that different subjects were affected by variations in angle of rocking motion in different ways.

6.3.2.2 Error Rates

The analysis of error rates showed the effect of angle of motion on the total number of errors made by all subjects to be significant at $p = 0.05$. The main cause of the significant

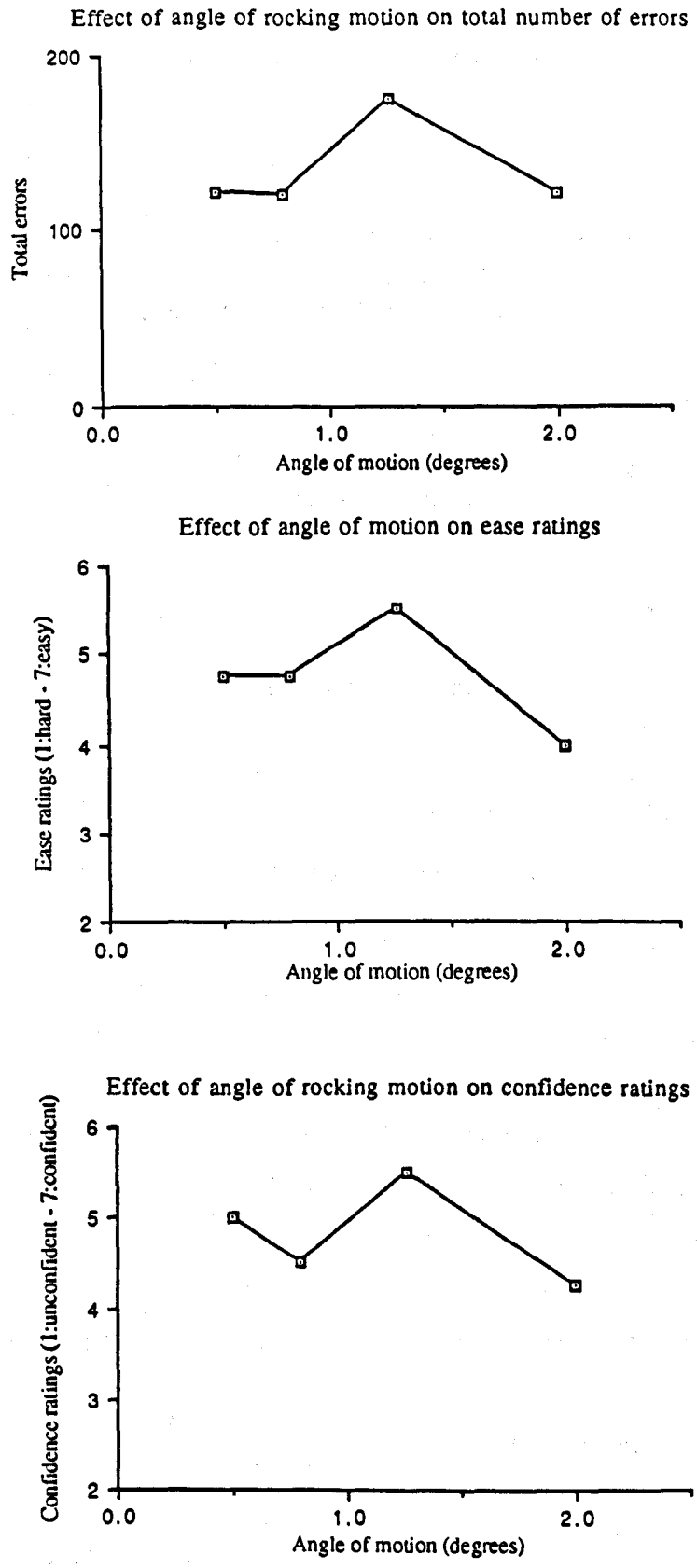


Figure 6.5: Graphs showing the effects of angle of motion on total numbers of errors and on median ratings of ease and confidence for all subjects

effect appeared to be the high number of errors made with an angle ot motion of 1.26 degrees.

6.3.2.3 Ease

An analysis of ratings of ease of task showed the effect of angle on perceived ease to be highly significant ($p = 0.005$). The task was most often judged to be easiest with an angle of 1.26 degrees and most often judged hardest with 2 degrees.

6.3.2.4 Confidence

The effect of angle of motion on subjects' confidence in their ability to make depth discriminations was significant at $p = 0.05$. The subjects were most often most confident with an angle of 1.26 degrees and most often least confident with an angle of 2 degrees.

6.3.2.5 Comfort

Subjects' ratings of comfort with the motion of stimuli were found not to be affected by the angle of motion.

6.3.2.6 Subjective Responses

Comments written by subjects after trials with different angles of motion were as follows:

Angle 1 (small)

Stimulus Type I: Layered

S3 *Better - slower*

Stimulus Type II: Hierarchic

S11 *Movement seemed useful here*

S12 *Less exaggerated movement made the shape easier to look at*

Stimulus Type III: Random

S3 *Definitely queezy-making!*

S10 *Didn't seem to be enough movement to judge the distances properly*

Angle 2

Stimulus Type I: Layered

S3 *Fastish*

Angle 3

Stimulus Type I: Layered

S3 *Very fast and rocky!*

S4 *Seemed to rock violently*

Stimulus Type II: Hierarchic

S3 *The shape moved quickly*

S7 *Seemed a bit unnatural in motion*

Stimulus Type IV: Regular

S7 *This shape moved too much for my liking*

Angle 4 (large)

Stimulus Type I: Layered

S3 *Very fast again*

S11 *Moving less obviously contributed*

S12 *Shape moved quite fast. Although this was a bit uncomfortable to watch, it seemed to make it easier to judge depth*

Stimulus Type II: Hierarchic

S3 *Fast*

Stimulus Type IV: Regular

S2 *It made me feel seasick ...*

S3 *Fast again!*

S7 *The shape moved too much for comfort*

S8 *The extent of movement in this shape was distracting from the task*

S12 *Fast moving shape was hard to follow - too jerky*

6.3.3 Effect of Period of Motion

Table 6.2 summarises the results of analysis carried out in order to determine the effects of period of motion on the dependent variables of interest. Graphs showing the effects of period of motion on mean logged response times, total numbers of errors and ratings of ease are shown in figure 6.6.

6.3.3.1 Response Times

The analysis of response time data from the second experiment showed that the main effects of both the period and subject variables were significant at $p = 0.0001$ This indicated that although there were significant differences in performance between different subjects, variations in period of motion had a consistent effect across subjects. Inspection of the data shows that subjects tended to respond most quickly with a period of motion of 1.57 seconds (*ie* with the slowest motion), and most slowly with a period of 0.35 seconds (*ie* with the fastest motion).

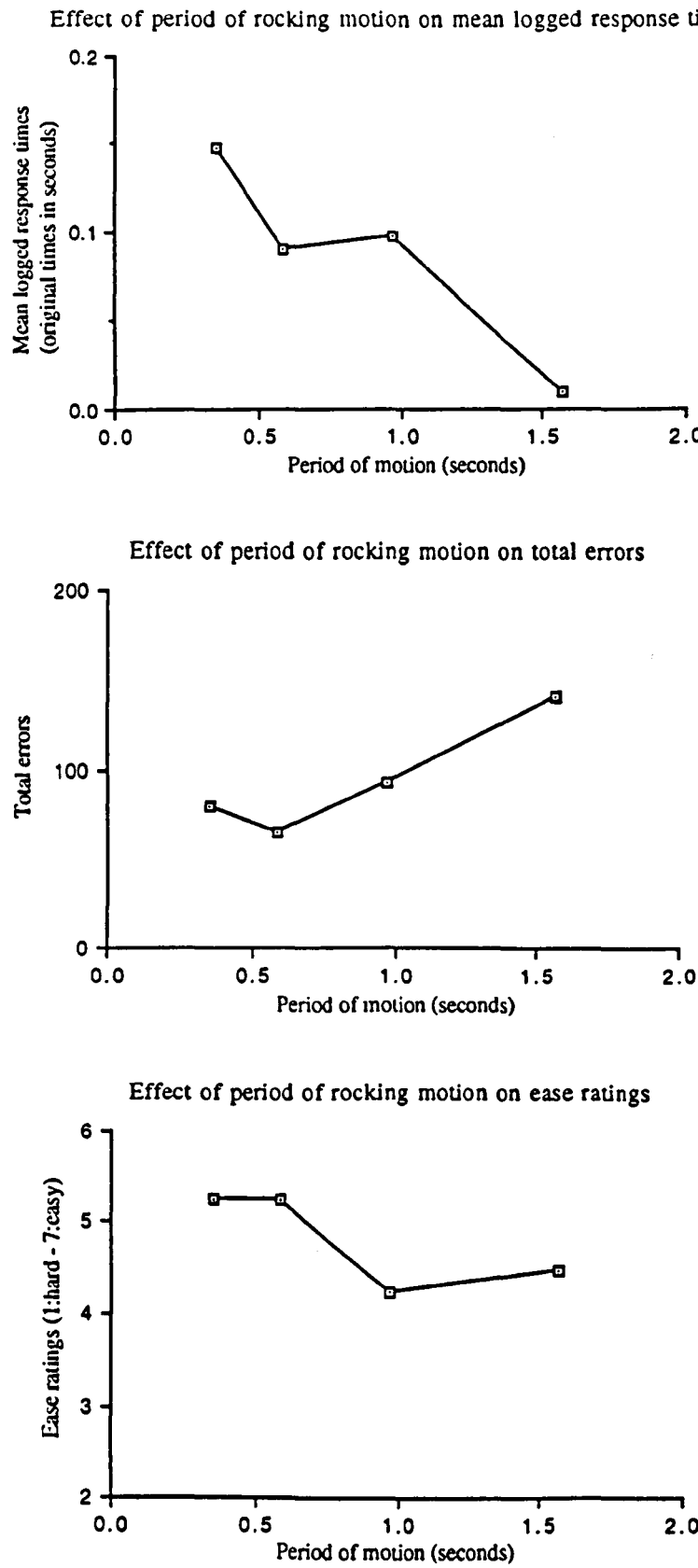


Figure 6.6: Graphs showing the effects of period of motion on mean logged response times, total numbers of errors and median ratings of ease for all subjects

	Period of Motion								Significance Level
	0.35		0.59		0.97		1.57		
Response Times	4	45.5	2	28.5	3	31	1	15	0.0001
Error Rates	3	28.5	1	22.5	2	25	4	44	0.005
Ease	2	24	1	21.5	4	41	3	33	0.05
Confidence	(1)	(24)	(2)	(25)	(4)	(38.5)	(3)	(32.5)	NS
Comfort	(4)	(36.5)	(1)	(24.5)	(2)	(27)	(3)	(32)	NS

Figures on the left in each cell show rankings over all subjects. 1 denotes the best outcome (eg lowest errors or highest confidence) and 4 denotes the worst (eg highest errors or lowest confidence). Figures on the right show sums of subject by subject rankings. Figures in parentheses relate to non-significant effects.

Table 6.2: Summary of the Effects of Period of Motion

The period × subject interaction was also significant at $p = 0.001$. This means that there were some significant differences in the way different subjects were affected (*ie* not all subjects performed best with the slowest motion and worst with the fastest).

6.3.3.2 Error Rates

The analysis of error rates showed the effect of period of motion on the total number of errors made by all subjects to be significant at $p = 0.005$. Inspection of the data suggested that the main cause of this effect was the high number of errors made with a period of 1.57 seconds.

6.3.3.3 Ease

An analysis of ratings of ease of task showed the effect of angle on perceived ease to be significant at $p = 0.05$. Subjects most often found the task most difficult with a period of 0.97 seconds though some subjects judged the task to be most difficult with a period of 1.57 seconds. Subjects judged the task to be easiest with periods of either 0.35 or 0.59 seconds.

6.3.3.4 Confidence

The effect of period of motion on subjects' confidence in their ability to make depth discriminations was found not to be significant.

6.3.3.5 Comfort

Subjects' ratings of comfort with the motion of stimuli were found not to be significantly affected by the period of motion.

6.3.3.6 Subjective Responses

Comments written by subjects after trials with different periods of motion were as follows:

Period 1 (fast)

Stimulus Type I: Layered

S2 *A bit too fast, made you constantly aware of its moving which became annoying*

S3 *Movement seemed a bit unnatural, maybe too fast*

S11 *Very fast*

Stimulus Type II: Hierarchic

S4 *Movement rather fast*

Stimulus Type III: Random

S2 *A bit fast*

S5 *Too fast*

Stimulus Type IV: Regular

S2 *Too fast*

S4 *Movement seemed rather excessive, tiring to watch. Also a bit jerky.*

S5 *Too fast for maximum comfort but helps response speed!*

S11 *Fast!*

Period 2

Stimulus Type I: Layered

S11 *Too fast*

Stimulus Type II: Hierarchic

S11 *Fast*

Period 3

Stimulus Type I: Layered

S11 *Rather quick*

Period 4 (slow)

Stimulus Type I: Layered

S4 *Movement jerky. Very difficult to compare nodes a long distance apart*

S9 *Too slow*

Stimulus Type II: Hierarchic

S4 *Movement a bit slow and jerky*

Stimulus Type III: Random

S4 *Because the shape moved so slowly, it took longer to make a decision*

S5 *Rocking too slow*

Stimulus Type IV: Regular

S4 *Movement rather slow and jerky*

6.3.4 Effect of Stimulus Type

Table 6.3 summarises the results of analysis carried out in order to determine the effects of stimulus type on the dependent variables of interest. Graphs showing the effects of stimulus type on total numbers of errors, ratings of ease and ratings of complexity are shown in figure 6.7. Further graphs illustrating the relationship of ratings of perceived complexity to total numbers of errors and ratings of confidence are shown in figure 6.8.

6.3.4.1 Response Times

Differences in the type of stimulus were not found to significantly affect the speed with which subjects were able to make judgements about the relative depths of components of the stimulus in either experiment.

6.3.4.2 Error Rates

There were significant differences between the number of errors made with stimuli of different types in both experiments. In experiment 1 the difference was significant at $p = 0.05$, and in experiment 2 the difference was significant at $p = 0.001$.

In the first experiment, most errors were made with random stimuli and least with regular, with medium numbers of errors using layered and hierarchic stimuli. In the second experiment, most errors were made with regular stimuli, and least with random.

6.3.4.3 Ease

There were no significant differences in judgements of the ease of depth discrimination tasks for stimuli of different types in either experiment.

6.3.4.4 Confidence

Subjects were significantly more confident about making relative depth judgements in stimuli of some kinds than they were with others. Differences in confidence for different types of stimulus were significant at $p = 0.05$ in both experiments.

In each case, subjects tended to be relatively unconfident of their ability to make judgements about relative depths in random stimuli. In the first experiment, subjects tended

		Stimulus Type								Significance Level
		Layered		Hierarchic		Random		Regular		
Response Times	E1	(4)	(35.5)	(1)	(24)	(2)	(28)	(3)	(32.5)	NS
	E2	(2)	(30.5)	(3)	(33)	(4)	(35)	(1)	(21.5)	NS
Error Rates	E1	3	29.5	2	29	4	40.5	1	21	0.05
	E2	2	25.5	3	37.5	1	13	4	44	0.001
Ease	E1	(4)	(36.5)	(2)	(27.5)	(1)	(27)	(3)	(29)	NS
	E2	(3)	(31)	(2)	(30.5)	(4)	(35)	(1)	(23.5)	NS
Confidence	E1	3	35.5	1	20.5	4	36	2	28	0.05
	E2	3	32	2	29	4	39	1	20	0.05
Comfort	E1	(1)	(24.5)	(4)	(38)	(2)	(26)	(3)	(31.5)	NS
	E2	(2)	(29)	(3)	(31)	(1)	(28)	(4)	(32)	NS
Complexity	E1	2	27.5	3.5	40	3.5	40	1	12.5	0.001
	E2	2	28	3	34.5	4	45	1	12.5	0.001

'E1' denotes results from experiment one, 'E2' results from experiment 2.
Figures on the left in each cell show rankings over all subjects. 1 denotes the best outcome (eg lowest errors or highest confidence) and 4 denotes the worst (eg highest errors or lowest confidence). Figures on the right show sums of subject by subject rankings. Figures in parentheses relate to non-significant effects.

Table 6.3: Summary of the Effects of Stimulus Type

to be most confident about judgements hierarchic stimuli, and in the second, they were most confident with regular stimuli.

6.3.4.5 Comfort

Subjects did not judge their comfort to be significantly affected by differences in the type of stimulus in either experiment.

6.3.4.6 Complexity

Differences in judgements of the complexity of stimuli of different types were significant at $p = 0.001$ in both experiments.

Inspection of the data revealed that for experiment 1, the main source of difference was the fact that regular stimuli were often judged to be least complex and layered stimuli next least complex with hierarchic and random stimuli each being judged most complex by a number of subjects. For experiment 2, the pattern was the same with the exception that random stimuli were more often judged most complex than hierarchic.

6.3.5 General Comments

Further general comments made by the subjects in relation to the experiment as a whole were as follows:

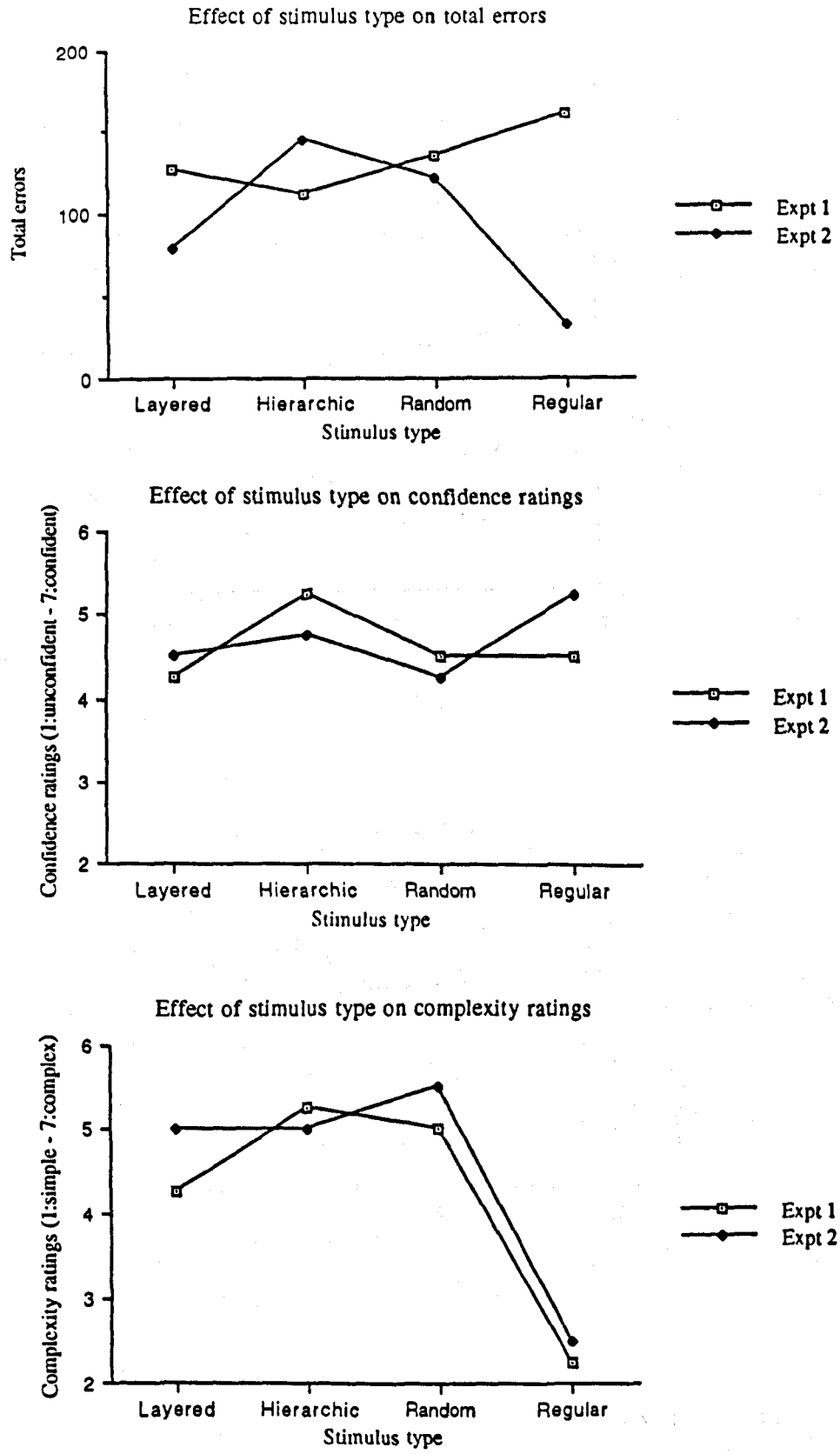


Figure 6.7: Graphs showing the effects of stimulus type on total numbers of errors and median ratings of confidence and complexity for all subjects

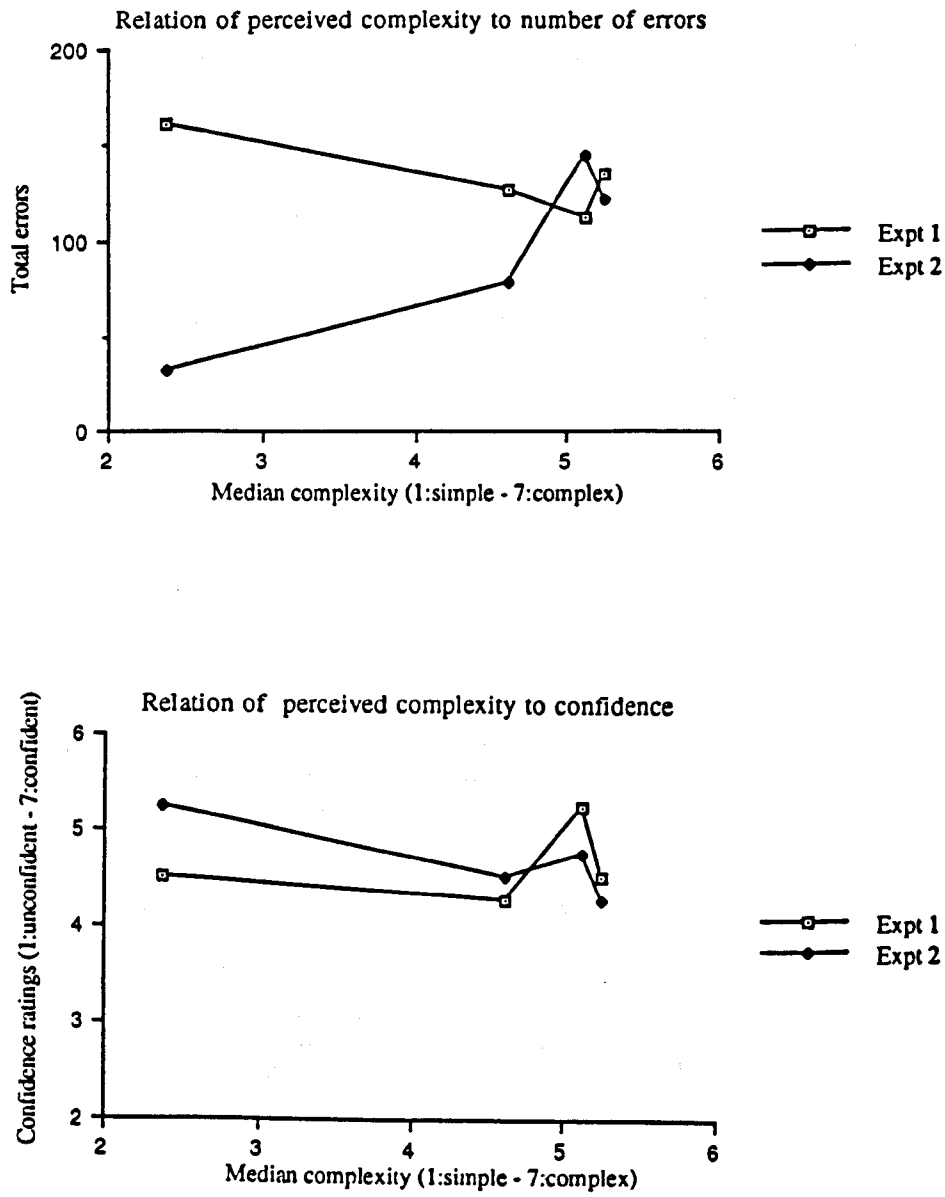


Figure 6.8: Graphs showing the relationship of ratings of perceived complexity to total numbers of errors and ratings of confidence for all subjects

i) markings used to distinguish between targets acted as depth cues:

S3 (expt 1) *I can see the [vertical shading] better than the [horizontal shading] so I favour it*

S9 (expt 1) *When shaded areas almost parallel the 'heavier' shaded vertical shading seemed closer*

S5 (expt 2) *Seems easy to think [vertical shading] is nearer than it is relative to [horizontal shading]*

ii) nodes appeared to change size:

S7 (expt 1) *I didn't like the nodes changing size*

S11 (expt 1) *Change in size of blobs seemed misleading*

S12 (expt 2) *Size of spheres at some nodes seemed to change quite suddenly - produced an off-putting pulsating effect*

iii) movement can become tiring:

S4 (expt 1) *I think the movement may have bothered me because I've been looking at the screen for a while*

S4 (expt 1) *Movement starting to annoy me*

6.3.6 Analysis of Difficult Pairs

Inspection of the data suggested that some pairs caused subjects particular problems in that judgements about the relative depths of the two nodes were often made incorrectly (see table 6.4). In the first experiment, 5 pairs were judged incorrectly by 11 out of the 12 subjects, 7 pairs by 10 and a further 10 were judged wrongly by 9 of the subjects. In the second experiment, for which the first experiment's problematic pairs had been excluded from the pool as explained in section 6.2.2.2, the situation was not quite as bad: here 1 pair was judged wrongly by all 12 subjects, 1 by 11, 2 by 10 and 2 by 9.

On the basis of these observations, it was decided that an informal investigation into the possible causes of such difficulties should be conducted. Pairs about which judgements were apparently hard to make were examined and factors common to a number of these pairs were listed. Pairs examined were those for which the judgements of more than one third (*ie* more than 4 out of 12) of the subjects were incorrect.

Factors thought to contribute to the difficulty of making a correct judgement about the relative depths of some pairs of nodes were as follows.

6.3.6.1 Small Difference in Depth

It seems that judgements about the relative depths of two nodes were harder to make when those nodes were actually close in depth. Note that in such cases, there may be no difference in the apparent size of the two nodes (due to perspective), or in the amount of motion which each undergoes. Hiding may then provide a direct cue in a minority of cases (in which the two nodes actually overlap) but viewers are often likely to have to make judgements on the basis of their overall understanding or 'gestalt' of the structure

No. Subjects with Errors	0	1	2	3	4	5	6	7	8	9	10	11	12
No. Pairs (Experiment 1)	207	75	17	12	13	7	7	8	6	10	7	5	0
No. Pairs (Experiment 2)	232	63	25	15	10	10	3	6	4	2	2	1	1

Table 6.4: Numbers of Pairs Causing Particular Numbers of Errors

as a whole. This may not provide a very accurate basis on which to discriminate the depths of two nodes which are actually close in depth.

6.3.6.2 x-y Distance

It seemed harder to judge the relative depths of pairs of nodes whose projections are well separated on the screen. This difficulty seemed to increase for pairs of nodes which were actually close in depth. In this case, subjects might again be forced to make judgements on the basis of an overall gestalt in the absence of more local cues.

6.3.6.3 Masking

There was apparently a tendency to pick the more visible node of a pair. When the front node of a pair was more obscured by other nodes or links than the back node, this lead to an incorrect judgement. This effect again appeared to be exacerbated in cases in which the actual difference in depth was small. Two examples of cases in which nodes are masked are shown in figure 6.9.

6.3.6.4 Perspective Cues from Links

If the front node of a pair was connected to a number of nodes in front of it (so that visual cues provided by the links tended to place it towards the back of the network—see, for example, figure 6.10) and the back node had no such links (or was connected with other links in such a way as to create an opposite effect), subjects were likely to make an incorrect judgement. This was also true in the reverse situation, that is, when visual cues provided by links to the back node tended to place it towards the front, and cues provided by links to the front node had either no effect or tended to place it towards the back.

6.3.6.5 Centrality

Subjects tended to pick the node which was more central to their field of view. In cases in which this node was actually further back than the other target, this lead to an incorrect judgement. Note again, that this tendency was most marked in cases in which judgements might in any case have been difficult to make as nodes were actually close in depth. Note also that this effect may well have been exacerbated by the pressure to respond quickly.

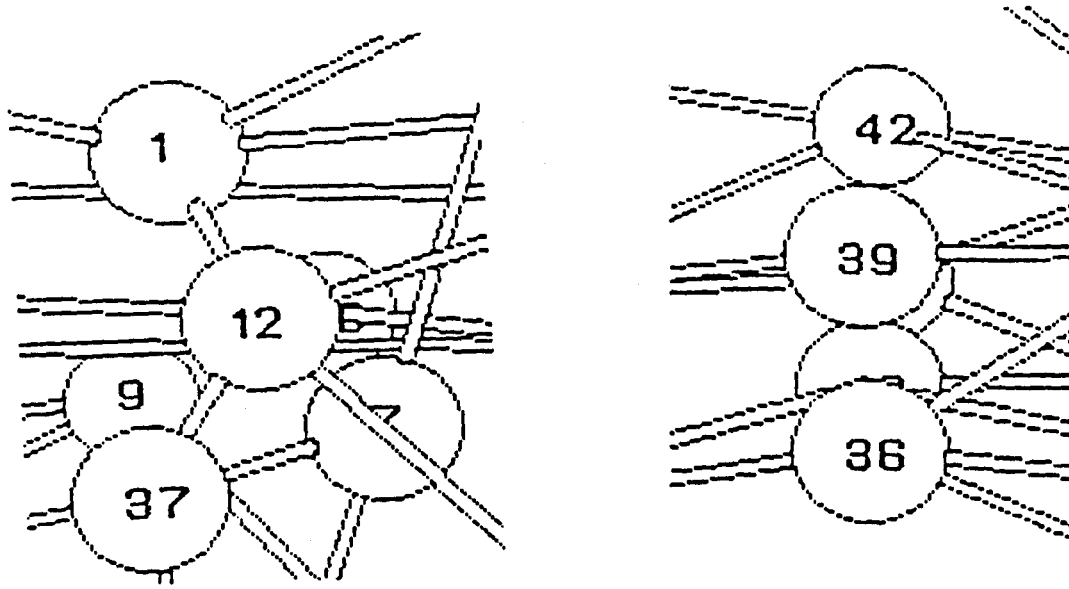


Figure 6.9: Examples of Masking

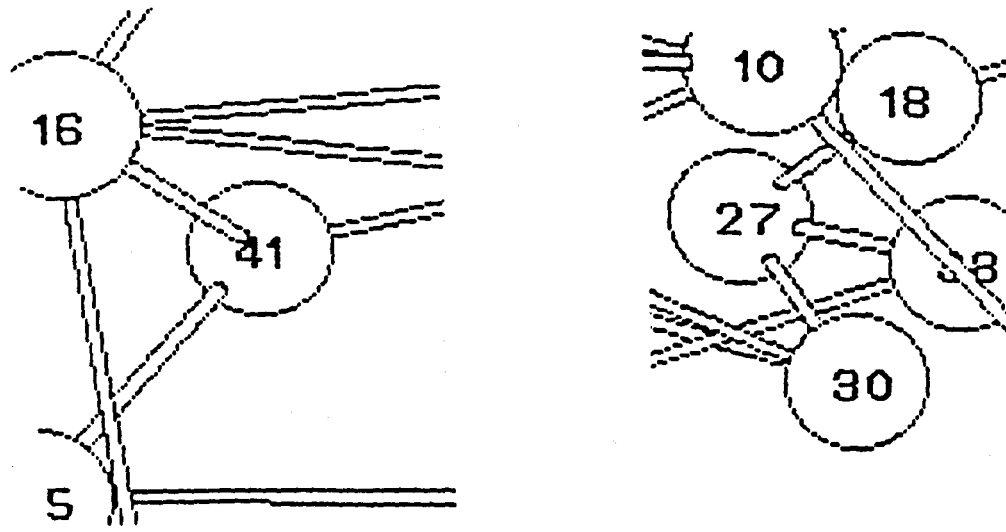


Figure 6.10: Examples of Perspective Cues from Links

6.4 Conclusions

The results of these experiments indicated that users are able to make reasonably effective judgements about the relative depths of components in 3-d ICDs of the kind supported by JIN. It was therefore felt that 3-d ICDs produced using the particular approach to creating the illusion of depth described in previous chapters constituted a reasonable medium with which to conduct broader investigations into the utility of 3-d node and link diagrams in knowledge engineering and the usability of depth-related features of ICDEDIT.

There was some support for each of the experimental hypotheses:

- H_1: that variations in angle of motion would significantly influence the effectiveness with which judgements about relative depth in 3-d ICDs could be made;
- H_2: that variations in period of motion would significantly influence the effectiveness with which judgements about relative depth in 3-d ICDs could be made;
- H_3: that variations in stimulus type would significantly influence the effectiveness with which judgements about relative depth in 3-d ICDs could be made.

Variations in angle of motion were found to affect the accuracy, ease and confidence with which such judgements could be made; variations in period of motion affected the speed, accuracy and ease; and variations in stimulus type affected accuracy, confidence and judgements of perceived complexity.

In many cases, the dependent variables (relating to speed, accuracy, ease, confidence and complexity) were affected in different ways by the same variations in angle, period or stimulus type. For example, the period of motion associated with the greatest speed, was also associated with the lowest accuracy. Thus the choice of a particular angle or period of motion in a future 3-d ICD interface may depend on the particular effects required (*eg* high speed, high accuracy or feelings of ease or confidence). If each of these factors are, however, judged to be of equal importance, we may choose an angle of motion of approximately 0.5 degrees (the smallest used in these experiments) which was shown to be associated with relatively low error rates and high ratings of ease and confidence, and a period of motion of 0.59 seconds (the second slowest used here) which was shown to be associated with low error rates, high ratings of ease and reasonably fast response times.

The choice of a particular type of stimulus is more difficult. The results of both experiments showed regular and hierarchic stimuli to be associated with high ratings of confidence, while subjects were relatively unconfident about making relative depth judgements with random stimuli. Results concerning error rates were, however, apparently contradictory: in the first experiment, regular stimuli were associated with the lowest error rates and random stimuli with the highest, but in the second experiment, this situation was reversed. It is assumed that this difference reflects the removal of 'difficult pairs' with more problems having been caused in the first experiment by the existence of such pairs in random shapes than in regular, but further research should ideally be carried out to investigate this effect further. In any case, it is likely that the choice of a particular type of 3-d ICD for use in the interface will depend largely on the type of abstract structures to be represented. This claim will be supported by the results of the case studies described in the following chapters in which knowledge engineers developed different kinds of 3-d ICD to represent different kinds of knowledge structure.

6.5 Discussion

6.5.1 Comments on the Experimental Design

The experiments described above used a number of dependent variable in order to assess various aspects of the effectiveness with which relative depth judgements could be made. 'Comfort' with the motion of the stimulus was found not to be significantly affected by any of the independent variables of interest: future experiments might therefore not attempt to measure such a variable. Effects on ease and confidence were closely linked: where both were significantly affected, they were both affected in the same way, and where only one was significantly affected, the other was also affected in a similar way. Future experiments might therefore measure only one of these variables. Finally, since perceived complexity was not a predictor of speed, accuracy, ease or confidence, it is not clear that ratings of perceived complexity should in future be collected. The significance of perceived complexity of stimulus might, however, be investigated further in different experiments, since there were obviously substantial differences in the perceived complexity of stimuli of different types.

One problem with the design of the experiments was as follows. Despite the efforts made to choose discriminatory markings for the two target nodes which would impose no bias on the performance of the experimental task, analysis showed that the markings of the two nodes (that is, whether the front node was vertically marked and the back node horizontally, or vice versa) had a significant effect on the speed with which judgements were made by 6 out of 12 subjects in the first experiment and 2 out of 12 in the second. Fortunately, however, it was found that in exactly half the cases in which colour of front node had a significant effect on response times, the mean response time for trial in which the vertically shaded node was at the front was lower, and in the other half, the opposite was the case. The differences in discriminatory markings would therefore not have imposed a consistent overall bias on the results. Further attempts should, however, be made to avoid such effects completely in future.

A second problem was that due to the inadequacies of the random number generator used, the same pair of nodes was occasionally presented for two of a block of 20 trials with a particular angle/period combination. This was unfortunate since the number of trials per angle/period stimulus type combination was already low, and since, as discussed above, all subjects saw the same set of pairs for each angle/stimulus combination so that the effects of any such repetition would have been replicated across all subjects.

The greatest problem was, however, the inadvertent confounding of angle/period and stimulus type combinations with stimulus type variants. This meant that on every occasion when a particular level of angle or period was used with a particular type of stimulus, the same one of the four possible examples of stimuli of that type was always used. As described in section 6.2.2.2, each of the stimuli had been constructed according to certain rules which were thought of as defining particular types. However, while little is known of the salient characteristics of diagrams of this kind, it was felt that such typing may not have been fully valid in that stimuli may have differed in significant ways not anticipated at the stage of design. Because of this confounding and the fact that stimulus typings may not have been valid, it was felt that interactions between the effects of level of angle or period and those of stimulus type could not validly be considered in the analysis of the results. This was unfortunate as it had previously been envisaged that such interactions would yield some of the most interesting results. Designs for future experiments should

take care to avoid such pitfalls.

Difficulties inherent in designing experiments such as those described in this chapter will be discussed further in chapter 12.

6.5.2 Implications for Future Implementations

Future implementations of tools supporting 3-d ICD interfaces might provide a default type of rocking motion. As described above, the results of these experiments suggest that it might be appropriate to use a default period of motion of approximately 0.59 seconds and a default angle of motion of approximately 0.5 degrees.

Since subject × angle/period interactions were also significant, a further implication of the results is that users should ideally be provided with facilities which would allow them to adjust the type of motion provided to suit their own individual preferences. This seems the most reasonable response to the large differences in performance and preference between subjects and also means that users are free to adjust the type of motion depending on the type of ICD displayed, even if such adjustments are not made automatically as described above. Note that it should also allow subjects to effectively 'switch motion off' if it comes to seem tiring or annoying.

The problems of anti-aliasing, which caused nodes to apparently change in size in a way which bothered a number of subjects (see section 6.3.5) should also be tackled in any future implementations as should some problems with presentation as described in the following section.

6.5.3 Implications for the Layout of 3-d ICDs

Heuristics for the layout of 3-d ICDs would be useful, both as a basis on which to manually edit diagrams to achieve a good layout as described in sections 5.2.2.2 and 6.2.2.2, and as a basis for future implementations of tools which might produce such layout automatically. A number of sources of information could contribute to the development of a suitable set of heuristics as described below.

The use of manual editing to produce what seemed to be good layouts for the stimuli used in the experiments lead to the observation of a number of arrangements of nodes and links which might best be avoided. Firstly, it was noted that the existence of large numbers of apparently disordered crossings between links caused diagrams to look more complex than was necessary given a particular connectivity, and easily caused the illusion that the display represented a structure consisting of spheres and cylinders to break down. Similar findings are often reported in the literature [140, 139]. Such crossings should therefore be avoided wherever possible. Secondly, the illusion of a structure of spheres and cylinders was broken in cases in which nodes and links were positioned so closely that in a real model, parts of their volume would have occupied the same space. This kind of arrangement should also be avoided, perhaps by imposing a minimum allowable separation between components. Further arrangements which apparently caused problems for subjects attempting to make judgements about the relative depths of particular nodes. Such arrangements should therefore be avoided in diagrams in which information about relative depths is to be interpreted in a meaningful way, perhaps by imposing a

minimum allowable depth between components. Further research would be needed to investigate this possibility more fully.

The imposition of a minimum allowable separation in depth might also have the effect of increasing the reliability with which judgements about relative depth could be made in other cases. Overall error rates in the two experiments reported were 12.6% and 8.9% respectively and the overall figure for the pilot study was 12%. This level of inaccuracy may be thought unacceptably high for some tasks or areas of application. It should, however, be remembered that as pairs of nodes for use in the experiments were picked at random from stimuli spanning a depth of approximately 800 pixels, the difference in depth for two nodes in a target pair could also vary at random between 1 and 800 pixels (pairs with depth difference 0 were automatically ruled out). Some of the judgements about relative depth would therefore have had to be made on the basis of minimal cues. Two nodes within 1 or 2 pixels of each other in depth would often appear to be exactly the same size and to move just the same amount since both size and amount of motion would effectively have been rounded to the nearest pixel. For depth differences greater than a certain threshold (whose value would depend on such factors as the size of nodes displayed, the distance into the screen at which they were intended to appear, and the resolution of the screen itself), differences in apparent size or amount of motion would reach a value of at least 1 pixel. Further work could investigate whether the imposition of particular thresholds as minimum allowable depth differences would raise the reliability with which judgements about relative depths could be made to more acceptable levels.

6.5.4 Implications for Phase Two

As stated earlier, the main motivation for carrying out the experiments described in chapters 5 and 6 was that of determining whether 3-d ICDs produced using the techniques employed by JIN constituted a reasonable medium for use in further investigations into meaningful depth-based spatial coding in graphical representations of knowledge structures for use in knowledge engineering. The results of these experiments have indicated that users are able to make reasonably effective judgements about the relative depths of components in 3-d ICDs of the kind supported by JIN. It is therefore argued that such 3-d ICDs do indeed constitute a reasonable medium for use in further investigations of the kind envisaged.

By the time the experiments reported in this chapter were finished, a further prototype tool (ICDEDIT) with a considerably greater functionality was available for use in supporting 3-d ICDs generated using the same techniques for creating the illusion of depth as employed in JIN. It was felt that it would be preferable to use ICDEDIT as the basis for further studies as its more advanced functionality would permit investigation of the use of 3-d ICDs in more realistic situations than would have been possible using JIN.

The general approach to creating the illusion of depth is the same in JIN, JINGLE and ICDEDIT as described in chapter 1. The types of diagram supported by JIN and ICDEDIT are, however, somewhat different. While diagrams in JIN consist of spheres and cylinders, those in ICDEDIT consist of rectangles and single line links. Some cues to depth in JIN (those which can be inferred from visual information about the shapes of components of the diagram) were therefore absent in ICDEDIT.

Of course, particular values of rocking motion parameters derived from experiments with JIN would not be directly applicable to ICDEDIT. The results of the experiments per-

formed did, however, suggest that the overall approach to creating the illusion of depth was effective. For a variety of reasons (to be discussed further in chapter 12) it was therefore decided that work on phase two should be begun without first carrying out a further experimental investigation of the effectiveness with which judgements about the relative depths of components in 3-d ICDs of the kind supported by ICDEDIT could be made. The effectiveness of depth cues in ICDEDIT was instead investigated in a less formal manner as part of the broader investigation of usability of depth-related features of ICDEDIT whose results are reported in chapter 9.

6.6 Summary

This chapter described two experiments carried out in order to investigate whether effective judgements could be made about the relative depths of components in a range of 3-d ICDs and whether the effectiveness of those judgements was likely to vary depending on the kind of motion used.

The results of the experiments provided limited support for each of the major experimental hypotheses. Variations in angle of motion were found to affect the accuracy, ease and confidence with which such judgements could be made; variations in period of motion affected the speed, accuracy and ease; and variations in stimulus type affected accuracy, confidence and judgements of perceived complexity. The implications of the results of these experiments for further experiments and for future implementations of tools supporting the use of 3-d ICDs were discussed. It was concluded that 3-d ICDs produced using the particular approach to creating the illusion of depth described in previous chapters constituted a reasonable medium with which to conduct broader investigations into the utility of 3-d node and link diagrams in knowledge engineering and the usability of depth-related features of ICDEDIT.

Part III

Utility and Usability of 3-d ICDS

This part of the thesis is concerned with the utility of 3-d ICDs for knowledge engineering and the usability of depth-related features of ICDEDIT, a tool intended to support the use of such diagrams. Chapter 7 discusses the concepts of utility and usability and describes case studies used to investigate the utility and usability of 3-d ICDs for knowledge engineering. Findings regarding the usability of ICDEDIT are presented in chapter 9 and chapter 8 presents estimates of the utility of 3-d ICDs in various knowledge engineering situations.

Chapter 7

Investigation of Utility and Usability

7.1 Introduction

This chapter describes the methods used in investigating the utility in knowledge engineering of interface technologies based on the use of 3-d ICDs of the kind supported by ICDEDIT, and the usability of features in ICDEDIT which permit 3-d ICDs to be viewed and edited.

It begins with an introduction to the notion of usability and to the methods commonly used in the evaluation of software usability. Issues relating to software usability have, in recent years, been the subject of a considerable amount of research. Many approaches to usability evaluation have been proposed and research aimed at developing these approaches is ongoing. The review presented in the following section focuses on the use of 'ecological approaches'.

The following section presents a definition of utility and discusses the relevance of this notion to developers of new interface technologies. The idea of utility has received little attention from the HCI community [69] and methods for estimating the utility of particular systems or interface technologies are not well established. In section 7.2, the focus is on estimating the relative utility of a particular interface style or technology with respect to other existing styles, and an approach to making such estimates is proposed.

The chapter ends with a description of how a series of six case studies was used to investigate both the utility of 3-d ICDs and the usability of ICDEDIT within a unified framework. The procedure developed to incorporate aspects of an ecological approach to usability evaluation and a contextual approach to the estimation of utility is described in some detail. Information about the subjects who took part in the case studies is also presented.

7.2 Estimation of Utility

7.2.1 Definitions of Utility

The concept of 'utility' has not commonly been considered by the HCI community in recent years. The term 'utility' does not in fact appear in the index of Helander's comprehensive 'Handbook of Human-Computer Interaction' [45].

In the Oxford English Dictionary, utility is defined as 'the fact, character or quality of being useful or serviceable; fitness for some desirable purpose or end' or 'The ability, capacity or power of a ...thing to satisfy the needs ...of the majority'. This definition is the product of work by a number of philosophers including, most notably, Jeremy Bentham [43].

Interpreting this definition in terms of the concerns of those working in the field of HCI yields a broad definition of system utility as 'the ability, capacity or power of a system to support the majority of users in carrying out the majority of tasks in the majority of environments which they are likely to encounter' where ideas of the majority of users and tasks are to be interpreted in terms of intended user populations and task pools for a particular system. One system can therefore be said to have greater utility than another if it can support a greater proportion of the user population in performing a given task, or if it can support the performance of a larger number of tasks from the task pool, or if support can be provided in a wider range of environments.

Considering such a definition in the context of the software development lifecycle also reveals a strong relation with the notion of requirements capture. Early estimation of the potential utility of a system corresponds to assessing whether there is a need for that system, and attempts to discover in what situations a system will be most useful (*ie* in what situations its utility will be highest) can be seen to lead almost directly to initial statements of requirements. At the beginning of the software development lifecycle, the range of contexts in which the developing software might potentially be used may be wide. This is especially true in the case of tools supporting the use of particular interface technologies which may often be developed before any useful application has been identified: 'Too frequently we attempt to build things that will use technology, rather than trying to use technology to solve human problems' [83]. An early consideration of utility will permit developers to make informed decisions about whether work on developing a particular system or technology is justified, and, if so, to identify the situations within which the proposed system or technology is likely to be most useful. Developers will then need only to consider what facilities will be required in a restricted range of situations and the time, effort and money spent on development can be directed at providing the precise functionality needed in those contexts.

7.2.2 Approaches to the Estimation of Utility

Owing to the lack of attention to the concept of utility in recent years, there are at present no widely accepted methods for utility assessment. The method used here was developed specifically to permit an early estimation of the utility of 3-d ICD interfaces in tools for knowledge engineering and to support some assessment of the relative utility of 2- and 3-d ICD interfaces in particular knowledge engineering situations.

The method is based on a definition of the utility of an interface technology as: 'the ability, capacity or power of that technology to support the majority of users in carrying out the majority of tasks using the majority of underlying software tools in the majority of environments which they are likely to encounter'. Since the focus in this work is on considering the utility of 3-d ICD technologies as interfaces to tools for knowledge engineering, users of interest were knowledge engineers, tasks were knowledge engineering tasks and the environments of interest were those in which knowledge engineering typically takes place. The case studies were used to collect information relevant to each of these factors and this information then formed the basis for estimations of the utility of 3-d ICD interfaces for knowledge engineering discussed in chapter 8.

7.3 Evaluation of Usability

7.3.1 Definitions of Usability

Issues relating to usability have for some while been of interest to those in the field of human-computer interaction. Definitions of these terms have, however been changed or augmented a number of times over the past few years in a way that reflects changes in emphasis in the concerns of HCI as a whole.

Eason [42] defined usability as 'the extent to which a user can exploit the potential utility of a system' [146] thereby intimately linking usability with utility. At the same time however, Bennet [7] drew a distinction between utility, or functionality of a system, and usability, the former being assessed in terms of the degree to which a system answered particular task needs, and the latter relating more to the user's characteristics and skills.

Three years later, Richardson [112] introduced a further dimension of usability: that of 'acceptability' which encompassed more subjective factors (such as whether the user liked the system) and began to take in broader concerns such as that of the organisational context within which a system would be used.

A view of HCI which has recently been gaining in popularity is set out by Kellogg [78]. According to this view, HCI should ideally help us to 'understand real people acting in real situations' in order that useful and usable computer artefacts may be designed on the basis of that understanding. This view places considerable emphasis on the notion of 'context' where particular contexts are believed to be characterised as particular combinations of users, tasks and physical, social and organisational environments.

The recent ISO draft proposal for a usability statements standard [17] reflects a similar view. It incorporates subjective and environmental factors by defining usability as 'the effectiveness, efficiency and satisfaction with which specified users can achieve specified goals in a particular environment.' According to this definition, a system's usability may be determined by assessing the performance and attitudes of particular groups of users as they carry out particular tasks in particular environments in the manner intended by the system's designers.

Ideas of utility and usability are, according to the definitions used here, strongly related; but they can be seen to differ in a number of important ways. Firstly, it can be seen a priori that usability is a necessary condition for utility but not a sufficient one. Thus, a system or technology cannot be useful unless it is usable. It may however be usable but not useful if, for example, the tasks whose performance it supports are already

supported by other systems. While the usability of a particular system or technology may be assessed independent of any other, its utility can only be assessed in relation to other systems intended for use in the same contexts. Secondly, usability is a function of a particular context and is determined by considering the degree to which 'specified users can achieve specified goals in a particular environment'[17]. Utility, on the other hand, is a function of a range of contexts: the greater the number of contexts in which support is provided by a given system, the more useful that system is deemed to be.

7.3.2 Approaches to the Evaluation of Usability

In the past, human factors experts might typically have been drafted in towards the end of the system development lifecycle and asked to 'sort out the interface'. There would have been little in the way of formal or tool-based support, and any changes which could be made at such a late stage would have been largely cosmetic.

Over the last 10 or 15 years, as understanding of the issues involved in human-computer interaction has increased, this situation has improved. Formal models of interaction which can be used predictively to assess potential interface designs have been developed. There has also been increasing emphasis on the incorporation of user-centred considerations throughout the development lifecycle so that a large number of evaluation techniques have been developed, each suitable for use under slightly different sets of circumstances.

Approaches to the evaluation of usability vary in terms of the resources required to use them, the type of results yielded and the stage in development or type of application for which they are suitable [153]. A brief description of a number of different approaches is given below.

7.3.2.1 Expert Evaluation

The kind of expert evaluation described above relies almost exclusively on the knowledge and experience of the human factors specialist. Such an expert may be called in early on in development to assess possible system or interface designs. More typically though, he or she may be called on to evaluate a working prototype. The objective in this case is to assess the mismatch between a system's performance in its current state and its performance in some desired goal state. This objective might typically be achieved by the expert interacting directly with the system or simulating some realistic scenario for use [101]. The focus is on performance diagnosis, with the diagnosis depending heavily on the expert's intuitions.

Expert evaluations can provide a valuable integrated view of a system's performance in terms of usability, and can often be done quickly and cheaply. They do, however, carry a high element of risk: they often rely on the intuitions of a single individual and may therefore be incomplete or biased. It is difficult to validate such intuitive reports and the consequences of a poor evaluation may be costly.

7.3.2.2 Analytic Evaluation

Analytic techniques for evaluation rely on theories, often derived from those of cognitive psychology, and formal models of interaction. They may be used early on in a system's

development lifecycle as they can be done in the absence of real users or working prototypes. Typically, they provide predictions about system performance. An example of the kind of model which might be used in such an evaluation is the Keystroke Level Model (KLM) derived from the GOMS (Goals, Operators, Methods and Selection rules) model of a human-computer system for text editing [22]. The Keystroke Model can be used to predict execution time for optimal task performance, that is, for error-free performance by an expert user of a given program in a given task using a given method.

The advantage of such methods is that they are generally cheap to use and require few resources. They are, however, often extremely limited in scope [160]. Since models such as GOMS and KLM assume error-free performance, they are not appropriate for use in predicting performance in a real situation. Most have so far focussed on text processing tasks and very simple models of I/O devices and their applicability to more complex tasks carried out using more sophisticated hardware has not yet been demonstrated [23]. There is also a need for the psychological claims they embody to be made more explicit and therefore more testable [23]. It is currently not certain whether models used in analytic evaluations represent exactly the right sets of mental process and structures and further research is deemed to be needed in order to investigate whether this is indeed the case [23]. Finally, it has been observed that analytic techniques for usability evaluation are time-consuming and difficult to use [6] as there is often little or no support available.

Owing to the combination of these factors, analytic methods for evaluation are at present to be seen mainly as a subject or tool for research and are thought to be of little practical use in the field [153].

7.3.2.3 Empirical Evaluation

Empirical techniques for evaluation can be divided roughly into two main categories: those using experimental techniques and those based on the use of observation.

7.3.2.3.1 Experimentation Experimental techniques have their roots in the established paradigms of experimental psychology. They typically rely on the use of established psychological tools such as tests of free recall to assess, for example, the user's ability to remember what functions are implemented in a particular system and what commands are required to assess them [89].

Many variations on the basic arrangement are possible. The experimenter may choose to look at one or more of a variety of dependent variables such as speed and accuracy, and may use either pencil and paper simulations, partial prototypes or full working systems.

Experimental investigations have the advantage of being highly controllable, and can be conducted relatively quickly and cheaply. They provide objective and detailed data, and can be invaluable in testing out particular well-defined hypotheses. On the other hand, carrying out such formal experiments may take a great deal of time. Because of the controls which are necessarily imposed, findings may be incomplete or ungeneralisable: individual results may contribute little to the existing pool of human factors knowledge. Perhaps most important of all, the experimental methodology tells us little or nothing about which of the numerous variables which might have an effect in any particular situation should be formally investigated: the choice of the wrong set of variables can mean that any results are hard to interpret and that misleading conclusions may be

drawn. Experimental evaluations are suitable in some situations but run the risk of producing irrelevant or distorted results and must therefore be used with care.

7.3.2.3.2 Observation Observational evaluations typically involve recording the attempts of users, ideally from the target user population, carrying out one or more tasks either on a prototype or on a simulation of a working system. The number of users involved may vary and the tasks carried out may be either normal or 'critical' and either strictly or broadly defined.

There are many ways of collecting and analysing data from such a session. Audio and video recordings are commonly used, also system logs and post hoc interviews, discussions and questionnaires. Maguire and Sweeney [91] advocate the use of a variety of methods for collecting data which can then be mutually supportive. Such an integrated approach is facilitated by the construction of usability laboratories [151] containing multi-media facilities for data collection. Data analysis can then be done in a flexible manner and tailored to the particular needs of individual studies.

A particular technique for the empirical evaluation of usability which relies heavily on the use of observation is that of 'usability engineering' [152] which provides the engineer with a well-defined framework for evaluation involving checklists of measurement operations and criteria as well as advice on how to define operational specifications of usability.

7.3.2.4 The Ecological Approach

Improvements in usability yielded by approaches to evaluation such as those described above are still relatively small [152]. A number of authors have powerfully argued that in order to obtain better results, a further shift in emphasis is needed.

Several reasons for the failure of traditional methods for evaluation in producing large improvements in usability have been suggested. As Whiteside *et al.* have pointed out [152], traditional laboratory-based methods ignore important aspects of actual system use such as motivational, social, time and work constraints. Even when usability evaluations are performed in the field, the criteria for usability set by the human factors engineer may be far removed from those which are important to the actual user, so that the wrong things are measured. Usability should, it is argued, be 'an experiential relation between a user and an artefact' [155] so that in evaluating usability, we should focus on tapping the user's actual experiences. According to this view, the human factors specialist should no longer act as a scientist searching for universal objective truths, but as an engineer, aiming to produce an improved design, which will yield increases in usability, productivity and satisfaction. This new approach to usability evaluation, which recognises the importance of confronting realistic software situations on their own terms has been christened 'the ecological approach' [32].

An ecological approach to evaluation relies heavily on the use of qualitative, specific data rather than the quantitative and supposedly generalisable forms commonly generated by traditional experiments. The main vehicle for research is the case study. Case studies are typically carried out in the field, on site rather than in the laboratory and are used to collect detailed information about context and task performance. Interviews and think-aloud protocols are used to collect arbitrarily rich data which can then be analysed from many different perspectives. Large amounts of information are collected about relatively

small numbers of users and are analysed by looking at the semantics and pragmatics of individual cases, rather than by applying conventional statistical tests. The aim is not to confirm or disconfirm, or to test hypotheses, but to interpret and discover.

This kind of approach has been found to be highly successful in yielding 'artefacts and systems that are genuinely usable' [152]. There have been many successful examples of 'contextual field research', that is, research carried out in the field which places an emphasis on contextual information in assessing usability [44]. For example, Good [44] describes how contextual interviews were used by DEC in order to build up theories of usability, provide detailed information for the design of specific products and even to generate and develop ideas for new products. Poltrock [44] describes the use of semi-structured interviews by contextual researchers who first became familiar with clients' practices and techniques in order to gain a fuller insight into relevant contextual information.

While contextual research is usually carried out in the field, it is possible to borrow some of its techniques for use in the laboratory. Wright and Monk [156] describe the use of an ecological approach to evaluation of a bibliographic database system in the laboratory. Flexible, naturalistic data are collected from system logs both of free use and of set task performance. The investigator may question the user during task performance in order to determine his or her mental model of the system and assess overall understanding of operations available and information provided on the screen. The user is once again considered to be a co-evaluator.

The approach to evaluating the usability of ICDEDIT used in the case studies described in section 7.4 is an ecological approach and the particular methods employed are based closely on those suggested by Wright and Monk in [156].

7.4 Case Study Design

This section describes methods used both in and out of the laboratory in evaluating the utility of 3-d ICDs for knowledge engineering and the usability of depth-related features of ICDEDIT. These methods were heavily influenced by the aims of contextual field research as described above and an attempt was made to achieve ecological validity by focusing on real target users and real applications (see below).

An initial design for the case studies was developed and tested in a pilot study using a member of the project team who was already an experienced user of ICDEDIT as a subject. The pilot study is referred to here as study 1. After some slight refinements this design was carried forward for use in another five studies.

The rest of this section describes the concerns about ecological validity which were taken into account in designing the studies, and the structure and function of the three stages used in each study. Information about the circumstances in which each of the studies took place is summarised in tables 7.1 and 7.2. The results of the studies are reported in chapters 9 and 8, and the success with which their objectives were achieved is discussed in chapter 12.

		Study 1	Study 2	Study 3
Organisation	Nature of Involvement	Academic Research	Industrial Research	Academic Research
	Length of Involvement	4 years	5-10 years	7 years
	Current Involvement	6 people	30 people	10-12 people
Project	Size of Project	1.25 person years	6 person years	20 person years
Subject	Relevant Experience	4 years	2.5 years	2.5 years
	Number of Projects	0 - 5	0 - 5	3
	Hardware and Software Experience	Mainframes, Prolog	Mainframes, Workstations, PCs, shells, Lisp, Poplog	Macs, Suns, Prolog
Stage 1	Interview	1h 30m	2h 13m	2h 38m
Stage 2	Design	56m	45m	1h 52m
Stage 3	Observation	—	26m	32m
	Free Use	—	13m	11m
	Task Session	1h 12m	1h 13m	1h 55m
	Interview	19m	20m	48m
Total Time		3h 57m[1]	5h 10m	7h 56m

Note 1: Experienced ICDEDIT user: no training required.

Table 7.1: Details of Case Studies 1 - 3

		Study 4	Study 5	Study 6
Organisation	Nature of Involvement	Industrial Development	Industrial R & D	Industrial Research
	Length of Involvement	4 years	10 years	7 years
	Current Involvement	3 people	50 people	6 people
Project	Size of Project	6 person months	15 person years	1 person year
Subject	Relevant Experience	3 years	12 years	3 years
	Number of Projects	2	5	4
	Hardware and Software Experience	PCs	Mainframes, Workstations, tools, shells, Prolog	Workstations, Prolog,
Stage 1	Interview	1h 24m	1h 16m	43m
Stage 2	Design	29m	13m	30m
Stage 3	Observation	17m	31m	21m
	Free Use	7m	13m	17m
	Task Session	21m	31m	46m
	Interview	16m	13m	13m
Total Time		2h 54m	2h 57m	2h 50m

Table 7.2: Details of Case Studies 4 - 6

7.4.1 Ecological Validity

It was intended that data collected from the case studies should be as ecologically valid as possible. The development of a design for these studies which would maximise the ecological validity of the data they yielded entailed a number of considerations. These considerations are briefly discussed in the following paragraphs.

7.4.1.1 Users

All six of the subjects had considerable experience in the field of knowledge engineering, and all except one were from large organisations with considerable interest and experience in the use of knowledge-based systems (see tables 7.1 and 7.2). It was therefore felt that their opinions and abilities would be reasonably representative of those of knowledge engineers in general.

One of the subjects (number 1) was a member of the team who had been working on the development of JIN, JINGLE and ICDEDIT at City; two others (subjects 3 and 5) were from organisations with which contact had been made during the course of the project; and three (subjects 2, 4 and 6) were contacted through the project's industrial sponsors. Of the six subjects who took part, only two (subjects 1 and 3) had previously been aware of the interests of the project before agreeing to participate. It was therefore felt that subjects' opinions would not, in general, be unduly biased in favour of the use of 3-d ICDs.

7.4.1.2 Tasks

To achieve maximum ecological validity, these studies should ideally have evaluated the performance of real knowledge engineering tasks. At the time when this work was carried out, it was not yet feasible to connect ICDEDIT to a real knowledge-based system application. The performance of real tasks such as those involved in developing or debugging a knowledge base could not be supported and the ecological validity of findings from the case studies is therefore somewhat limited in this respect.

The functionality of the tool was, however, sufficiently well developed to allow the investigation of 'atomic tasks' [44] or fine-grained tasks, such as finding, editing or moving components of the knowledge base representation, of which real tasks are held to be composed. The performance of these tasks was observed with a view to obtaining fine-grained information about the usability of particular functions of interest in ICDEDIT.

7.4.1.3 Environments

To be ecologically valid, data about usability should ideally be collected in the setting in which the human-computer system under evaluation will operate.

Early interviews, aimed at collecting contextual information for use in evaluation of the utility of 3-d ICDs, were carried out on site in the setting in which subjects normally worked.

The final stages, however, involved the use of ICDEDIT. As ICDEDIT required a particular hardware configuration not available in every organisation visited, it was decided

that the final stage of a study should in each case be carried out in the evaluator's own office. Although this meant the loss of some ecological validity, it was still possible to collect flexible, naturalistic data in a way similar to that suggested by Wright and Monk [157, 156].

7.4.1.4 Systems

The 3-d ICDs developed and used during the course of these studies represented parts of knowledge structures from real knowledge-based systems with which subjects were currently or had recently been working. This lent a considerable degree of ecological validity to data about the utility and usability of 3-d ICDs for knowledge engineering.

7.4.2 Procedure

7.4.2.1 Stage 1: Contextual Interviews

Each case study began with an interview conducted in the knowledge engineer's normal place of work. This interview was used to obtain basic information about the subject, the knowledge-based systems he typically worked on, the tools he typically used, the tasks he performed and the environments in which he worked. The main topics to be covered were defined on the basis of consideration of reports of contextual interviews presented by Good [44]. The checklist of topics used by the interviewer is presented in appendix D.

At the end of the interview, the subject was asked to think of a knowledge-based system they had recently been working on, some structure of which would be suitable for representation as a 3-d ICD. Details of the system and the subject's role in producing that system were obtained and some initial ideas for the representation of the relevant structures were discussed. Any existing representations of other forms and any further relevant information about the system was presented to the interviewer.

Semi-structured interviews were used. Campbell *et al.* describe how the limitations on access to professional users mean that rigidly structured and drawn-out interviews are not appropriate and lead to boredom and frustration on the part of the user (see [44]). The flexibility allowed by a semi-structured interview means that the subject's time can be used as productively as possible. The pre-definition of certain topics facilitates communication by providing a focus for discussion and encouraging a systematic approach to information gathering. It is, however, possible to maintain a feeling of natural conversation which allows the user to interrupt or ask for clarification as necessary and means that unforseen aspects of importance can be discovered.

All interviews were tape-recorded and were carried out in accordance with the guidelines for general behaviour and question design outlined in [29].

7.4.2.2 Stage 2: Design and Development of 3-d ICD Representations

After the initial interview described above, the author used information about the knowledge-based system of interest together with the subject's proposals to develop initial ideas for representing the knowledge structure of interest in terms of 3-d ICDs. The main aim of

the second stage of the study was to discuss and develop these initial ideas. Interviews were again carried out in the knowledge engineer's place of work.

The knowledge engineer was first shown some examples for possible schemes for knowledge representation using 3-d ICDs, in order to give some feeling for the way in which they might be used. A brief description of the graphical primitives supported by ICDEDIT and their possible uses was also given (see appendix D). The possible uses of the third available dimension were discussed: it was explained that depth could either be used as a means of representing larger amounts of information within the same visual space (*ie* on one screen), or as a means of representing or coding a variable or ordering on the entities in the knowledge structure. Some examples of this latter use were given.

Once the basic aims of the session were clear and the knowledge engineer had a reasonable grasp of the kinds of representation that might be feasible, it was possible to discuss representation of the particular structure of interest in more detail. Some structure was imposed on this discussion by asking the knowledge engineer to describe what were the important entities to be represented, what were their attributes, and what the relationships between them were. Once this had been decided, concrete representations for each of the important features were chosen and a scheme for the use of spatial information was suggested.

Designs created in this way are shown in appendix G.

7.4.2.3 Stage 3: Atomic Task Performance

In the final stage of each study, the subject came to the author's laboratory to be trained to use ICDEDIT and was then asked to carry out a series of low level 'atomic' tasks using the representation developed in stage 2. The subject's performance in these tasks was observed and recorded in order to assess the usability of ICDEDIT in a manner similar to that proposed by Wright *et al.* [157]. The session ended with a third interview in which the subject's reactions to ICDEDIT and to the possibility of using 3-d ICDs in the interface of tools for knowledge engineering were assessed.

Tasks to be performed by particular subjects were derived from the generic task taxonomy presented in appendix D by specialising particular tasks according to the representation to be used. For example, the generic task 'find a single entity in terms of its position relative to other entities' was specialised to 'find a non-instantiable element which referentially contains an instantiable element' in case study 2 where different types of node were used to represent instantiable and non-instantiable elements of the knowledge structure and the referential containment relationship was represented in depth.

Tasks performed had to be relatively simple owing to the prototypical nature of ICDEDIT and the fact that it was not connected to any real knowledge engineering tool or knowledge-based system. The kind of tasks specified in the generic task taxonomy were, however, thought to be the 'atomic tasks' [44] of which real knowledge engineering activities carried out using 3-d ICDs would be composed.

The basic method used in investigating task performance was similar to that described in the York manual for co-operative evaluation [157]. When the subject arrived, he was informed of the basic aims and structure of the session. He was asked to think of himself as contributing to the process of evaluation and encouraged to give as much feedback about the system and the graphical representation as possible, regardless of how critical

it might be. He was also assured that any mistakes he made or problems he encountered were the fault of the system, which was still at the stage of an early prototype.

The first part of the session involved training the subject to use ICDEDIT well enough to perform the atomic tasks. It was decided that training should be relatively informal. This was because many of the subjects already had experience of similar graphical tools, and it was felt that an overly rigid training program could take up unnecessarily large amounts of time and could cause subjects to get bored and frustrated.

Training was therefore carried out as follows. First of all, the investigator demonstrated the complete functionality of the tool using a prompt sheet to ensure that this was done in a structured way and that nothing was omitted (see appendix D). The subject was free to ask questions throughout the demonstration and dialogue between the investigator and the subject was unstructured and informal at this point. The subject was then allowed free use of the system and was encouraged to work with a specially prepared training diagram (see figure 7.1) until he felt happy that he had mastered the system sufficiently well to perform some simple tasks.

Once the subject was happy with the system, the task session was begun. The subject was first shown a key explaining the representational scheme which had been developed for his application. The key used in study two is shown in figure 7.4.3 as an example. The procedure for the task session was then explained.

As described above, the investigator had prepared a set of representation-specific atomic tasks on the basis of a taxonomy of generic tasks intended to cover the whole range of relevant atomic tasks in a structured way. Instructions for individual tasks were read out by the investigator, and the subject was asked to carry them out, while telling the investigator as much as he could about what he was trying to do and why. According to Ericsson and Simon [50], this kind of concurrent verbalisation can yield valid information about a subject's strategies and beliefs about a system as long as no requests are made for information which would not otherwise have been attended to.

The investigator was mainly silent while the subject performed the tasks, except for occasional prompts when the subject fell silent (*eg* 'Can you tell me what you are doing now?'). The subject was asked to complete tasks without assistance whenever possible and to indicate to the investigator when he was satisfied that he had done so. The investigator would then give feedback as to whether the task had been correctly carried out. In cases where the subject was unable to complete the task, the investigator would enter into a dialogue with the subject in order to try to determine the cause of the failure, and then to provide the necessary information for the subject to carry on with the task or to redo it in cases where the first attempt had been incorrect.

Tasks in all studies except the pilot were performed using a dummy version of the graphical representation developed in stage 2 in which any specific textual labels used in these dummy representations had been re-coded in an attempt to prevent subjects from using their existing knowledge of the structure represented in performing the tasks. The dummy representation developed for use in the second study is shown in figure 7.4.3 as an example.

After the task session, the subject was asked to participate in a final interview in order to give feedback on his overall impressions of the usability of ICDEDIT and of the utility of 3-d ICD interfaces for knowledge engineering in the light of some experience with their use. Finally, they were shown the actual representation of the knowledge structure of interest

(as opposed to the dummy version with which they had been working) and were asked to comment on the success or otherwise of the scheme developed. The representation of the knowledge structure of interest in the second study is shown in figure 7.4.3. Specific questions were again used to determine whether anything had been represented in a misleading way, or whether anything looked wrong or had been missed out.

7.4.3 Analysis of the Results

Data from the case studies described above was used to assess the usability of depth-related features of ICDEDIT and to estimate the utility of 3-d ICDs in the interface to tools for knowledge engineering.

The usability of features of ICDEDIT was assessed on the basis of observations of subjects' performance in the task sessions described in section 7.4.2.3 as well as the verbal protocols recorded during task performance and the interview protocols recorded at the end of the session. Data of this kind was analysed in a qualitative manner as proposed in section 7.3.2.4. It was felt that a quantitative analysis of, for example, numbers of errors or times taken by different subjects to perform similar tasks, would not be appropriate in this case. Different subjects worked with different representations in each of the case studies and tasks were slightly different in each case (see section 7.4.2.3) so comparison of data from different studies would not have been valid. The number of independent variables which might potentially be considered is also huge. In any case, it was felt that an open-ended qualitative analysis would yield more valuable insights into the kinds of changes in design which would be required to improve the usability of future versions of ICDEDIT. Findings and recommendations concerning the usability of depth-related features of ICDEDIT are presented in chapter 9.

The utility of 3-d ICDs for knowledge engineering was estimated partially on the basis of observations of task performance, but also on the basis of comments made by the knowledge engineers in interviews in the first and last stages of the study. Information obtained in the first stages was used as the basis for creating the framework within which the utility of 3-d ICD interfaces (relative to their 2-d counterparts) was estimated for relevant categories of user, task, environment and knowledge-based system as suggested by the definition given in section 7.2.1. Estimations of utility derived in this way are presented in chapter 8.

7.5 Summary

Definitions of utility and usability were presented. Methods used by human factors engineers in carrying out usability evaluations were briefly reviewed and the ideas behind the recently developed 'ecological approaches' were introduced. The method used in case studies aimed at investigating the utility in knowledge engineering of an interface technology based on the use of 3-d ICDs, and the usability of depth-related features of ICDEDIT was then described.

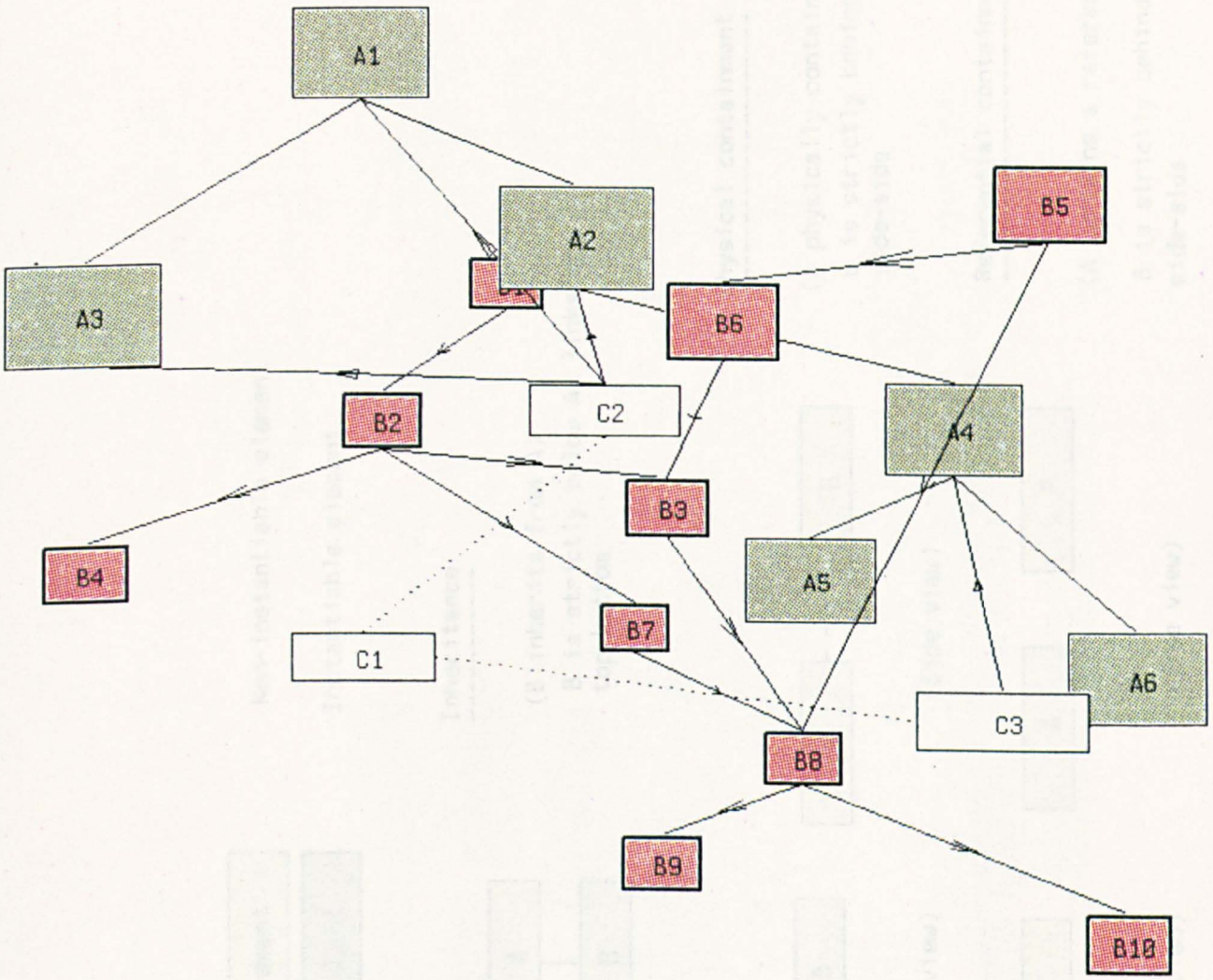


Figure 7.1: Training Diagram

Figure 7.2: Key Describing the Visual Language Used in Case Study 2

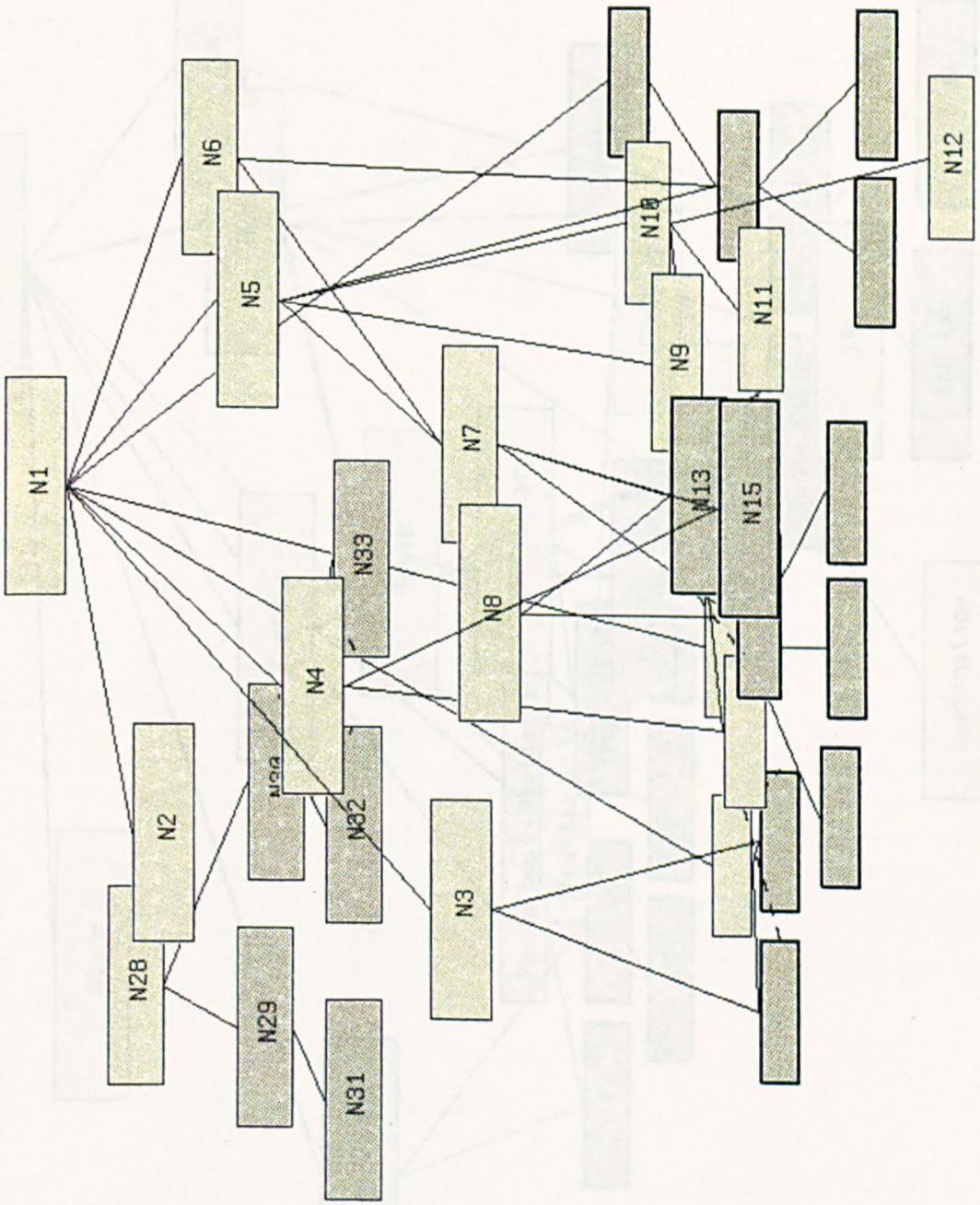


Figure 7.3: Dummy 3-d ICD Representation Used in Case Study 2

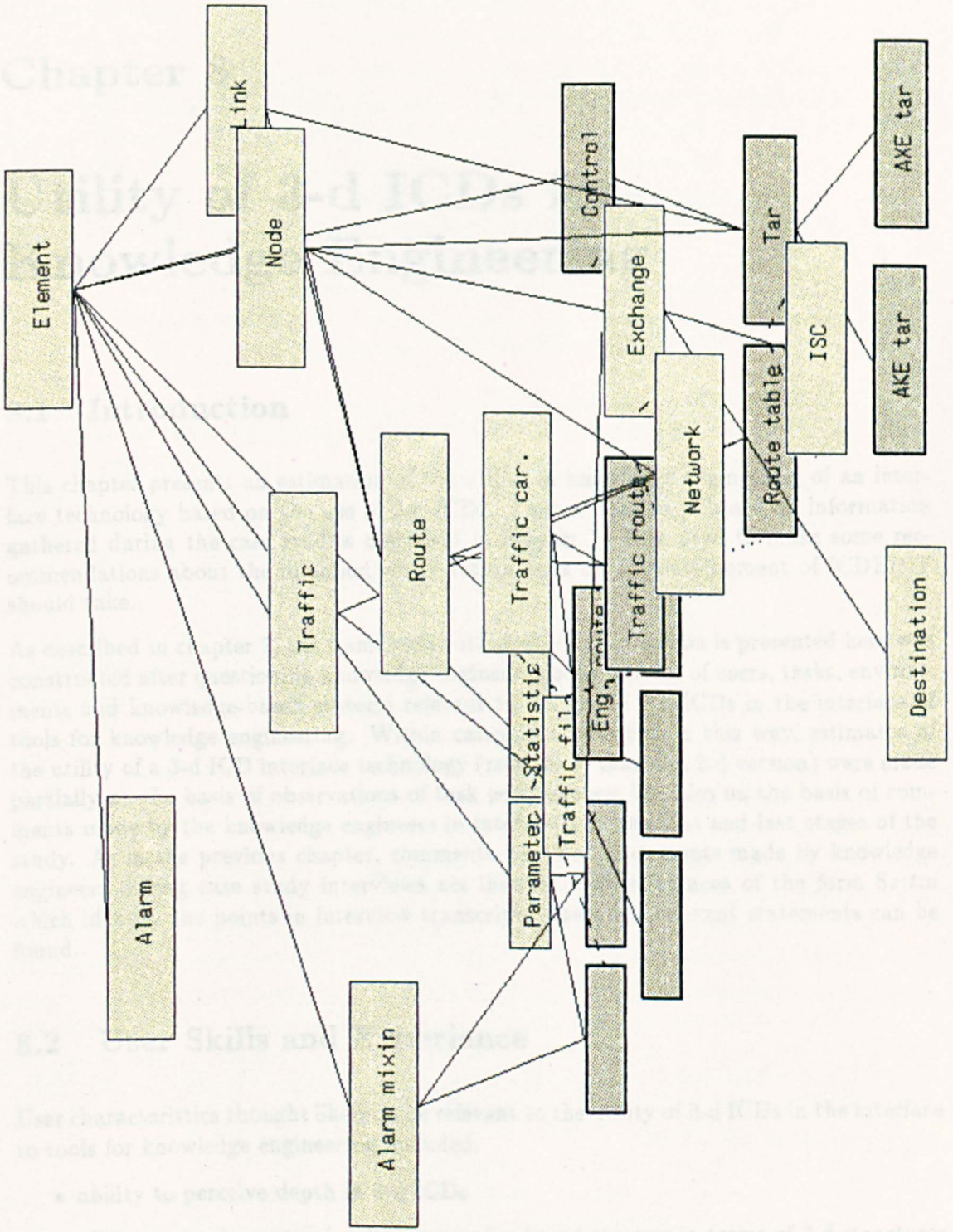


Figure 7.4: Actual 3-d ICD Representation Used in Case Study 2

Chapter 8

Utility of 3-d ICDs for Knowledge Engineering

8.1 Introduction

This chapter presents an estimation of the utility in knowledge engineering of an interface technology based on the use of 3-d ICDs. This estimation is based on information gathered during the case studies described in chapter 7. It is used to make some recommendations about the direction which future work on the development of ICDEDIT should take.

As described in chapter 7, the framework within which information is presented here was constructed after questioning knowledge engineers about aspects of users, tasks, environments and knowledge-based systems relevant to the use of 3-d ICDs in the interface of tools for knowledge engineering. Within categories identified in this way, estimates of the utility of a 3-d ICD interface technology (relative to that of a 2-d version) were made partially on the basis of observations of task performance, but also on the basis of comments made by the knowledge engineers in interviews in the first and last stages of the study. As in the previous chapter, comments based on statements made by knowledge engineers during case study interviews are indexed with references of the form S*s*:*t*:*u* which identify the points in interview transcripts where the relevant statements can be found.

8.2 User Skills and Experience

User characteristics thought likely to be relevant to the utility of 3-d ICDs in the interface to tools for knowledge engineering included:

- ability to perceive depth in 3-d ICDs
- ability or tendency to visualise knowledge-based systems in terms of 3-d structures
- degree of experience with tools for knowledge engineering using 2-d graphical interface technologies
- degree of experience with knowledge engineering

This section will consider whether there is any evidence from the case studies to indicate whether, for example, the utility of 3-d ICD interfaces depends on users having particular perceptual capabilities or particular skills in visualisation, or whether such interfaces will only be useful to users with a high degree of experience of 2-d graphics or of knowledge engineering in general.

Some of the factors identified above were, to some extent, confounded in the case studies. For example, the subject with the longest experience of knowledge engineering (subject 5) also had relatively wide experience of different kinds of hardware and software for knowledge engineering and more experience of tools using 2-d graphical interfaces than all other subjects except subject 3 (see tables 7.1 and 7.2). Despite these interactions, an attempt is made to consider the relevance of each factor in isolation in the following paragraphs.

8.2.1 Perceptual Capabilities

As reported in the previous chapter, all subjects appeared able to perceive and use information represented in depth. Subject 2 experienced some difficulties, for example with the use of perspective as described in the previous chapter. Most of his difficulties were, however, caused by problems with the facilities for diagram manipulation: *'if it wasn't for the initial barriers the interface has got, the actual diagram itself is helpful'* (S2:3c:527), and despite these difficulties, he was still able to correctly identify nodes described only in terms of their position in depth relative to other nodes.

Summary: A small number of users may experience difficulties in perceiving depth in 3-d ICDs of the kind generated by ICDEDIT. Such difficulties appear not to be likely to undermine the utility of 3-d ICD interfaces to a great extent.

8.2.2 Visualisation Skills

One factor judged to be important in considering the utility of abstract 3-d graphical representations of information is that of the user's skills in visualisation. There is little point in providing users with a 3-d graphical editor if they are unable to visualise or reliably construct the 3-d structures of interest. Many of us have been taught in school to understand and work with various kinds of 2-d graphical representations such as maps, plans and graphs, but the ability to understand and work with 3-d representations of anything other than the physical objects with which we are familiar is much less often taught. The ability of individuals to *visualise* or think in terms of abstract 3-d structures is correspondingly less common than that of being able to think in terms of 2-dimensional representations [104].

Three of the six subjects in the case studies tended naturally to think of knowledge-based systems in graphical terms. S4 claimed to *'visualise it all in pictures; well in blocks linked by arrows'* (S4:1a:482), and S1 said he would have *'some sort of vague spatial thing in mind about a tree'* (S1:1a:212). One subject reported having thought previously of knowledge-based systems in terms of an abstract 3-d representation (S5:3b:700). These findings suggest that tools for knowledge engineering which support such visualisation might be useful to a substantial proportion of knowledge engineers: *'for me, it would*

probably be a very useful way of representing knowledge' (S4:1b:9). A smaller proportion are likely to find 3-d interfaces more useful than 2-d in this respect.

It should also be noted that all subjects were able to suggest and discuss ideas for 3-d representations of knowledge structures of interest during the second stage of a study. This suggests that a wider range of users might develop ways of using 3-d representations once given the tools to do so. As subject 4 commented after seeing a 3-d representation of his knowledge structure for the first time: *'once you're looking at it like this it's really easy because you don't have to try and visualise how it looks, it's as if it's just there'* (S4:3a:713).

Summary: Some users tend naturally to think of knowledge structures in terms of 2- or 3-d graphical representations. These users would be likely to find graphical tools for knowledge engineering useful. In particular, users who think naturally of knowledge structures in terms of 3 dimensions would be likely to find 3-d graphical representations more useful than 2-d. Other users may also find uses for 3-d tools if they were provided.

8.2.3 Experience with Graphical Representations for Knowledge Engineering

Subjects had varying degrees of experience with 2-d graphical representations for knowledge engineering. All had seen examples of 2-d representations produced using commercial tools but only subject 5 had made extensive use of this kind of representation. Subjects 2, 3 and 6 had used or experimented with 2-d representations to some extent but subjects 1 and 4 had no real experience of using 2-d diagrams in the development of knowledge-based systems.

The results of the case studies suggested that a high degree of previous experience of 2-d representations might cause initial problems with the use of 3-d representations. Subject 5 reported that after extensive experience of working in 2-d, working with a knowledge structure displayed in 3-d took some getting used to: *'It's very easy to think in 3 dimensions, and it's very easy to think in 2, but viewing something that you know, that you're used to in 2 dimensions, and having a 3-d capability is quite difficult'* (S5:3a:640). By the end of the task session, he felt *'quite comfortable working in 3-d'* (S5:3c:25), though he felt that he would need to work with 3-d representations a little longer *'to really be able to get a feeling for it'* (S5:3b:675).

There was no marked difference between the performance of subjects who had some experience of 2-d representations and those who had none at all.

Summary: Users with a high degree of experience with 2-d representations may initially experience difficulties with the use of 3-d (and therefore find 3-d diagrams less useful than 2-d). Such difficulties are, however, likely to be surmounted after a relatively small amount of practice with 3-d representations.

8.2.4 Experience with Knowledge-Based Systems

All subjects except one had between 2.5 and 4 years of experience in the field of knowledge engineering and the remaining subject had 12 years' experience. There were no obvious

correlations between degree of experience and either judgements made about the utility of 3-d representations or the capacity to use and understand such representations in the task sessions.

It should be noted that since all subjects were relatively experienced in the field of knowledge engineering, the studies provide no direct information regarding the utility of 3-d ICDs for novice knowledge engineers or the relative utility of such representations for experts and novices respectively. Subjects' comments provide indirect support for a claim such as that often made in favour of 2-d representations which is that they should be of particular assistance to naive users in understanding knowledge-based system structure and functioning (see chapter 3). For example, although S4 thought initially that a 3-d ICD representation of a knowledge structure would *'totally bemuse'* the naive end user (S4:1b:340), he eventually became quite enthusiastic about the idea of using the design developed in his case study to explain backward chaining to an inexperienced knowledge engineer (S4:2a:385).

Other arguments made concerning the use of 2-d representations by expert and novice knowledge engineers may also apply to the use of 3-d versions. For example, the lack of a good understanding of the issues involved in knowledge-based system development may mean that novice knowledge engineers would find it hard to construct a useful 2-d graphical representation of a knowledge structure of interest, but would derive considerable benefits from viewing one which had already been developed. 3-d representations may be even harder to construct for the novice engineer, but may, on the other hand, be even more useful in conveying an understanding of the structures of interest if developed by some other party. Expert knowledge engineers are likely to find it easier to construct graphical representations than novices (this suggestion is supported by the fact that the subject with considerably more experience than the others – subject 5 – was the only one to have thought previously of representing knowledge structures in terms of 3-d structures), and while 3-d representations might still be harder to construct than 2-d versions, the understanding gained as a result of the process of design and development of a suitable representation might be greater in the case of 3-d than that of 2-d. Further research is needed to investigate these issues.

Summary: Experienced users may be more likely to generate 3-d representations than others. Information from the case studies suggests that the ability of subjects to use and understand 3-d knowledge structure representations may not depend on whether the user has moderate or high experience of knowledge engineering. The case studies yield no direct information concerning the utility of 3-d representations for novice knowledge engineers.

8.3 Task Factors

This section considers the extent to which 3-d ICD interfaces will be able to provide useful support for a variety of high level knowledge engineering tasks. At the time when these case studies were carried out, there was still no widely adopted methodology for knowledge-based system development. The strategy of most organisations visited was to use some form of prototyping: subject 1 had used *'evolutionary'* prototyping, subject 2 *'rapid'* prototyping, subject 5 *'incremental'* and subject 6 *'iterative'* prototyping. According to some subjects though, this often amounted to little more than 'hacking'

User Characteristics	
Perceptual capabilities	A small number of users may experience difficulties in perceiving depth in 3-d ICDs of the kind generated by ICDEDIT. Such difficulties appear not to be likely to undermine the utility of 3-d ICD interfaces to a great extent.
Visualisation skills	Some users tend naturally to think of knowledge structures in terms of 2- or 3-d graphical representations. These users would be likely to find graphical tools for knowledge engineering useful. In particular, users who think naturally of knowledge structures in terms of 3 dimensions would be likely to find 3-d graphical representations more useful than 2-d. Other users may also find uses for 3-d tools if they were provided.
Experience of KBS graphics	Users with a high degree of experience with 2-d representations may initially experience difficulties with the use of 3-d (and therefore find 3-d diagrams less useful than 2-d). Such difficulties are, however, likely to be surmounted after a relatively small amount of practice with 3-d representations.
Experience with KBSs	Experienced users may be more likely to generate 3-d representations than others. Information from the case studies suggests that the ability of subjects to use and understand 3-d knowledge structure representations may not depend on whether the user has moderate or high experience of knowledge engineering. The case studies yield no direct information concerning the utility of 3-d representations for novice knowledge engineers.

Table 8.1: Effects of User Skills and Experience on Utility of 3-d ICDs for Knowledge Engineering

(S2:1a:310, S5:1b:53) in the absence of computer-based support for a more structured development procedure.

A number of commonly recognised stages in the knowledge-based system lifecycle are considered in this section. The aim is to discover whether 3-d ICDs might be more useful at some stages than others, and thus whether further development of tools for knowledge engineering should be directed towards providing 3-d ICD interfaces to support particular knowledge engineering tasks.

Stages in the knowledge-based system lifecycle considered here are:

- knowledge acquisition
- knowledge validation
- knowledge-based system design
- knowledge-based system development
- knowledge-based system debugging
- knowledge-based system verification
- knowledge-based system maintenance

Since tasks performed during the case studies were atomic tasks, they provide no direct evidence (in terms of observations of performance) for the relative utility of 2- and 3-d representations at each stage. The conclusions of discussions in the following paragraphs do, however, have some significance as they are based on comments and observations made by real potential users with some experience of both the tasks of interest and the use of 3-d ICDs.

8.3.1 Knowledge Acquisition

The knowledge engineer's first task at the beginning of a knowledge-based system project is often to become familiar with the domain of interest. Many of the existing tools for knowledge-based system development provide little support for this process and the knowledge engineer is often reduced to simply *'sitting down with bits of paper and hacking things around'* (S6c:1a:575).

Four of the six subjects stated that diagrammatic representations would be helpful at this point. While subject 2 thought that his understanding would at first be too vague to be set down in any form, he thought that a diagrammatic representation might become more useful as his understanding became clearer, and indeed found that the process of discussing the design of a 3-d graphical representation of the system on which he was currently working (during stages 1 and 2 of the case study) lead to a greater insight into what he was trying to achieve (S2:2b:500). Subject 3 reported how, in the absence of a specialised tool for knowledge acquisition, he had used a stand-alone graphical editor to create 2-d diagrams to help him distil knowledge about the domain of interest from technical papers on the subject (S3:1a:500). Subject 5 had also used data flow diagram and decision tree notations in knowledge acquisition (S5:1a:550). Subject 6b also thought that a graphical tool could be very useful for *'organising your thoughts and getting the structure down'* (S6b:1a:620). Finally, subject 6d felt that it would be *'easier to use a*

diagrammatic paper model of the process and a graphical tool than to go through interviewing and transcript analysis' (S6d:1a:470) and suggested that such a diagrammatic model might form the basis for interactive knowledge acquisition sessions directly involving the expert (S6d:1a:520). It seems, then, that computer-based support for the use of some form of graphical representation could help knowledge engineers to structure their thoughts early on in knowledge acquisition.

The capabilities of a tool supporting the development of full 3-dimensional representations may not be fully exploited at this stage. As suggested by subject 2, the knowledge engineer's understanding is likely still to be quite vague, and dimensions which would form a suitable basis for overall structuring of the knowledge gathered may not yet have been identified. It may therefore not yet be appropriate to use spatial coding in a meaningful way.

The ability to cluster nodes in the graphical representation to a greater extent than would be possible using tools supporting strictly 2-dimensional representations (for example, by overlapping nodes) may, however, be very useful. The knowledge engineer is likely to want to consider large amounts of information at this stage, and to be able to see as much of it as possible in the same view: the ability to overlap nodes would facilitate the display of larger amounts of information than would be possible with strictly 2-dimensional representations. Furthermore, the use of overlapping would facilitate the creation of clusters of nodes representing conceptually related entities which might also be useful in this initial structuring of knowledge.

Summary: Computer-based support for the use of graphical representations in knowledge acquisition was felt by 3 of the 6 subjects to be desirable, but none felt that meaningful use of 3-d representations would be especially useful at this stage. It is, however, suggested here that limited 3-d facilities, such as support for clustering, may be useful in performing the tasks users described as necessary at the stage of knowledge acquisition.

8.3.2 Knowledge Validation

Two of the subjects suggested that diagrammatic representations of some form might be useful in discussions with experts, aimed at validating the knowledge engineer's developing understanding of the domain. Subject 1 suggested that diagrams might be used in discussions with the expert (S1:1b:75) and subject 6b suggested working interactively with the expert to produce a diagrammatic model of the knowledge or process of interest as described above (S6b:1a:470). Previous attempts of subject 4 to use diagrammatic representations in discussions with users had, however, not always been successful due to the users' lack of understanding of the representational paradigms employed (S4:1b:199). The approach suggested by subject 5's organisation to validation was to discuss early prototypes of the proposed system rather than abstract representations of any kind (S5:1a:250).

Some of the arguments already presented (for example those in section 8.2.4 concerning the use of 3-d ICD representations by novices as opposed to experts, and those in section 8.3.1 concerning the use of 3-d representations early on in knowledge acquisition) may again be applied here. Domain experts, with whom knowledge gathered for use in the developing system must be checked, are likely to be a novices with respect to the use of 3-d ICDs. They may therefore find it hard to *construct* such diagrams. Their knowledge

of the domain should, on the other hand, make it relatively easy for them to *comprehend* and check a 3-d ICD representation of salient features in the domain. Graphical representations (especially those based on the use of novel paradigms such as 3-d ICDs) should, however, be used with caution at this stage. It may be difficult for experts unfamiliar with the use of 3-d ICDs to spot that an inappropriate relation or dimension has been chosen as the basis for overall structuring. They may assume that a representation 'looks wrong' simply because they are not yet familiar with the conventions of 3-d ICDs.

Summary: Two of the six subjects suggested that graphical representations would be useful in knowledge validation, but a third stated that he had found them unsuitable in the past and a fourth described how his organisation preferred to validate the performance of a prototype. No subjects felt that 3-d ICD representations would be especially useful at this stage.

8.3.3 Design

Once the knowledge engineer has achieved and validated an understanding of the relevant knowledge, the next phase of activity involves producing high-level designs for the system and the knowledge structures on which it will be based.

Three subjects suggested that some form of diagrammatic representation would be useful at this stage. Subject 4 stated that graphical notations were useful in developing the system's logical structure (S4:1b:340), especially in the case of complex systems (S4:3b:189). Subjects 5 and 6d both mentioned the utility of graphical representations in entity relationship modelling techniques (S5:1a:665, S6d:2a:390), and subject 5 had also found data flow diagrams useful at this stage (S5:1a:660). Subject 5 stressed the utility of computer-based support for the use of graphical representations at this stage: if entity-relationship models *'could be captured on something other than paper, it would encourage people to BOTHER – to be more structured rather than hacking so you never know what you've got'* (S5:1b:53).

Subject 5 (the most experienced of the subjects) claimed that he had in the past wanted to lay out diagrams of the kind used at this stage in three dimensions: he stated that he would have liked to show increasing detail in data flow diagrams in depth (S5:1b:130). He also envisaged that it would be useful to combine graphical representations of inheritance and abstraction hierarchies in a single structure, by displaying the inheritance hierarchy from left to right and the abstraction hierarchy going into the screen (S5:1b:135). He thought that 3-d representations might at this stage be useful in a range of situations in which the knowledge engineer needs to be aware of the relations between two different knowledge structures: *'I think it might help to do that quite valuably'* (S5:3c:15).

Summary: Three out of the six subjects had found graphical representations such as entity-relationship and data flow diagrams useful in designing knowledge-based systems. Subject 5 said that he would find 3-d data flow diagrams and representations of abstraction and inheritance hierarchies useful at this stage.

8.3.4 Development

This section presents subjects' comments on the utility of 3-d ICDs in knowledge-based system development *ie* in turning high-level designs into code.

One subject was definitely in favour of the use of graphical representations in building or implementing a knowledge-based system. Subject 6d had developed a system using the facility provided in KEATS (see *eg* [48]) for the generation of knowledge-based system code from a 2-d node and link representation and had found this to be an extremely efficient way of working. He had found that the graphical notation effectively acted as shorthand for the relevant code and enabled him to do what would have been a large amount of work in a relatively short space of time: *'this is six weeks' work done in KEATS in ten minutes'* (S6d:2a:315). A graphical editor of this kind was in his view *'a productivity tool'* (S6d:2a:416).

Other subjects were more sceptical. Subject 3 thought that suitable visual programming paradigms might one day be developed but that existing tools lacked the necessary functionality (S3:3d:680). Subject 2 thought that diagrammatic representations of knowledge-based system code would be harder to build than textual versions, especially in the case of 3-d diagrams (S2:3c:630) but suggested that it would nevertheless be useful to be able to see a graphical representation of the system alongside the conventional code, provided such a representation could be generated automatically (S2:1b:19): *'I like being able to see some kind of graphical representation of the kind of knowledge structures you've built ... because it's a hell of a lot easier than having to scroll through hundreds and hundreds of lines of code to try and find out'* (S2:3b:470). However, even in graphical representations created automatically, it was thought that complexity could easily become a problem if all the relevant relations between entities were displayed simultaneously (S2:2a:134). This issue will be discussed further in section 8.5.

Again, it seems as though node and link representations may have a role to play in supporting visual programming or development of knowledge-based systems. Further research is, however, needed in order to determine the way in which graphical representations can most usefully be exploited at this stage, and in particular to investigate whether there is a distinctive role for 3-dimensional representations.

Summary: Two-dimensional node and link representations were found by one subject to be extremely useful at this stage. No subjects suggested that there might be a distinctive role for 3-d diagrams.

8.3.5 Debugging

Several subjects thought graphical representations of relevant knowledge structures would be useful during the debugging phase. Subject 3 thought that graphical representations would aid debugging, provided useful layouts could be found and produced automatically from the code (S3:3e:250). Subject 5 suggested that meaningful spatial (*ie* three-dimensional) organisation of elements would enable knowledge engineers to spot significant patterns, for example in variable bindings during the running of a Prolog program which might be represented *like a mountainous landscape with the tips being variable bound first and everything bound by the bottom'* (S5:1b:280). This would, he thought, highlight inefficiencies which could otherwise have gone unnoticed.

For rule-based systems, subject 6d thought that a graphical representation of a rule network would be useful in debugging in allowing knowledge engineers to check that the firing sequence was correct (S6d:2a:418). Subject 4 thought that such a representation would be useful in checking that all possible combinations of circumstances or routes through the network had been accounted for (S4:1a:690). He thought it would also be useful in checking the logic of what had been coded (S4:1a:580). He estimated that 20 - 30% of his time was currently spent in attempting to carry out such checks using *'scraps of paper with pictures on'* and that computer-based support for such tasks would therefore lead to large increases in productivity.

The fact that 4 out of 6 subjects were enthusiastic about the possibility of using graphical tools during debugging suggests that this would be a fruitful area for further research. A considerable body of literature on program visualisation and the use of graphical tools for debugging already exists, but while animation has been considered as a way of enhancing conventional 2-d diagrams, there has been less work on the use of 3-dimensional representations for such purposes.

Summary: Four of the six subjects thought that graphical representations would be useful in debugging. One subject suggested a way in which 3-d representations might be used. Support provided by graphical tools at this stage is likely to be valuable in terms of productivity increases, though the extra benefits of using 3-d representations are less certain.

8.3.6 Verification

Once the system has been built, it must be checked in order to verify that it performs as expected. This often involves consultation with the expert or intended end-user. Subject 3 suggested that graphical representations of the knowledge structures used would be appropriate at this stage in complementing demonstrations of the software itself (S3:3e:240). Arguments concerning the relative merits of 2 and 3-d representations as media for communication with experts have already been discussed (see section 8.3.2).

Summary: One subject suggested that graphical representations might be useful at this stage. No subjects thought that 3-d would be especially useful at this stage.

8.3.7 Maintenance

One subject thought that the availability of 3-d ICD representations of the relevant knowledge structures should ease the task of maintenance for knowledge engineers, or even end-users who had not been involved in the development of the system (S6d:2a:660). Arguments concerning the use of 2 and 3-d representations as media for communication with naive users of knowledge-based systems (presented in section 8.2.4) are again relevant here.

Summary: One subject suggested that 3-d ICD representations would be useful at this stage.

Tasks	
Knowledge acquisition	Computer-based support for the use of graphical representations in knowledge acquisition was felt by 3 of the 6 subjects to be desirable, but none felt that meaningful use of 3-d representations would be especially useful at this stage. It is, however, suggested here that limited 3-d facilities, such as support for clustering, may be useful in performing the tasks users described as necessary at the stage of knowledge acquisition.
Knowledge validation	Two of the six subjects suggested that graphical representations would be useful in knowledge validation, but a third stated that he had found them unsuitable in the past and a fourth described how his organisation preferred to validate the performance of a prototype. No subjects felt that 3-d ICD representations would be especially useful at this stage.
Design	Three out of the six subjects had found graphical representations such as entity-relationship and data flow diagrams useful in designing knowledge-based systems. Subject 5 said that he would find 3-d data flow diagrams and representations of abstraction and inheritance hierarchies useful at this stage.
Development	Two-dimensional node and link representations were found by one subject to be extremely useful at this stage. No subjects suggested that there might be a distinctive role for 3-d diagrams.
Debugging	Four of the six subjects thought that graphical representations would be useful in debugging. One subject suggested a way in which 3-d representations might be used. Support provided by graphical tools at this stage is likely to be valuable in terms of productivity increases, though the extra benefits of using 3-d representations are less certain.
Verification	One subject suggested that graphical representations might be useful at this stage. No subjects thought that 3-d would be especially useful at this stage.
Maintenance	One subject suggested that 3-d ICD representations would be useful at this stage.

Table 8.2: Utility of 3-d ICDs for Various Knowledge Engineering Tasks

8.4 Environmental Factors

This section considers whether 3-d ICD interfaces are particularly suited for use in particular kinds of environment. It considers aspects of:

- the organisational environment:
  - degree of communication with project management
  - degree of communication within the project team
- the physical environment
- the computing environment:
  - hardware
  - software

Once again, there is likely to be some interaction between these things. For example, the organisational environment will, to a large extent, determine what the computing environment will be like, in that larger organisations are more likely to be able to invest in more advanced computing facilities. For the sake of clarity however, an attempt has been made to discuss the significance of different factors in separate sections.

8.4.1 Organisational Environment

8.4.1.1 Communication with Management

As knowledge-based system technology matures, knowledge-based system projects are increasingly being integrated into the mainstream computing activities of a number of companies, and as such are being subjected to increasingly rigorous project management (S2:1a:425, S4:1a:250, S5:1a:210). This often means an increase in the involvement of managers from outside the project who need to be able to understand how development is progressing. There is therefore a need for a means of communication regarding knowledge-based system structure and development which non-specialists can easily understand.

It has been argued that 2-d representations go some way towards meeting this need (see chapter 2). The addition of a third dimension means that more information may be conveyed in a single representation. However, none of the subjects felt that 3-d representations would be especially useful in communicating with project managers.

Summary: There is no evidence to suggest that 3-d ICDs would be useful in communicating with project managers.

8.4.1.2 Communication Within the Project Team

Knowledge-based system projects vary enormously in size. Some subjects' organisations were still involved in relatively small projects (eg 6 person-months, S4:1a:150; 12 - 18 person-months, S6:1a:220), whereas others were involved in multi-million pound projects which might consume 20 person-years or more (S3:1a:520). In such large projects involving large numbers of people, finding an efficient means by which the various members of the team can communicate with each other about the work they are doing becomes very

important. One subject estimated that between 10 and 25% of his time was spent in discussions with other members of the group (S3:1a:300), and in larger projects, this percentage can increase dramatically to the detriment of the project as a whole (S6b:1a:250).

Graphical representations can be useful in such situations. As argued above (see chapters 2 and 3), an understanding of complex structures such as those at the heart of knowledge-based systems can be grasped much more readily from a graphical representation than from pages of code. It seems quite likely that 3-d representations would offer more than 2-d here. Individuals involved in a project over a period of time would be able to become confident and familiar with useful forms of representation, and could benefit from the extra freedom of expression which the use of 3-d provides. However, no subjects proposed that 3-d ICD representations should be used in this way.

Summary: There is no evidence to suggest that 3-d ICDs would be useful in communicating with other members of a project team.

8.4.2 Physical Environment

Two subjects stated that they worked mainly in their own offices (S2:1a:448, S5:1a:233), though there was sometimes a need to go out to the client's offices, often during knowledge elicitation or acquisition. Given the appropriate hardware (see below), 2- and 3-d ICD representations are equally well suited for use in either environment.

Summary: Given appropriate hardware, the relative utility of 2- and 3-d ICD representations is not likely to be affected by the physical environment.

8.4.3 Computing Environment

8.4.3.1 Hardware

A variety of hardware platforms are currently used for knowledge engineering activities. Some projects are based on mainframe computers and rely on basic vt100 terminals (eg S1:1b:350, S6a:1a:420). Some are based on PCs (eg S4:1a:240, S6c:1a:560) but the majority are apparently based on general purpose workstations (eg S2:1a:503, S3:1b:320, S5:1a:299, S6b:1a:470, S6d:2a:169).

The use of special purpose workstations was raised by subject 2 (S2:1a:491). As he pointed out, the more specialised or non-standard the development environment, the more work has to be done in porting a system to a suitable platform for delivery. This kind of problem had also been recognised by subject 6b who had found that tools for knowledge engineering which ran only on a specialised workstation were less accessible to the majority of knowledge engineers (who simply didn't know how to use the machine) and therefore less well used (S6b:1a:310).

The question of using special purpose hardware to support representation of 3-d images was explicitly discussed with all subjects. Few subjects liked the idea of having to wear helmets or head-tracking devices, or even special spectacles. Subject 6d thought that wearing something on his head would be restrictive (S6d:2a:450) and subject 4 said that it would simply not be feasible in the environment in which he worked which was geared towards integrating knowledge-based system technology with mainstream data processing systems (S4:1b:57). The suggestion of wearing such headgear provoked laughter in the

interview with subjects 6a - c (eg S6a: *'I wouldn't like to wear those glasses'*), although it was acknowledged that a less cumbersome form of apparatus would be more acceptable (S6b:1b:280). Subject 5 thought that the initial embarrassment of wearing an unusual gadget on his head would pass and pointed out that people do in any case spend a lot of time sitting in front of their terminals and do wear personal stereos at present. Even so, he thought it would be preferable if 3-d images could be viewed *'without such gimmicks'* (S5:1b:204).

One subject made the comment that with 2-d node and link representations such as those employed by a number of knowledge engineering tools, a PC screen easily became too cluttered to be useful (S6d:2a:325). The use of 3-d representations could ameliorate the situation somewhat if, as argued by Robertson *et al.* [116], more information can be displayed on a given screen using 3-d representations than is possible with 2-d.

Summary: 3-d ICD interfaces for knowledge engineering are most likely to be useful if they can be implemented on general purpose workstations or PCs and do not require the use of specialised hardware of any kind. They may be more useful that 2-d node and link representations in reducing the problem of screen clutter, especially for users working with small screens.

8.4.3.2 Software

Subjects had experience of using a wide variety of languages and tools for knowledge-based system development. Languages such as Lisp and Prolog had most commonly been used (eg S1:1b:250, S2:1a:505, S3:1a:350, S5:1a:515, S5:1a:550, S5:1a:588, S6c:1a:495, S6d:2a:169), also C (S2:1a:466, S3:1a:350). PC-based expert system shells had been encountered by many subjects but less often used (eg S4:1a:560, S4:1a:570, S5:1a:515). (One subject reported finding shells too restrictive (S2:1a:474)). Similarly, most subjects were aware of the existence of more powerful workstation tools for knowledge engineering, but only two had actually used any (S5:1a:458, S5:1a:568, S6b:1a:515).

Summary: It is proposed that 3-d ICD representations would be most likely to be useful if they were linked to Lisp or Prolog-based knowledge-based system development environments.

8.5 System Characteristics

This section will consider the relative utility of 2- and 3-d representations in developing knowledge-based systems of various kinds. Relevant aspects of the system are thought to include:

- the style of knowledge representation used:
  - rules
  - frames and objects
- the size of the knowledge structures to be represented
- the complexity of knowledge structures to be represented:
  - the number of different types of relationship to be represented

Environmental Factors	
Organisational environment	
Management communications	There is no evidence to suggest that 3-d ICDs would be useful in communicating with project managers.
Project communications	There is no evidence to suggest that 3-d ICDs would be useful in communicating with other members of a project team.
Physical environment	Given appropriate hardware, the relative utility of 2- and 3-d ICD representations is not likely to be affected by the physical environment.
Computing environment	
Hardware	3-d ICD interfaces for knowledge engineering are most likely to be useful if they can be implemented on general purpose workstations or PCs and do not require the use of specialised hardware of any kind. They may be more useful that 2-d node and link representations in reducing the problem of screen clutter, especially for users working with small screens.
Software	It is proposed that 3-d ICD representations would be most likely to be useful if they were linked to Lisp or Prolog-based knowledge-based system development environments.

Table 8.3: Effects of Environmental Factors on Utility of 3-d ICDs for Knowledge Engineering

- the average number of relationships in which individual knowledge-based system entities are involved

The initial reaction of some subjects to the possibility of a 3-d graphical representation was that it would be suitable for larger and more complex systems than would be easily manageable in 2-d (S4:1b:40, S6d:3b:684). The benefits here are, however, probably not as great as was expected.

8.5.1 Knowledge Representation Style

8.5.1.1 Rules

Subjects were often not very enthusiastic about the possibility of developing 3-dimensional representations of fragments of rule-based systems. It is interesting to note that while subjects 2, 3, 5 and 6 had all been involved in projects using rule-based representations, none suggested these as suitable candidates for representation by 3-d ICDs in preference to other available object or frame-based systems. One subject stated *'I only see graphical interfaces being useful for building object systems'* (S6d:2a:380).

Two of the case studies (numbers 1 and 4) did involve representing fragments of rule-based systems. As described above, the representation developed in study 1 was judged by the subject not to be very successful. (It should, however, be noted that study 1 was the pilot study and that depth information was not meaningfully used in the scheme developed.) The representation developed in study 4 was more successful. Subject 4 was reasonably happy with the representation as a whole (*'I think it works'* (S4:3a:700)) though he did have some detailed suggestions regarding possible improvements, and there were problems with the display of the English-like content of rule conditions and consequents.

It is difficult to generalise on the basis of these findings regarding the utility of 3-d ICDs in the representation of rule-based systems. A tool supporting the use of 3-d ICDs which would allow nodes representing components of a rule network to overlap might be useful in permitting the display of larger amounts of information than would be possible using 2-d alone, even if depth information is not used in a meaningful way. The possibility that 3-d ICDs might be useful in representing rule-based systems should not be ruled out.

Summary: There is limited support for the claim that 3-d ICDs might be more useful than 2-d in the representation of components of rule-based systems.

8.5.1.2 Frames and Objects

Studies 2, 3, 5 and 6 involved representing fragments of frame-based systems. In studies 2, 5 and 6, one kind of relation between frames or objects was shown using spatial coding in the $x - y$ plane and another in depth. This kind of arrangement or layout was in each case suggested by the subject rather than the evaluator, and was judged by subjects to be very successful: *eg* 'the actual diagram itself is helpful .... showing containment in depth works well' (S2:3c:527).

In one case (study 3), spatial coding was used in the representation of three separate relationships, one being shown in y (between objects mainly at the front of the network) one in z, and a third in x between objects at the back. This arrangement was judged by

the subject to be quite successful: *'the use of the extra dimension is helpful'* (S3:3d:480) but was found to be rather complex *'it still looks like a tangle at the bottom'* (S3:3d:470).

Other suggestions regarding the possible use of 3-d ICDs in the representation of frame-based systems were as follows. Subject 3 suggested that structures similar to the above could be used to represent disease taxonomies which could usefully be classified in terms of either causes, signs and symptoms or treatments, orthogonally to the usual 'kinds of' hierarchy (S3:3d:500). Subject 4 suggested the possibility of showing the involvement of individual objects in physical, functional and pricing structures within the same diagram (S4:1b:330). More generally, subject 3 suggested that the use of 3 dimensions should make the clustering of nodes in space according to the semantic relatedness of the objects or concepts they represented easier than was the case with 2-d representations (S3:1a:690). Finally, subject 2 suggested that it might be useful to position nodes representing objects with physical counterparts in such a way as to reflect the locations of the real physical objects (S2:2a:97).

It seems likely that an important reason for the success of the representations of frame or object-based systems used in studies 2, 3, 5 and 6 was the fact that it was possible in each case to use depth information in a meaningful way. Though all four subjects were satisfied with the use of depth in their own designs, subjects 5 and 6d emphasised the fact that depth information would be less useful in cases where it was not used in the representation of any particular relationship (S5:3c:35) since it would in this case be impossible to recognise or use any *'definite scheme in depth'* (S6d:3b:700). Indeed, subject 6d found that it was hard to see the training diagram (in which no obvious organisation in depth was used) as 3-dimensional at all, due to the lack of either a visual or a logical or conceptual structure to support this perception (S6d:3b:705).

On the basis of these case studies, it seems reasonable to suggest that it would often be possible to represent object or frame-based systems using 3-d ICDs in such a way that the meaningful use of depth information would be helpful to the knowledge engineer. As subject 3 pointed out, this use of depth information to represent the position of objects or frames in rich semantic structures is likely to assume even greater significance with the development of second generation expert systems using more 'deep' or semantically rich knowledge (S3:1b:100).

Summary: 3-d ICDs are more useful than 2-d for representing object or frame-based structures in which two or three different kinds of relationship between knowledge structure components are to be viewed simultaneously.

8.5.2 System Size

The initial reaction of some subjects to the possibility of a 3-d graphical representation was that it would be suitable for larger and more complex systems than would be easily manageable in 2-d (S4:1b:40, S6d:3b:684). The benefits of 3-d are, however, probably not as great as expected in this respect.

The provision of three dimensions of freedom in graphical representation rather than two is not, on its own, enough to solve the problems of size or volume of information encountered in the case of large knowledge-based systems. Knowledge-based systems can be extremely large (even the system considered in study 3 which was not yet fully developed consisted of around 60,000 Prolog clauses), and systems are set to become

larger in future (S3:1b:45). More information can indeed be seen on a screen that acts as a window on space (*ie* onto a 3-dimensional representation) than one which acts as a frame on a map (or 2-d representation). This has been demonstrated analytically by Robertson *et al.* [116]. The solution to the problem of representing large amounts of information in a manageable way is, however, felt by the author to lie in the provision of increased functionality (to support abstraction and encapsulation) in tools for the manipulation of graphical representations, both two- and three-dimensional.

Summary: 3-d diagrams permit the representation of more information than 2-d, but this is not, in itself, enough to solve the problems caused by the volume and complexity of information used by knowledge-based systems.

8.5.3 System Complexity

8.5.3.1 Number of Different Relations

As illustrated in section 8.5.1.2, the availability of a third dimension means that it is possible to display at least one more relation or partial ordering on entities in a system than would be possible with 2-d alone. In some cases, it would be possible to display larger numbers of relationships in a 3-d diagram by using spatial coding if sets of objects in different relations could, for example, be shown in different $x - y$ laminas, all linked by the same relation in depth. (The representation developed in study 6 provides an example of the way in which this could be done.)

A single knowledge-based system often embodies many different relations between the entities with which it is concerned. For example, the system considered in study 1 used 14 different relations, and that used in study 3 used 30. The attempt made in study 1 to represent 12 of the 14 relations used by means of visually coding the relevant links was, perhaps not surprisingly, rather unsuccessful: the diagram was found to be complex and unaesthetic (S1:3b:600), and the user was forced constantly to refer to the key for the meanings of links of various types. The system considered in study 3 used around 30 different relations but in order to promote visual and conceptual clarity, it was decided that only 4 of these relations should in this case be represented. This scheme used depth in a more meaningful way than that developed in study 1 and was indeed more successful. Part of the reason for its relative success was almost certainly the reduction in the total number of relations shown.

Even in 3-d diagrams in which a number of relations are depicted using spatial coding, the representation of all relevant relations between entities in a knowledge-based system within the same diagram can still be problematic (S3:3d:470). Although 3-d diagrams do permit more relationships to be depicted using spatial coding than is possible using 2-d alone, they will often still not be enough to cope with the degree of orthogonality in a typical knowledge-based system (S5:1a:115).

Summary: The use of 3 dimensions facilitates the representation of larger numbers of orthogonal relations in a single diagram than is possible with only 2. But this extra facility will not, in itself, be enough to cope with the degree of orthogonality in a typical knowledge-based system.

System Characteristics	
Knowledge Representation Style Rules Frames and Objects	There is limited support for the claim that 3-d ICDs might be more useful than 2-d in the representation of components of rule-based systems. 3-d ICDs are more useful than 2-d for representing object or frame-based structures in which two or three different kinds of relationship between knowledge structure components are to be viewed simultaneously.
System size	3-d diagrams permit the representation of more information than 2-d, but this is not, in itself, enough to solve the problems caused by the volume and complexity of information used by knowledge-based systems.
System Complexity Number of different relations Number of relationships per entity	The use of 3 dimensions facilitates the representation of larger numbers of orthogonal relations in a single diagram than is possible with only 2. But this extra facility will not, in itself, be enough to cope with the degree of orthogonality in a typical knowledge-based system. There is no evidence to suggest that 3-d diagrams are more useful than 2-d for representing knowledge structures in which individual entities are involved in many different relationships.

Table 8.4: Utility of 3-d ICDs in the Representation of Systems of Different Kinds

8.5.3.2 Number of Relationships Per Entity

These case studies provide no evidence that 3-d node and link diagrams are any better than 2-d versions for simultaneously representing the involvement of single frames or objects in many different relationships. One subject suggested that the maximum number of relationships which can comfortably be represented is probably between 4 and 6 in either case (S3:3d:490). Although rocking motion can be used to *'separate the clutter'* (S1:3b:650) and relieve the impression that everything is connected with everything else, a 3-d diagram can still *'look like a tangle'* (S3:3d:470) if complex structures are represented in an unprincipled manner with layout not conforming to any recognisable scheme.

Summary: There is no evidence to suggest that 3-d diagrams are more useful than 2-d for representing knowledge structures in which individual entities are involved in many different relationships.

8.6 Recommendations for Further Development

The knowledge engineers involved in the case studies suggested that 3-d ICDs are likely to be more useful than 2-d in representing object or frame-based systems involving more than one relation of interest, particularly at the stages of design, debugging and maintenance. It is therefore concluded that further development of tools such as ICDEDIT for supporting the use of 3-d ICDs would be worthwhile and that tools aimed specifically at supporting the use of such diagrams in knowledge engineering would also be useful.

In order to be most useful, it is suggested that 3-d ICD support for knowledge engineering should be aimed specifically at design, debugging and maintenance activities and might initially be developed for use on general-purpose workstation technology in conjunction with a Lisp or Prolog-based development environment. Facilities for complexity management (such as functions for generating views of subsets of a system's relations) should also be provided.

8.7 Summary

This chapter has discussed aspects of users, tasks, environments and knowledge-based systems identified as relevant to the utility of 3-d ICDs as an interface technology for use in tools for knowledge engineering. Discussion was based on observations made during the case studies described in chapter 7 and on comments made by knowledge engineers during stages 1 and 3 those studies *ie* both before and after having used ICDEDIT. Recommendations for the further development of tools to support the use of 3-d ICDs in knowledge engineering have been made on the basis of these observations and comments.

Chapter 9

Usability of Depth-Related Features of ICDEDIT

9.1 Introduction

This chapter presents some of the results of the case studies described in chapter 7. Results presented here are those concerning the usability of features of ICDEDIT which permit 3-d ICDs to be viewed and edited. On the basis of these results, a number of recommendations are made regarding the way in which such features could be implemented in future versions of the tool.

Usability is assessed on the basis of observations of subjects' performance in the task sessions described in chapter 7 as well as the verbal protocols recorded during task performance and the interview protocols recorded at the end of the session. Data was analysed in a qualitative manner. It was felt that a quantitative analysis of, for example, numbers of errors or times taken by different subjects to perform similar tasks, would not be appropriate in this case: as described in chapter 7, different subjects worked with different representations and tasks were slightly different in each case so comparison of data from different studies would not have been valid. The number of independent variables which might potentially be considered in a quantitative analysis is also huge. In any case, it was felt that an open-ended qualitative analysis would, at this early stage, yield more valuable insights into the need for changes in design to improve usability than detailed quantitative analyses. (Further arguments against the use of controlled experiments to guide the design of human-computer systems are presented in chapter 10.)

Usability is here assessed in terms of effectiveness and satisfaction. Efficiency (another factor frequently assessed in studies of usability) is not considered as there was no suitable benchmark against which to compare performance observed in these studies. Usability of features of ICDEDIT not involved with the use of depth information is also not considered, as it is assumed that these kinds of features may be implemented in a similar manner in tools supporting 2-d and 3-d diagrams. Finally, it should, once again, be noted that all except one of the subjects in the case studies were new to ICDEDIT at the time of the case studies and that recommendations made in this chapter are therefore relevant mainly to the design of features of a future tool intended for novice users.

In the following sections, comments based on statements made by knowledge engineers

during case study interviews are indexed with a reference to the point in an interview recording where the statement can be found. References take the form Ss:t:u where s identifies the subject, t identifies the relevant tape number and side for that study, and u identifies the position of the statement within that side. Note that in the sixth case study, a preliminary interview was conducted with a group of three knowledge engineers from the target organisation before carrying out the full scale study with a fourth subject from the same organisation. In this case, different subjects are further identified by the use of letters and are referred to as S6a, S6b, S6c and S6d.

9.2 Effectiveness of Support for Perceiving Depth

This section considers the effectiveness of hiding, perspective, rocking motion and colour as cues to depth in 3-d ICDs. The use of hiding, perspective and rocking motion as cues to depth was directly supported in ICDEDIT as described in chapter 4. The use of colour was also supported.

Studies 2, 3, 4, 5 and 6 all investigated the use of representations in which meaningful information was encoded in terms of relative positioning in depth (*ie* representations which made meaningful use of depth information). Three of these representations (those in studies 4, 5 and 6) used colour, as well as hiding, perspective and rocking motion as a further 'redundant' cue to depth. Of the subjects involved in these studies, all except one (subject 2) were apparently able to perceive representations used as three dimensional without difficulty. Subject 5 reported beginning to *'think in 3-d'* (eg S5:3b:635) by the end of his task session. The problems of subject 2 were thought to be caused partly by problems with perceiving the representation as three-dimensional (see below) and partly by problems with the facilities for manipulation (see section 9.3).

The effectiveness of depth cues, or in other words, their ability to support correct perception of information conveyed in terms of depth is, of course, a pre-requisite for effective interpretation of 3-d ICD representations which use such information in a meaningful way. The ease with which subjects were able to perceive this information in representations used in the case studies was evaluated in tasks in which they were asked to locate components of a diagram in terms of their positions. Subjects were asked to do this both in terms of features of the graphical representation (*eg to identify a node in front of the one marked 'p2'*) and in terms of the knowledge structure represented (*eg to identify an element which contains the element 'Traffic route'*). The effects of the depth cues on perception of other information encoded in the representation (in terms of node size or relative positioning on x or y axes) was also observed in other tasks.

9.2.1 Hiding

Hiding was apparently a strong cue to relative depth. For example, subject 2 identified nodes at the front of the network as those that he *'could see all of'* (S2:3b:35), and subject 4 thought that it would be much harder to perceive a diagram as 3-dimensional in the absence of such a cue (S4:3b:10).

9.2.2 Perspective

Five out of the six subjects used perspective projections in at least some of the tasks they performed. The use of perspective was found to cause two kinds of confusion.

The first confusion concerned differences in the relative sizes of nodes in the diagrams. Three of the six subjects (subjects 2, 4 and 6) did not realise that nodes which appeared to be different sizes under a perspective projection had actually been defined to be the same size.

The second concerned the relative positioning or alignment of nodes. One subject (subject 2) was confused by the fact that, under a perspective projection, apparent visual alignment of two nodes relative to the x or y axes may not be taken as evidence that they share the same x or y co-ordinates. Subject 5, on the other hand, stated that he was overcoming the possibility of such confusion by consciously accounting for it in making judgements about relative positioning.

It was noted that no subjects took advantage of the facility for altering the amount of perspective imposed on projections. However, one subject (subject 3) turned off the perspective completely at the beginning of the task session and worked all the way through without it. He explained that he found the extra information provided by a perspective projection (as opposed to a simple planar projection) too much to cope with (S3:3d:360).

9.2.3 Rocking Motion

Rocking motion proved to be a valuable cue to depth and was used in at least some of the tasks by all subjects. Two subjects (3 and 6d) used motion throughout the task session. For subject 6d, the diagram *'didn't look 3-d ... until you had the rocking motion going'* (S6d:3b:605).

Two subjects explained how they used the amount a node moved to judge its position relative to another: subject 2 identified the front node as *'the one that shakes most'* (S2:3b:60), and subject 3 explained how he had realised that, since the motion of a particular node was proportional to its z co-ordinate, two nodes which moved the same amount (and in the same direction) were at the same depth.

Other subjects (4 and 5), used rocking motion as a cue to local structure and saw it as a kind of 'local rotation' which allowed them momentarily to see the structure behind something which was in the way when the diagram was static (eg S5:3b:670).

The effectiveness of rocking motion as a cue to relative depth was found to be greatest in the case of diagrams whose spatial organisation was somewhat unprincipled. For example, subject 4 suggested that it seemed more useful with the training diagram (in which nodes were positioned at random) than with his own representation in which nodes were positioned according to a recognisable scheme for layout (S4:3b:19). In structured representations, rocking motion was only really useful in separating nodes which appeared to be close together in depth (S2:3c:605, S4:3b:25, S5:3b:200). A possible explanation for this is as follows. In representations (such as that used in training) whose layout is unprincipled, other cues to relative depth may be ambiguous or lacking. In the training diagram, for example, there was little hiding since nodes were quite sparsely distributed; differences in apparent size may have given misleading cues to relative depth as they resulted both from the use of a perspective projection, and from differences in actual size;

and differences in colour did not correspond to differences in depth. Any recognisable scheme for layout is itself likely to provide the user with strong cues to relative depth. This might be the case if, for example, all nodes are positioned so that they fall into one of a small number of $x - y$ planes or into a hierarchy in an $x - z$ or $y - z$ plane. Cues to depth provided by the use of rocking motion are likely to assume greater importance when other cues - such as those provided by hiding, perspective, colour or the use of principled layouts - are absent.

Finally, three subjects commented on the type of motion available. Subject 6d would have preferred it if the movement were smoother (S6d:3c:50). Subjects 4 and 5 both found the ability to vary the type of motion useful. Subject 4 commented that with a structured representation, the use of the wrong kind of motion could actually break the illusion of a rigid network in motion and cause individual nodes to appear to *'bob up and down'* independently of each other (S4:3b:19). Subject 5 suggested that facilities for varying the kind of motion used should be made more directly accessible by providing the user with a slider to control amplitude and a dial to control angle of rocking motion axis.

9.2.4 Colour

Colour was found to be very useful as an additional cue to depth. For subject 4, it was a major cue: he predicted that if such a cue were not present, the user could be *'in big trouble'* (S4:3b:10). Subject 6d also found that the meaningful use of colour (*ie* the use of a colour scheme which meant that differences in colour corresponded to differences in depth) made it significantly easier to perceive differences in depth (S6d:3b:640).

9.2.5 Summary and Recommendations

The depth cues supported in ICDEDIT were successful in conveying information about relative depths of nodes within a 3-d ICD to all the relevant subjects except one. Even subject 2, who claimed not to have a proper 3-d perception of the representation used in his case study, was in fact able to successfully identify nodes described only in terms of their position in depth relative to other nodes.

A number of recommendations for the implementation of depth cues in future tools for viewing and editing 3-d ICDs may be made on the basis of the findings reported above. These may be summarised as follows:

1. support hiding
2. support the use of perspective:
    - also provide for non-perspective views (*ie* allow users to 'switch off' perspective)
3. support the use of rocking motion:
    - attempt to make rocking motion smoother
    - maintain facilities for adjustment of rocking motion type:
        - allow rocking motion to be 'switched off'

- make facilities more directly accessible, through, for example, the use of sliders and dials

4. support the use of colour in redundant depth coding

9.3 Effectiveness of Support for Manipulating 3-d ICDs

This section concerns the support provided in ICDEDIT for tasks in which users were required to manipulate the representations used in the case studies in order to search for particular nodes or links. A brief description of facilities for manipulating 3-d ICDs in ICDEDIT is given in appendix C together with a number of snapshots of ICDEDIT control panels which illustrate the way in which these facilities were made available to the user.

In order that this kind of search can be performed as reliably as possible, it is important that the necessary manipulations of the structure should be of minimal cost to the user. If the cost of the search is too great, users will be inclined either not to search at all, or to search less thoroughly than they otherwise might. Use of the manipulation functions supported by ICDEDIT apparently carried quite a high cost, as some subjects preferred to guess or infer the existence of certain hidden diagram components rather than manipulate the diagram in order to check that they were really there (eg S4:3a:95, S6d:3a:680). Indeed one subject found the constant need to adjust his view of the diagram very stressful after a time as the system continually failed to respond in the way he expected (S2:3b:380).

The ease with which subjects were able to manipulate 3-d ICDs using functions provided by ICDEDIT was evaluated in tasks in which they were asked to search for particular nodes and links (or sometimes groups of nodes or links) within the 3-d representations used. These nodes and links were either visible, partially visible or hidden at the time subjects were asked to begin searching. In the case of visible components, the task was easy. The only need for depth information was in making judgements about the depth of the target component relative to other nodes in order to ensure that the one picked satisfied the experimenter's description. Support for making judgements about relative depth was discussed in the previous section. In cases in which the target (or at least some members of the target group) began by being partially or completely hidden, the task was more difficult as it required the user to manipulate the diagram as a whole in order to reveal the parts that had been hidden in the initial view due to occlusion by other nodes. The rest of this section highlights the strategies commonly used to manipulate diagrams in 3-d, discusses problems encountered in using these strategies, and makes some recommendations for improvements in the usability of relevant functions in ICDEDIT.

9.3.1 Strategies

The strategy most commonly used by subjects attempting to reveal partially visible nodes or links or searching for hidden ones was to rotate the diagram as a whole about either the x or the y axis. Some subjects (*eg* subject 4) used mainly 90 degree rotations in order to give an orthogonal projection of the diagram, others (*eg* subject 6d) used 30 - 45 degrees of rotation in order to allow them to see over or round occluding elements. The only other strategy observed was a variation on this in which rocking motion was used

as a form of 'local rotation' which, at a slow enough speed and large enough amplitude allowed subjects to get an impression of the structure behind the foremost nodes.

It is worth noting at this stage that certain facilities for diagram manipulation were not commonly used. *Z* shift and scaling operations were rarely used and were also confused by one subject (S5:3a:230). Incremental rotations were never seen to be used: subjects apparently preferred to keep initial rotations simple and then calculate any adjustments required in terms of the original operations.

9.3.2 Problems

A number of problems were encountered in using the strategies described above. The use of rocking motion has already been discussed (see section 9.2.3). This section will therefore concentrate on difficulties with the use of rotation.

Subjects encountered several problems when attempting to rotate diagrams in ICDEDIT. They were disorientated by the fact that rotations were performed relative to the abstract co-ordinate system, rather than the diagram itself: subjects 2, 3 and 6 expected rotations to be done about an axis through the centre of the diagram (rather than an axis through the point (0,0,0) as was the default in ICDEDIT) and were bewildered when it transpired that this was not how it was done. As subject 2 explained, *'Its very difficult because it doesn't rotate round the centre of the diagram - I find that confusing'* (S2:3b:90).

Disorientation was further increased by the fact that rotations could only be achieved indirectly by setting numerical parameters (*eg* S2:3c:390, S4:3b:140). All subjects who needed to manipulate a diagram in depth to any extent (*ie* all subjects except subject 1), stated that they were frustrated by the need to think of doing transformations in terms of numbers, rather than by direct manipulation. Typical comments were: *'I don't want to have to deal with numbers'* (S2:3c:400), *'I think these numbers make life very difficult'* (S5:3a:647). Subject 5 suggested that other operations such as shifting or translating the diagram within the window should also be under more direct mouse control, perhaps via scroll bars (S5:3a:647).

Finally, the fact that transformations affected by ICDEDIT happened suddenly, all in one go, rather than gradually and continuously was also found to be problematic. Subjects 2, 4 and 6 all suggested that they would have preferred transformations to happen more smoothly and gradually. Subject 4 suggested that it should be possible to rotate the diagram in a *'gradual and controlled way'* using direct manipulation as in packages for CAD (S4:3b:140).

Problems such as these meant that subjects often took a very long time to achieve the transformations they intended. Subject 4 took 4 minutes and 20 secs to obtain the view he needed for one task, while subject 2 took 7 minutes attempting to achieve a particular view and at the end of that time gave up, having still not been successful.

9.3.3 Summary and Recommendations

Subjects apparently found it quite difficult to manipulate 3-d ICDs using the facilities provided by ICDEDIT. The most common strategy for manipulating a diagram in order to search for hidden or partially hidden targets involved the use of rotation. A slow rocking motion was also sometimes used to enable subjects to 'see around' occluding elements

of a diagram. A number of problems with facilities in ICDEDIT for the manipulation of whole diagrams were encountered.

Recommendations for the implementation of diagram manipulation facilities to support search operations in future tools for viewing and editing 3-d ICDs may be summarised as follows:

1. support rotation about the center of a diagram
2. support direct manipulation of the diagram as a whole:
   - support direct control of rotation, possibly by dragging
   - support direct control of translation, possibly using scroll bars
3. support gradual smooth rotations
4. omit either the Z shift or the scaling operation (subject to confirmation of the above findings in further testing with expert users)
5. omit incremental rotations (subject to confirmation of the above findings in further testing with expert users)

9.4 Effectiveness of Support for Positioning Elements in 3-d ICDs

This section will concentrate on the support provided in ICDEDIT for tasks in which the user was asked to position or relocate individual elements in a diagram. Once again, the reader may refer to appendix C for a brief description of facilities provided by ICDEDIT as well as snapshots of the ICDEDIT control panels.

If 3-d ICDs are to be used to support knowledge engineers in building or editing knowledge-based systems, it is important that the positioning of nodes within them should be quickly and easily achievable. Accuracy of placement may also be important if spatial information in the graphical representation is used in a meaningful way.

The effectiveness of support provided by ICDEDIT for positioning nodes within a 3-d ICD was evaluated in tasks which required subjects to place elements within a 3-d ICD, either at some position specified only in terms of its co-ordinates, or in a position specified relative to that of another node. The rest of this section describes strategies commonly used in the placement of elements in a 3-d ICD and makes a number of recommendations on this basis.

9.4.1 Strategies

Nodes were normally placed in absolute terms simply by entering the appropriate co-ordinates in numerical form. The need for the use of co-ordinates in placing diagrams components accurately in this way was acknowledged (S3:3e:90, S5:3b:572).

For placing a node relative to another node, the strategy was typically more complex. A common strategy was to begin by dragging the new node (using the mouse) to an apparently reasonable position in the $x - y$ plane. If the new node was required to be either in front of or behind an existing node, then any further positioning in depth was

usually achieved using the modified dragging operation (*ie* mouse and shift key) available for dragging in depth. However, if the new node was required to be at exactly the same depth as another existing node, subjects first determined the z co-ordinate of the existing node and then entered the appropriate co-ordinate for the new node in numerical form, without bothering to use the operation of dragging in depth.

No particular problems were encountered with the placement of individual nodes. However, subjects often preferred to use direct manipulation rather than working in terms of numbers and it was thought that numerical co-ordinate information should not feel central to the system (S3:3e:90). One subject suggested the provision of an 'alignment' facility, which would allow users to specify, maybe directly, an existing node with which a new node could be matched or aligned in x, y or z. This would lessen the user's reliance on numerical co-ordinates.

9.4.2 Summary and Recommendations

Few problems were experienced in positioning individual nodes within a 3-d ICD. Accurate positioning was achieved by entering numerical co-ordinates. Approximate positioning relative to other nodes was achieved using the dragging operations provided in ICDEDIT.

Recommendations for the implementation of node placement facilities in future tools for viewing and editing 3-d ICDs may be summarised as follows:

1. maintain facilities for positioning nodes by entering numerical co-ordinates
2. maintain facilities for positioning nodes by dragging (both in the $x-y$ and $x-z$ planes)
3. provide an alignment facility

9.5 User Satisfaction

The subjective responses of users to a system are indicators of what is sometimes called its 'acceptability' and should play an important part in considerations of overall usability.

The degree of satisfaction felt by subjects in the case studies was evaluated on the basis of information gathered in the interviews carried out at the end of the task session. Subjects' reactions to ICDEDIT and its representation of depth information varied.

Subjects 3, 4 and 6d were favourably impressed by the system as a whole. Subjects 4 and 6d found that the diagrams looked more 3-dimensional than they had expected on the basis of having seen only static black and white versions of the kinds of diagrams used (S4:3a:613, S6d:3b:700), and subject 3 found that the use of depth as part of the representational paradigm worked out better than he had anticipated it would (S3:3d:480). Subject 5 initially found dealing with the existence of a third dimension unexpectedly difficult (S5:3a:635), but by the end of the session *'felt quite comfortable working in 3-d'* (S5:3c:30). Subject 3 underwent a similar transition from being *'hopelessly lost'* at the beginning of the session (S3:3d:330) to *'quite enjoying it towards the end'* (S3:3d:405) and said that he *'would certainly play with it'* again in future if given the opportunity to do so (S3:3e:250).

The evaluation of user satisfaction paid particular attention to subjects' reactions to the use of rocking motion, since this was a relatively novel feature of ICDEDIT. Reactions to the use of rocking motion were almost universally favourable. Subject 2 thought this was the best thing about the system (S2:3c:376) and subject 4, while having initially thought that *'looking at that for a while would make you feel very sick'* (S4:3a:162), was quite enthusiastic by the end of the session, saying that the motion *'looked very good'* under certain circumstances (S4:3b:30).

Some subjects' reactions to the system as a whole were, however, less favourable. Subject 2 found the experience *'horrendous'* (S2:3c:370) and subject 6d found it *'frustrating'* (S6d:3d:590). These reactions were certainly due in part to the prototypical nature of the tool. Subject 6d cited the slowness of system response as the main cause of his frustration (S6d:3d:595) and subject 2 suffered two system crashes during the course of his task session. But at least some of subject 2's difficulties were due to his inability to derive an overall impression of a 3-d structure from the cues provided as described in section 9.3.

9.5.1 Summary and Recommendations

Since reactions to the system as a whole, and in particular to its management of depth information were not always positive, it seems clear that a number of modifications of the kinds suggested above should be made in order to enhance the acceptability and overall usability of future versions of ICDEDIT.

Some aspects of the system, for example its use of rocking motion as a cue to depth, were, however, favourably evaluated, and it is anticipated that a new implementation incorporating the best aspects of the old as well as some new facilities would achieve more completely positive ratings.

9.6 Summary

This chapter has considered the usability of features of ICDEDIT which permit 3-d ICDs to be viewed and edited. The discussion has been based on observations of knowledge engineers performing simple atomic tasks involving the use of depth information in a laboratory environment as described in chapter 7. The strategies commonly used in performing these tasks were recorded and considered together with problems frequently encountered and comments made by knowledge engineers either during task performance or in the subsequent interview. Subjective responses of individual users to the portrayal of depth information in the tool used are also reported. On the basis of these considerations, a number of recommendations regarding future implementations of tools to support the use of 3-d ICDs have been made.

Part IV

Conclusion

The final section of this thesis concludes by laying the foundations for further work on the use of 3-d ICDs in the human-computer interface of tools for knowledge engineering. Chapter 10 makes a number of suggestions regarding the ways in which issues raised in the consideration of the utility of 3-d ICDs might be further investigated using a series of controlled laboratory experiments. Chapter 11 lays the foundations for producing a formal specification of the interface of a tool for knowledge engineering which would use 3-d ICD representations of knowledge structures of interest. Finally, chapter 12 evaluates the success of work described in earlier chapters and makes some specific recommendations for the further development of tools supporting the use of 3-d ICDs in knowledge engineering.

Chapter 10

Further Investigation of the Utility of 3-d ICDs for Knowledge Structure Representation

10.1 Introduction

Information relating to the utility of 3-d ICDs as an interface technology in tools for knowledge engineering was collected during the case studies described in chapter 7. This information was used in the discussion of the utility of 3-d ICDs presented in chapter 8. Recommendations for the development of tools such as ICDEDIT were made on the basis of this discussion.

It has been suggested that information of the kind collected during the case studies provides only weak justification for the recommendations for further development (and the claims about the relative utility of 2- and 3-d ICD interfaces which are implicit in such recommendations) presented in chapter 8. This chapter describes ways in which further empirical data concerning the relative utility of 2- and 3-d ICD interfaces in knowledge engineering could be collected. It is intended that findings from studies of the kind proposed here should provide a basis on which to make more definite recommendations regarding the further development of a tool such as ICDEDIT for use in knowledge engineering.

Section 10.2 contains an outline for a conventional controlled experiment which could be used to investigate the degree of support provided by 3-d, 2-d and textual representations of knowledge structures for knowledge structure comprehension. The hypothesis to be investigated in that experiment is that 3-d representations provide better support for such a task than either 2-d or textual representations and that further development of tools supporting the use of 3-d representations in knowledge engineering is therefore justified.

If a decision were taken to further develop support for the use of 3-d ICDs in knowledge engineering, there would then be a need for further information to determine exactly what kind of support might most usefully be provided. The findings of the case studies indicated that 3-d ICDs might be more useful to certain users and in certain situations or

contexts rather than others, and development managers might therefore wish initially to direct development work at providing 3-d ICDs only in those situations where the benefit is thought likely to be greatest.

It would, in principle, be possible to investigate the relative utility of 2- and 3-d representations in many of the situations of interest using further controlled experiments. Various aspects of a situation could be controlled: values of attributes along certain dimensions (relating to the user, the task or the situation) could be held constant while those on others could be varied in a controlled manner. Designs for such experiments are presented in appendix E. However, it is argued here that the goal of determining what direction further development of a tool such as ICDEDIT should take would not be well served by the use of factorial controlled experiments. The problems associated with using such experiments in this context are discussed in section 10.3, and an alternative approach to gathering empirical data of the kind needed is proposed.

10.2 Experimental Comparison of 3-d, 2-d and Textual Representations of Knowledge Structures

This section describes an experiment which could be used to investigate differences in the support provided by 3-d, 2-d and textual representations of knowledge structures in tasks involving comprehension of complex knowledge structures.

The method was proposed after consideration of various studies aimed at investigating the use of graphical displays to support a number of different activities. Approaches considered include the method used by Wright and Reid in investigating the impact of information presentation on problem-solving performance [158]; that used by Brooke and Duncan in evaluating the effect of system display format on performance in a fault location task [18]; and those used in evaluating the effectiveness of various methods of graphical presentation of quantitative data [26, 27, 4, 133, 24].

The aims of some of the work on graphical data presentation seemed ostensibly to be similar to that proposed here, in that two- and three-dimensional presentations were compared for effectiveness in supporting a number of tasks. In fact, however, the similarity was not that great. Such work has concentrated on the investigation of techniques for the display of numerical data such as bar charts, pie charts and line graphs and its relevance to the investigation of node and link diagrams is therefore limited. It has focused on the presentation of quantitative information and has (with the exception of [4]) looked at the use of the third dimension as a purely presentational device to which no meaning is attached (see, for example, [24]). In contrast, the aim here is to investigate the presentation of qualitative information and the meaningful use of the third dimension.

The proposed approach is based most closely on Wright and Reid's [158] investigation of the effects of using a variety of styles (including text, tables and 'algorithms' or flow charts — in other words two-dimensional node and link diagrams) for the presentation of qualitative information on performance in problem-solving tasks.

10.2.0.1 Hypotheses

It is hypothesised that performance in comprehension and problem-solving tasks will be better for subjects using 3-d ICDs than for those using either 2-d node and link diagrams or text. Specifically, the experimental hypotheses are as follows.

H_1: Time taken to perform comprehension and problem-solving tasks using 3-d ICD representations which make meaningful use of depth information will be less than that taken to perform the same tasks using 2-d node and link or textual representations of the same information.

H_2: Less errors will be made in the performance of comprehension and problem-solving tasks using 3-d ICD representations than in the performance of the same tasks using 2-d node and link or textual representations of the same information.

10.2.1 Method

10.2.1.1 Design

The design proposed for use in this experiment is summarised in figure 10.1.

It is suggested that an independent groups design (similar to Wright and Reid's) should be used so that each subject would work only with one format of knowledge representation. Each group should contain 18 subjects (since Wright and Reid's experiment obtained significant results with 17 subjects per group). Group I would work with only textual representations, group II with only 2-d, and group III with only 3-d.

Each group would see representations of a total of 4 different knowledge structures. The first of these ('P' in figure 10.1) would be the same in each case and would be used in performing 5 practice tasks. Thus group I would start with 5 trials performed using a textual representation; group II with a 2-d representation; and group III with a 3-d representation of that same structure.

Representations of the remaining 3 knowledge structures ('A', 'B' and 'C' in figure 10.1) would be used to perform the experimental tasks. Presentation order for these representations would be controlled within groups in order to eliminate the possibility of ordering effects. Thus for each group, 6 subjects (subgroup '(i)' in figure 10.1) would see representations of the remaining structures in order ABC, 6 (subgroup '(ii)') in order BCA and 6 (subgroup '(iii)') in order CAB. All subjects would thus perform the same 30 experimental tasks using different (*ie* 3-d, 2-d or textual) representations of the same three knowledge structures.

Between group variation of factors other than that of primary interest (the dimensionality of the representations used) would, of course, need to be carefully controlled. The way in which factors such as the operations available for manipulating different displays and the information content of different displays could be controlled is discussed in sections 10.2.1.2.2 and 10.2.1.2.3.

		Practice Trials	Experimental Trials		
Group I: Textual	(i) Know. Struct.	P	A	B	C
	Task	P1 P2 P3 P4 P5	A1 A2 A3 A4 A5 A6 A7 A8 A9 A10	B1 B2 B3 B4 B5 B6 B7 B8 B9 B10	C1 C2 C3 C4 C5 C6 C7 C8 C9 C10
	(ii) Know. Struct.	P	B	C	A
	Task	P1 P2 P3 P4 P5	B1 B2 B3 B4 B5 B6 B7 B8 B9 B10	C1 C2 C3 C4 C5 C6 C7 C8 C9 C10	A1 A2 A3 A4 A5 A6 A7 A8 A9 A10
	(iii) Know. Struct.	P	C	A	B
	Task	P1 P2 P3 P4 P5	C1 C2 C3 C4 C5 C6 C7 C8 C9 C10	A1 A2 A3 A4 A5 A6 A7 A8 A9 A10	B1 B2 B3 B4 B5 B6 B7 B8 B9 B10
Group II: 2-d	(i) Know. Struct.	P	A	B	C
	Task	P1 P2 P3 P4 P5	A1 A2 A3 A4 A5 A6 A7 A8 A9 A10	B1 B2 B3 B4 B5 B6 B7 B8 B9 B10	C1 C2 C3 C4 C5 C6 C7 C8 C9 C10
	(ii) Know. Struct.	P	B	C	A
	Task	P1 P2 P3 P4 P5	B1 B2 B3 B4 B5 B6 B7 B8 B9 B10	C1 C2 C3 C4 C5 C6 C7 C8 C9 C10	A1 A2 A3 A4 A5 A6 A7 A8 A9 A10
	(iii) Know. Struct.	P	C	A	B
	Task	P1 P2 P3 P4 P5	C1 C2 C3 C4 C5 C6 C7 C8 C9 C10	A1 A2 A3 A4 A5 A6 A7 A8 A9 A10	B1 B2 B3 B4 B5 B6 B7 B8 B9 B10
Group III: 3-d	(i) Know. Struct.	P	A	B	C
	Task	P1 P2 P3 P4 P5	A1 A2 A3 A4 A5 A6 A7 A8 A9 A10	B1 B2 B3 B4 B5 B6 B7 B8 B9 B10	C1 C2 C3 C4 C5 C6 C7 C8 C9 C10
	(ii) Know. Struct.	P	B	C	A
	Task	P1 P2 P3 P4 P5	B1 B2 B3 B4 B5 B6 B7 B8 B9 B10	C1 C2 C3 C4 C5 C6 C7 C8 C9 C10	A1 A2 A3 A4 A5 A6 A7 A8 A9 A10
	(iii) Know. Struct.	P	C	A	B
	Task	P1 P2 P3 P4 P5	C1 C2 C3 C4 C5 C6 C7 C8 C9 C10	A1 A2 A3 A4 A5 A6 A7 A8 A9 A10	B1 B2 B3 B4 B5 B6 B7 B8 B9 B10

Figure 10.1: Summary of Experiment Design

10.2.1.2 Apparatus

10.2.1.2.1 Hardware The experiment would be performed using a colour Sun 3/60 workstation suitable for running ICDEDIT.

10.2.1.2.2 Software

1. **Experiment Control Program**

   A program to run through the protocol described in section 10.2.1.3 would be required. This program would provide for a modified version of ICDEDIT and an appropriate text editor (see below) to be called upon to display the representations of knowledge structures needed.

   The program could be used to control various factors relating to the way in which different representations were displayed. Manipulation of the representations themselves would never be permitted, but the mechanism for choosing a particular view of the representation displayed at any point should be the same for 3-d, 2-d and textual versions (users might, for example, be asked to select different views using buttons on the numerical keypad in each case). The time taken for new views to be displayed should also be controlled: delays might need to be built in to the experiment control program to ensure that new views of textual representations appeared no more quickly than views of 2- or 3-d representations.

2. **Modified Version of ICDEDIT**

   Two- and three-dimensional representations could be presented using a modified version of ICDEDIT in which direct manipulation facilities were not made available and swapping in of different views of a representation and of new diagrams always took the same amount of time as described above.

3. **Text Editor**

   Textual representations could be presented using a modified version of a text editor in which text could be viewed and different 'pages' of text could be selected. Time to swap between pages would need to be the same as that taken to swap between views or diagrams in ICDEDIT (see above).

10.2.1.2.3 Stimuli As described in section 10.2.1.1, representations of three different knowledge structures would be required for the experimental trials. It is proposed that representations used in this experiment should be derived from those developed on the basis of knowledge engineers' suggestions during the case studies described in chapter 7.

Representations developed in studies 2, 4 and 6 would be good candidates. Each of these representations made meaningful use of depth information and each was judged by the relevant knowledge engineer to have been successful in representing important characteristics of the knowledge structures concerned. They also provide a reasonable spread of representations of different kinds of knowledge structures which means that the results of the experiment would be generalisable in this respect: of the three identified above, one represents part of a frame-based system, one part of a rule-based system and one represents part of a system written in a purpose-built notation involving aspects of rule and frame-like structures. Note that of the other representations developed during

the case studies, that developed in study 1 would not be suitable for use in this context since it made no meaningful use of depth; that developed in study 3 might not be chosen because it was judged to be rather complex; and that developed in study 5 might not be chosen because it made less use of depth information than other examples of object or frame-based systems such as those in 4 and 6.

For each of the 3-d representations chosen, corresponding 2-d and textual versions would need to be developed. All three different representations would need to convey the same information and would need to be displayed in the same number of views. For example, in the case of the representation from study 2, all the necessary information could conveniently be displayed in three two-dimensional views: one showing one hierarchical relation between entities; another showing a second relation; and a third showing the relations between entities in the two different hierarchies. This would mean that three views of the three-dimensional representation would also need to be provided and that the textual representation would need to be displayed in three separate views. Examples of 3-d, 2-d and textual stimuli derived from the representation developed for case study 2 are shown in figures 10.2 - 10.4. Information represented for study 4 could conveniently be displayed using only two 2-d views and only two views of the 3-d structure and two screens of text would therefore need to be provided in this case. In the case of the representation developed for study 6, four views of each kind would be needed.

All views of all representations should fit within the window provided so that no scrolling or movement of the viewport would be needed in order to see the whole of a particular view of the stimulus. This would further reduce the possibility of extraneous between group variations in time spent performing tasks. 2-d and 3-d representations should also, as far as possible, use nodes and links of the same size and type to reduce the possibility that differences in performance could be caused by differences in representations other than those with which the experiment is primarily concerned.

Finally, keys explaining the semantics of each of the representations used would need to be provided on paper. These should take a form similar to that used in the case studies (see chapter 7).

10.2.1.2.4 Tasks Different sets of tasks would need to be developed for use with each of the experimental stimuli. According to the design described above, 10 tasks are needed in each case.

Tasks would in general take the form of multi-choice questions requiring retrieval of information from the relevant knowledge structure. These tasks should be answered by the subjects as quickly and accurately as possible. Questions relating to a particular knowledge structure would be of varying difficulties; some would simply require subjects to search for the relevant information, while others would require a certain amount of reasoning. Easier questions would be asked first.

A sample set of questions intended for use with the stimuli shown in figures 10.2 - 10.4 is shown in figure 10.5 where the correct answers are shown underlined. Note that some of the concepts used in the questions (*eg* instantiability, inheritance, containment, and reference) are specific to the knowledge structure represented.

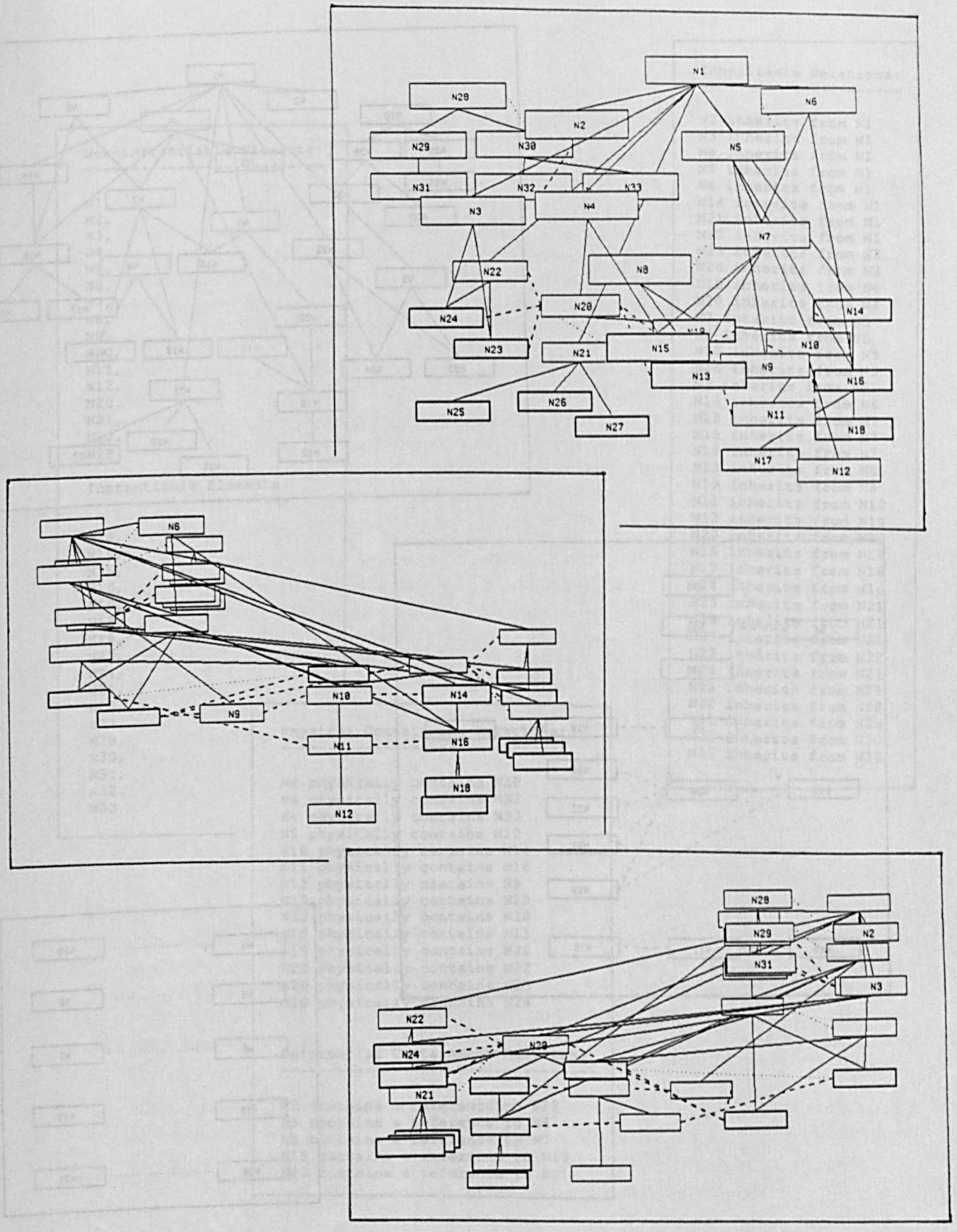
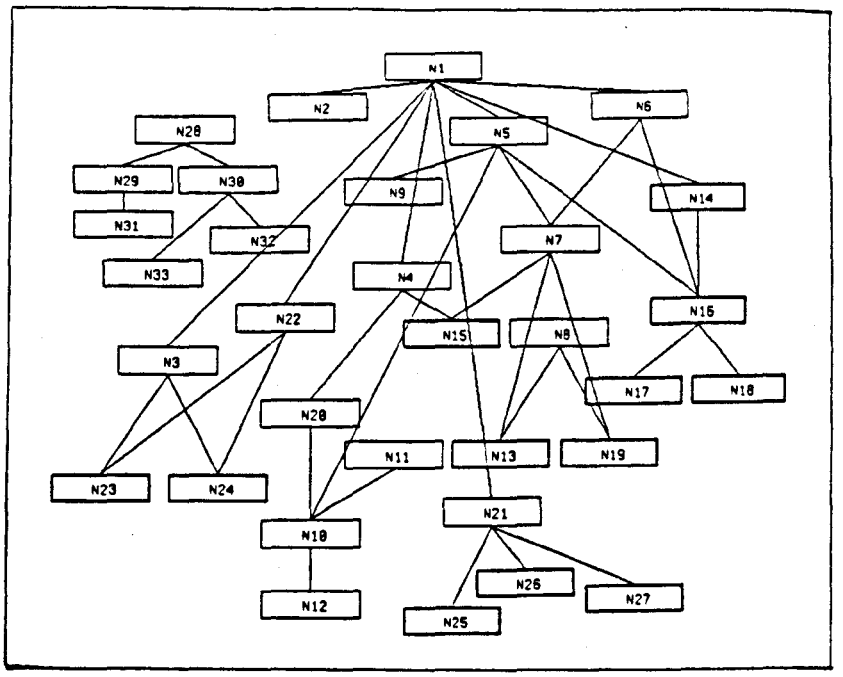
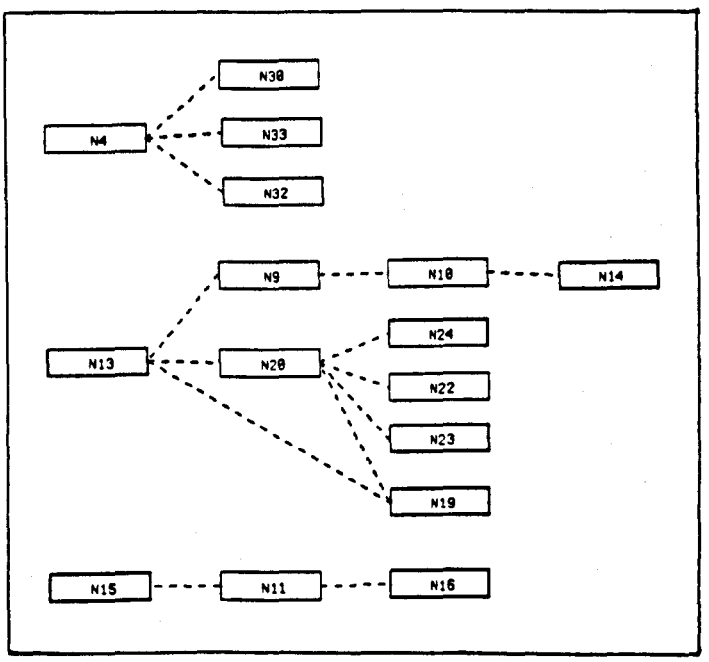


Figure 10.2: Examples of 3-d Stimuli





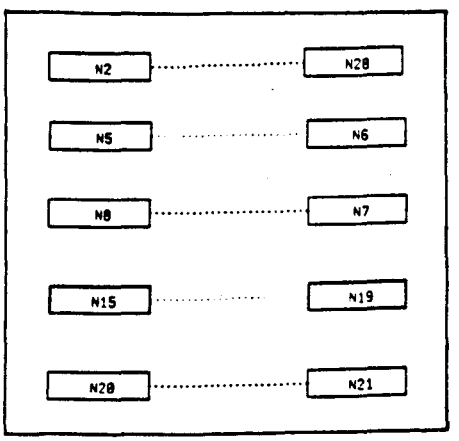


Figure 10.3: Examples of 2-d Stimuli

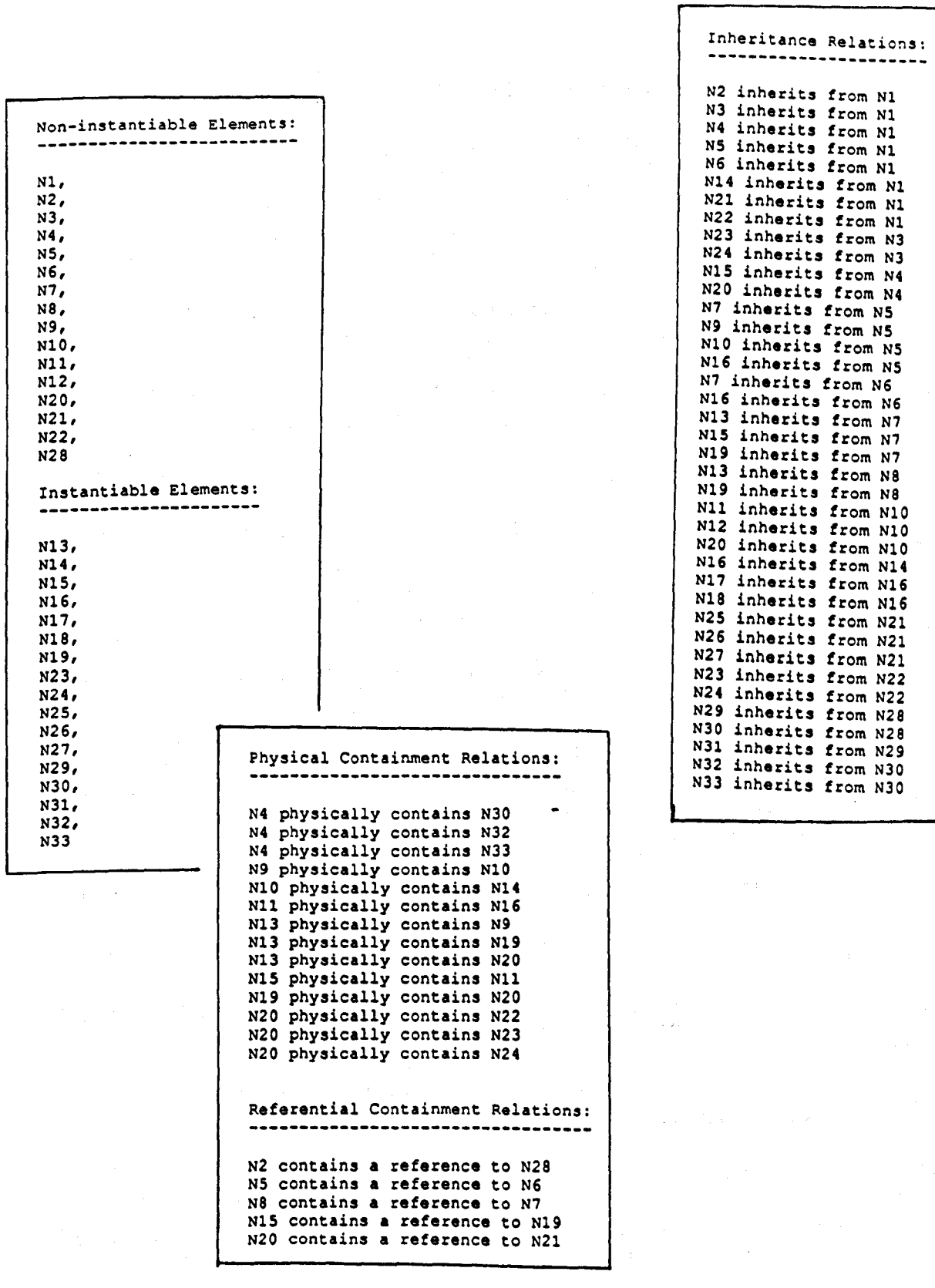


Figure 10.4: Examples of Textual Stimuli

A1) Which of the following is an instantiable element ?

a) N7 b) N12 c) N33

A2) Which of the following inherits from N10 ?

a) N15 b) N12 c) N21

A3) Which of the following contains a reference to N28 ?

a) N2 b) N29 c) N32
--

A4) Which of the following is physically contained by N4 ?

a) N1 b) N15 c) N33

A5) Which of the following is an instantiable element which physically contains N20 ?

a) N19 b) N18 c) N10

A6) Which of the following is an instantiable element which inherits from N8 and physically contains N20 ?

a) N13 b) N21 c) N11

A7) Which of the following is physically contained by a non-instantiable element which inherits from N10 ?

a) N12 b) N23 c) N9

A8) Which of the following is a list of all non-instantiable elements which are referenced by a non-instantiable element ?

a) N20,N22,N30,N33 b) N6,N7,N19,N21 c) N6,N7,N21,N28

A9) Which of the following is a list of all the instantiable elements which inherit from N7 ?

a) N13,N15 b) N6 c) N13,N15,N19

A10) Which of the following is a list of all non-instantiable elements which are physically contained by any other element ?

a) N14,N16,N30,N32,N33 b) N9,N10,N11,N20,N22 c) N6,N7,N19,N21,N28

Figure 10.5: Examples of Questions

10.2.1.3 Procedure

Subjects would be tested individually.

Each subject would begin by performing five practice tasks. These tasks would be the same for all subjects and would always be based on representations of the same knowledge structure. For group I, the stimulus used in practice trials would be a textual representation of this structure, for group II it would be a 2-d representation and for group III it would be 3-d. After the practice tasks, subjects would have the opportunity to ask the experimenter any questions before moving on to the experimental tasks.

The experimental tasks would be performed in three blocks of ten, each block using a representation of a different knowledge structure (referred to as A, B and C and derived from studies 2, 4 and 6 respectively). Subjects would be able to signal that they were ready to start a new block by pressing a button on the keyboard. They would at this point be directed to familiarise themselves with the key describing the representation to be used in the coming block of tasks. This key would be provided on paper and would be available for use at any time during the relevant group of tasks.

Subjects would also be able to signal that they were ready to carry out a new task by pressing a button on the keyboard. Pressing this button would cause two windows to be displayed on the screen: a small one containing a description of the task and a list of the possible answers; and a larger one containing the initial view of the representation to be used in solving the problem presented. Subjects would be able to select alternative views of the same representation in order to solve the problem presented (see section 10.2.1.2.2). When subjects were satisfied that they had solved the problem, they would press the button corresponding to the appropriate answer and the two windows would be removed from view. The time from the initiation of the task to the subject's response would be recorded automatically. (Note that this method of presenting tasks and stimuli approximates to the light box method used by Wright and Reid.)

10.2.1.4 Subjects

Three groups of 18 subjects (making a total of 54) would be needed for the design proposed above. Subjects should ideally be of a similar age and experience and equal numbers of females and males should be used. Subjects could be assigned to experimental groups by first categorising them according to sex and degree of experience (high or low), and then assigning equal numbers of subjects from each category (female, high experience; male, high experience; female, low experience and male, low experience) randomly to each group.

Subjects should have some familiarity with knowledge-based systems but need not be very experienced in the construction or use of such systems. It is suggested that students who have taken a course in knowledge-based systems might be suitable candidates.

10.2.2 Results

Response times would be analysed using an analysis of variance with subject group (I, II or III, corresponding to use of textual, 2-d or 3-d representations) as the main variable. Knowledge structure (A, B or C) and task number (1 - 10) should be included in order to

remove unwanted variance. The task number variable should be nested within knowledge structure.

Response time data would probably need to be logged before carrying out the analysis of variance in order to lessen the skewness of the distribution.

Mean log response times for subjects using 3-d, 2-d and textual representations should be plotted in three separate graphs, one for each of the knowledge structures (A, B and C). In each case, the y axis should represent the group mean of the logged response times and the x axis should represent the task number (1 - 10). Some predictions as to the form such results might take are shown in figure 10.6.

Errors rates should be analysed using a logistic regression with subject group as the main variable. Knowledge structure (A, B or C) and task number (1 - 10) should be included in order to remove unwanted variance. The task number variable should again be nested within knowledge structure. Note that because the total number of errors is likely to be low, the results of such a test are quite likely not to be significant.

Total numbers of errors for subjects using 3-d, 2-d and textual representations should be tabulated in three separate tables, one for each of the knowledge structures. In each case, different rows in the table should be used for errors made using different representation formats and different columns should be used for different tasks. Some tables of this form are shown in figure 10.7.

10.3 Discussion

10.3.1 Problems with the Use of Controlled Experiments in Design

As described earlier, the aim of collecting data relating to the utility of 3-d ICD interfaces for knowledge engineering tools was to determine whether further work on the development of a tool such as ICDEDIT was justified and, if so, at what aspects of the tool such work could most usefully be directed. Few attempts to obtain empirical data of this kind have so far been made. For example, although the SemNet project produced software to support the development and use of 3-d diagrams in knowledge engineering, the benefits of employing such software have apparently been investigated only informally [52]. One reason for this is simply that the technology for producing 3-d node and link diagrams suitable for use in knowledge engineering has only recently become available as explained in chapter 3. However, a more important reason is, perhaps, the difficulty of obtaining such data from controlled investigations of the kind proposed above.

It has been argued [83] that the main goal of human computer interaction research (that of improving complex interactive human-computer systems) 'is not sufficiently served by the standard tools of experimental psychology such as factorial controlled experiments on preplanned variables.' Young and Barnard have also argued that 'it is not necessarily appropriate for HCI to adopt the methodologies of cognitive psychology and cognitive science as they stand' and a number of other authors have expressed the concern that efficient progress towards the design of better user interface software cannot be made in this way [103].

Problems with the use of controlled experiments in this context are as follows. First, the classical controlled experiment 'produces too little information and requires too much to

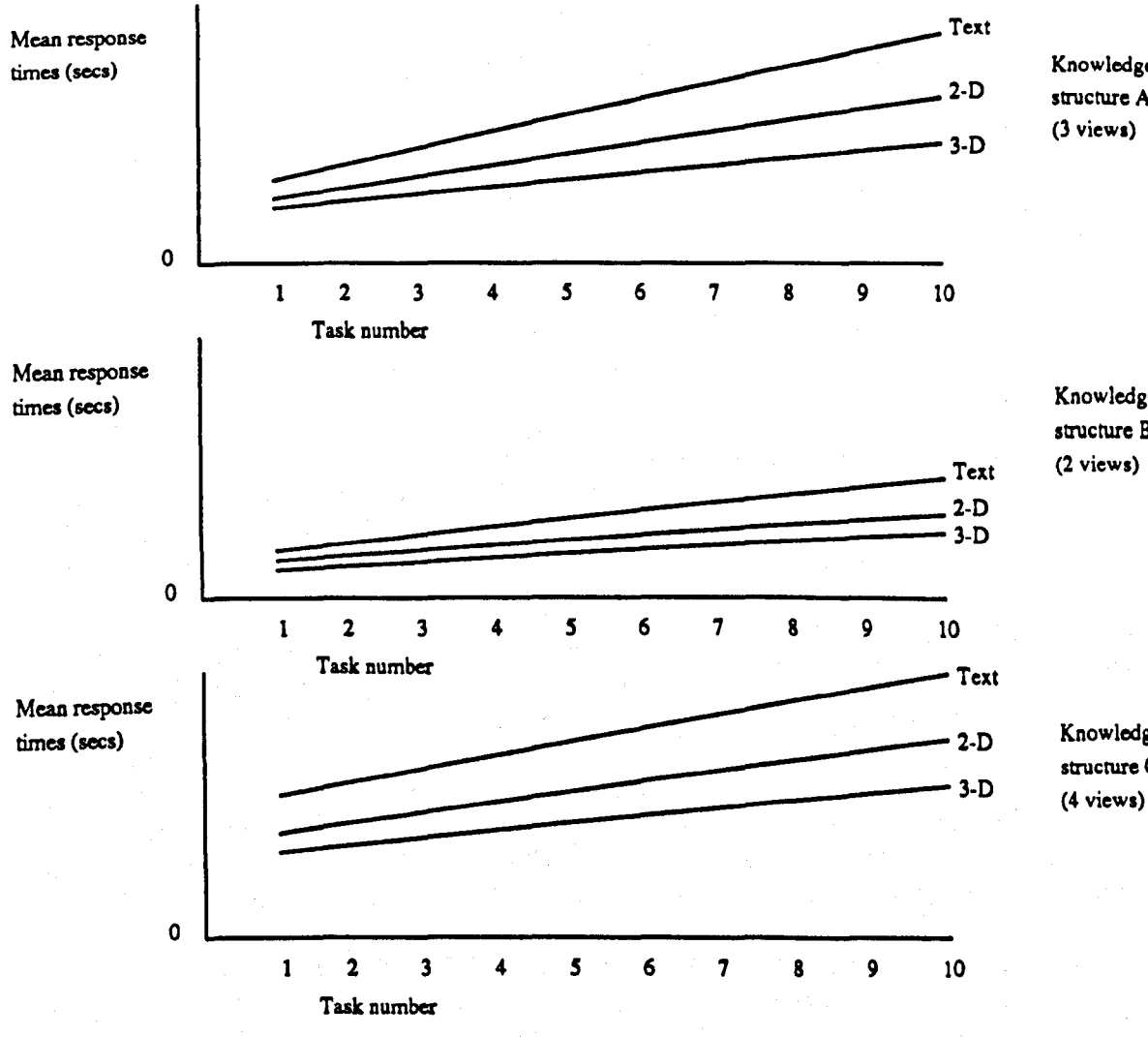


Figure 10.6: Predicted Results

	Task number									
	1	2	3	4	5	6	7	8	9	10
Text										
2-D										
3-D										

Knowledge structure A (3 views)

	Task number									
	1	2	3	4	5	6	7	8	9	10
Text										
2-D										
3-D										

Knowledge structure B (2 views)

	Task number									
	1	2	3	4	5	6	7	8	9	10
Text										
2-D										
3-D										

Knowledge structure C (4 views)

Figure 10.7: Error Rate Tables

be known in the choice of experimental variables, to be much use in exploring the functions needed by humans in a novel task.' If the conclusions drawn from the results of such experiments are to be reliable, then the experimental environment must be tightly controlled, but while so little is known of the way in which the situation under investigation works, it is difficult to know what aspects of that situation to control.

The isolation of particular features for study can, furthermore, be misleading. Since the systems which are the object of study are intended to be used without restrictions in a real setting, 'research that aims at understanding important design differences needs to study them in place ...and needs to study their effects on complete tasks that users will actually perform, rather than only on isolated subtasks.' [83]. For example, a major limitation of the framework for experimental investigation proposed above results from the fact that users are asked only to view the 3-d representations and never to edit or manipulate them. This kind of constraint has to be imposed in an experimental setting in order that task performance conditions can be standardised across different levels of the variable of interest. The introduction of the different kinds of functionality needed to manipulate the textual, 2-d and 3-d representations would lead to an uncontrollable explosion in the number of variables and interactions which would need to be considered. Permitting subjects to manipulate representations used in solving problems would also have introduced a high degree of variance into the response time data and would have meant that the comparison between groups was effectively one between different tools rather than simply one between the three different forms of presentation. In practice, however, many of the knowledge engineer's tasks are likely to involve editing representations of knowledge structures and the ability to carry out such manipulations is likely, in itself, to contribute to the engineer's understanding of those structures. Excluding the possibility of such manipulations from formal experiments means that the applicability of experimental findings to real situations will be somewhat limited.

A further problem concerns the generalisability of the results of such experiments. For

example, in the experiment proposed above, stimuli might easily have been derived from any of the knowledge structures in a different way. Representations used might also have differed in layout, or in terms of the underlying scheme for representation. While parameters important in producing particular layouts or developing particular schemes for representation are so badly understood, it is impossible to control rigorously for variations in such factors.

Finally, it might be argued that in applied research, we should be principally interested not in whether there is any effect at all of some variable, but rather in how large the effect is. According to Landauer [83] 'It is not enough, and often not even very interesting, simply to know that a choice has a "statistically significant" effect'.

What we need, then, are 'data collection methods that are sufficiently quick and economical' and permit us to make 'good judgements as to what functions and methods of operation will be most effective in helping people to get results that are valuable to them' [83]. The following section makes some suggestions about methods which might be used to investigate the utility of 3-d ICD interfaces in a variety of knowledge engineering situations so that further development work can be directed at answering real needs.

10.3.2 An Alternative Approach to Assessing Utility

A number of problems with using conventional controlled experiments to collect empirical data relating to the utility of 3-d ICD interfaces were identified above. This section suggests ways in which these problems may be addressed and outlines proposals for an alternative approach to the collection of such data.

Rather than carrying out experiments, it is suggested that further investigation of the utility of 3-d ICD interfaces should involve the use of scenario-based studies (see, for example, [82, 98] and [161]) in which subjects could be asked to carry out more realistic knowledge engineering tasks. Protocols collected during the performance of such tasks could be analysed with the aim of characterising user performance at the level of common errors or difficulties and identifying causes in aspects of the situation that we can control. The results of such an analysis could then be used to direct further developments.

In such studies, the problem of not knowing what aspects of the experimental situation to control or what data to collect could be addressed by collecting rich data. Data about the participants should include information about age, sex, experience and other relevant characteristics (such as skills in visualisation), and data collected during the study might include information about the times taken to perform low level tasks, the context, nature and frequency of errors, and the subjective reactions of the participants to various aspects of the system. Information about other aspects of the experimental scenario (relating, for example, to the physical environment, the equipment used, and the number and significance of other people in the same room) should also be collected. Landauer [83] recommends that 'the set of observations and conditions used be such that a large number of possible outcomes, expected as well as unexpected, are allowed to occur, be noticed and be measured. The trick is to design experiments so that many things can happen and be observed. Don't unnecessarily restrict the possible things that subjects can do; allow them some way to make free responses that the experimenter can observe.' Differences in the experience or situation of various participants can in this way lead to the discovery of interesting hypotheses rather than simply undermining generality.

The problem of the lack of applicability of results from studies of isolated variables could be addressed by requesting subjects to carry out tasks which are as realistic as possible in real contexts. The subjects should also be representative of the target user population: ideally, a random sample from the population of interest. In order to address the problem of generalisability (in this case, of the results of a study using one representation to use of other representations) each subject could be asked to carry out the same kinds of task using a range of different representations, and their responses to each could be collected and compared.

Finally, it is proposed that verbal protocols should be collected both during (see [50]) and after task performance. Statistical analysis of data concerning task completion times and errors might indeed yield useful information and help to guard against the drawing of unwarranted conclusions. However, the qualitative analysis of verbal protocols is likely to uncover information about the reactions, concerns and preoccupations of real users which must be of central importance in developing a tool which will be useful as well as usable, and which might not be discovered through quantitative analysis alone.

Representations for use in scenario-based studies could be designed using either ICDEDIT or a graphical interface prototyping tool such as Macromind's 'Director' [90]. Representations of a range of knowledge structures of different types (for example object- or frame-based as well as rule-based) would ideally be developed, with 3-d, 2-d and text-based versions of each. Initial designs could be evaluated by HCI experts or graphical designers according to certain criteria (for example, how easily they were judged to support users in identifying information of a certain kind) and refined on the basis of their comments.

Representations developed in this way could then be presented to subjects (ideally knowledge engineers) who would be asked to perform a range of simulated knowledge engineering tasks. Task descriptions would be at a higher level than those used in the case studies, allowing more flexibility in the choice of mechanisms by which to carry them out. They should be representative of a range of activities, with attention being paid to design, debugging and maintenance, identified in chapter 8 as being of particular interest in this context. Support for these tasks would ideally be provided by a version of ICDEDIT modified in response to the findings of the case studies reported in chapters 7 - 9. Valuable information could, however, also be gained from studies simulating the kind of facilities which should be made available using tools such as Director (see above) and Swivel 3D Professional [111] which would permit 'real' interaction to be simulated using animated sequences of screen images (see, for example, [138]).

Studies would be carried out as described above, collecting rich data (including verbal protocols) before, during and after the task performance sessions. Low-level tasks and errors observed might then be classified in such a way as to permit a quantitative analysis of data about task completion times and error rates. The data should also be analysed in a qualitative manner. This analysis would support the development of a process model of the search strategies employed in browsing complex images, and would allow system designers to assess the reactions of users to particular features of the representations or facilities supporting their use. The results of both quantitative and qualitative analysis would, it is argued, contribute to a greater understanding of the relative utility of 2- and 3-d ICD representations in a range of knowledge engineering situations. Since scenario-based studies consume far fewer resources (in terms of subject, investigator and system time) than controlled experiments, it would be possible to investigate a much broader

efficiently.

10.4 Summary

This chapter has suggested ways in which further empirical data concerning the relative utility of 2- and 3-d ICD interfaces in knowledge engineering could be collected. It described a conventional controlled experiment which could be used to investigate the degree of support provided by 3-d, 2-d and textual representations of knowledge structures for knowledge structure comprehension. An argument was, however, made that the goal of determining what further development of a tool such as ICDEDIT would be most useful would not be well served by the use of such controlled experiments. The problems associated with using such experiments in this context were discussed and an alternative approach to gathering empirical data of the kind needed was proposed.

Chapter 11

Formal Specification of 3-d ICD Languages for Knowledge Structure Representation

11.1 Introduction

The aim of this chapter is to lay the foundations for further developments of a tool such as ICDEDIT by formally specifying the range of 3-d ICD languages such a tool might aim to support. ICDEDIT is a tool intended to support the use of a particular interface technology (*ie* one relying on the use of 3-d ICDs). A specification of key aspects of that technology (*ie* of the interface objects available for manipulation and the way in which they may be manipulated) will therefore be central to a specification of the tool as a whole.

Section 11.2 defines the conceptual framework within which specifications have been constructed. The range of 3-d ICD languages which future versions of a tool such as ICDEDIT might aim to support is then specified in section 11.3. The way in which this permits specification of particular languages for knowledge representation which a future version of ICDEDIT might be configured to support is illustrated in section 11.4 using a worked example based on the second of the case studies described in chapter 7.

Specifications presented in this chapter are written using the Z notation [134]. Z uses the principles of set theory, predicate logic, relations and functions and provides a means of structuring specifications in such a way that different facets of a system may be described separately and then related or combined. Z schemas can be used to describe both static and dynamic aspects of a system. Descriptions of static aspects include specifications of system states and the invariant relationships that are maintained as the system moves from state to state. Dynamic aspects specified include possible operations, relationships between the inputs and outputs of such operations and accompanying changes of state. A brief introduction to aspects of the Z notation used here is given in appendix F.

The Z notation is increasingly being used in system specification as the benefits of early formal specification in terms of reductions in overall system development costs are becoming more apparent [131]. The advantages and disadvantages of using Z in the manner proposed in this chapter is discussed in section 11.5.

11.2 Definition of Visual Languages

The most fundamental idea relevant to the specifications presented in this chapter is that of a *visual language.* A visual language is defined simply to be a set of pictures [67] or 'presentations' [87]. Visual languages, like other languages, rely on the use of a set of primitive symbols which may be arranged and combined according to certain rules corresponding to the language's syntax.

Different authors have expressed different views on the kind of information which should be expressed in such rules and on the way in which they should be used.

Borning [11] proposed that visual languages should be specified in terms of predicates specifying constraints and methods for producing diagrams in which those constraints held, all this being written in Smalltalk.

Mackinlay [87] followed Bertin's classification [8] of the properties of visual primitives or 'marks' into the categories of 'retinal' and 'positional'. He suggested that individual pictures or presentations could be thought of as sets of objects located within a 2-dimensional plane, and might therefore be specified by defining positional, spatial and retinal properties of their primitive components. The syntax of a new language could therefore be defined by predicates expressing constraints on these properties which must be satisfied by all components of a particular picture in order for it to qualify as a well-formed expression in that language.

Golin and Reiss described a third method for specifying the way in which graphical primitives or picture components may be composed or connected. Permissible graphical primitives were defined as specialised instances of the two basic types of shapes and lines; operators defining the permissible compositions of those primitives in terms of their positions in a 2-dimensional plane were to be specified as productions in an 'attributed multiset grammar' [66].

Kamada and Kawai's approach [76] is also based on the assumption that diagrams may be viewed as arrangements of graphical objects in which the abstract relations between abstract objects represented may be shown as 'graphical relations'. Once again, the set of allowable graphical relations is taken to include both 'geometric' ('composition') and 'connection' relations as well as attribute relations which allow the specification of objects of different types.

The approach to visual language specification described in the following sections is based on similar assumptions to those used by Mackinlay, Golin and Reiss and Kamada and Kawai. In order to specify the syntax of a new visual language, one must first specify the graphical primitives to be used. Various types of graphical primitive (intended to be used to represent significant types of knowledge structure components) may be specified in terms of the visual attributes of members of the type and the vocabulary of a particular language may then be defined in terms of the types of graphical primitives it uses. The way in which graphical primitives may be combined in diagrams may then also be specified, in terms of both their relative positions (defined by *composition constraints*) and other relationships between them (some of which may be defined in terms of appropriate *connection constraints*). Finally, since diagrams of languages supported by ICDEDIT are intended to be interactive, operations on diagrams may also be specified.

11.3 Specification of 3-d ICD Languages Supported by ICDEDIT

This section specifies important properties of the range of languages currently supported by ICDEDIT. This specification is intended to form the basis for specifications of future tools which will support a similar range of 3-d ICD languages.

11.3.1 Abstract Graphs

As stated above, ICDEDIT is a tool intended to support the use of 3-d ICDs as an interface technology independent of any particular application (though interest here has, of course, focused mainly on the use of such technology in the interface to tools for knowledge engineering). Representations supported by ICDEDIT are therefore defined in terms of abstract graphs, whose relations with any underlying application are intended to be specified independently.

This approach is similar to the one taken by Kamada and Kawai in producing TRIP, a tool for the production of graphical representations of abstract structures of interest [76]. In TRIP, the problem of producing graphical representations of knowledge structures is seen as one of carrying out two transformations, first from program structure to abstract graph, and next from abstract graph to graphical representation.

The only basis type needed to specify an abstract graph is:

$$[NODE]$$

Links may be defined in terms of nodes:

Link

$$Origin : NODE$$
$$Dest : NODE$$
$$Origin \neq Dest$$

and graphs in terms of nodes and links:

Graph

$$Nodes : \mathbb{P}\, NODE$$
$$Links : \mathbb{P}\, Link$$
$$\forall l : Links \bullet \{l.Origin, l.Dest\} \subseteq Nodes$$

The first of these definitions states that a link is specified in terms of its origin and destination nodes and that for any given link, the origin and destination nodes must be different (or in other words that the class of graphs to be considered are irreflexive). The second schema states that a graph can be defined as consisting of a set of nodes and a set of links which must only connect nodes in the set specified, that is, in the same graph.

11.3.2 Graphical Primitives

This section describes the graphical primitives used in languages supported by ICDEDIT.

First, we define a basic type:

$[COLOUR]$

which will be used in describing the colours of various components of the nodes and links. Note that because of the hardware restriction described in chapter 1, only 4 different colours may be displayed by ICDEDIT at any one time. This means that all languages supported by ICDEDIT must use a maximum of 4 different colours (although these colours may be chosen from a large selection). We may therefore define a global constant:

$$| \; NumberOfColours : \mathbb{N}_1$$

whose value is taken to be 4 under the current implementation.

ICDEDIT also provides a number of line styles which can be used to draw node borders and links. As well as solid lines, it provides dashed and dotted lines and two distinct patterns made up of combinations of dots and dashes:

$$LineStyle ::= solid \mid longdashed \mid shortdashed \mid dotted \mid pattern1 \mid pattern2$$

Lines of various widths or thicknesses can be drawn. ICDEDIT provides for 3 different thicknesses (as well as allowing the user to specify that no line should appear with *noline*):

$$LineWidth ::= thick \mid medium \mid thin \mid noline$$

Individual nodes can be given textual labels, so we define a further basic type:

$[TEXT]$

These textual labels can be written in a number of styles. The text style *upper* refers to text written in upper case, *lower* to lower case, *capitalised* to text with initial letters of each word written in upper case, and *numeric* to text consisting purely of numbers. Defining a text style as *misc* implies that some other convention (or no convention at all) has been used. Again the user may specify that no textual labeling should be used with *notext*.

$$TextStyle ::= upper \mid lower \mid capitalised \mid numeric \mid misc \mid notext$$

In ICDEDIT, links are characterised not only in terms of the style and thickness of lines with which they are drawn, but also in terms of two further properties. The first is *JoinStyle* which describes whether a particular type of link will always join the top of one node to the bottom of another, or the side of one to the side of its opposite number.

$$JoinStyle ::= topbottom \mid sideside$$

The second is *ArrowStyle* which describes whether a link is shown with an arrow head to indicate directionality, and if so whether the head is open or closed.

$$ArrowStyle ::= open \mid closed \mid noarrow$$

Now we can define the way in which the graphical primitives provided by ICDEDIT can support the definition of distinct graphical vocabularies of node and link types which can be used in the representation of particular kinds of knowledge-based system.

11.3.3 Vocabularies

Using the available graphical primitives, the vocabulary for a new visual language may be specified by defining the various types of nodes and links which will be used to represent entities and relations of various types.

The visual representation of a node can be seen as consisting of a number of components. In order to define the kind of visual node which will be used to represent a particular kind of element in a particular knowledge-based system, ICDEDIT allows us to define values for the significant characteristics of the following visually salient components:

VisualNodeType

$Height, Width : \mathbb{N}_1$
$BodyColour, BorderColour, LabelColour : COLOUR$
$BorderStyle : LineStyle$
$BorderWidth : LineWidth$
$LabelStyle : TextStyle$

This schema describes the fact that we may define a class or type of visual nodes in terms of the height or width of its members, the colours used to draw their bodies, borders and labels, the line style used to draw their borders, the width of their borders or the style in which their textual labels must be written.

Similarly for the visual representation of a kind of link which will be used to represent a particular kind of relationship between elements in a knowledge-based system, we need to specify the significant characteristics of the following components:

VisualLinkType

$Line : LineStyle$
$Colour : COLOUR$
$Join : JoinType$
$Arrow : ArrowStyle$

This definition of the notion of node and link typing was included here as types such as these were used in developing the case study representations. It should, however, be noted that the current version of ICDEDIT provides no support for the definition of types of node and link so that they must simply be defined independently by the user.

We can now define the fact that the vocabulary used by any visual language of the kind supported by ICDEDIT must consist of the following components and satisfy at least the following description:

Vocabulary

$$
\begin{array}{l}
NodeTypes : \mathbb{P}\ VisualNodeType \\
LinkTypes : \mathbb{P}\ VisualLinkType \\
Colours : \mathbb{P}\ COLOUR \\
\hline
\#Colours = NumberOfColours \\
\forall l : LinkTypes \bullet l.Colour \in Colours \\
\forall n : NodeTypes \bullet \\
\quad n.BodyColour \in Colours \land \\
\quad n.BorderColour \in Colours \land \\
\quad n.LabelColour \in Colours
\end{array}
$$

This schema describes that fact that a visual language of the kind supported by ICDEDIT may draw on a vocabulary of node and link types. Four colours are available for the display of the various node and link type components, and all components of all node or link types in the vocabulary must be displayed using one of the colours defined.

11.3.4 Diagrams

A diagram in ICDEDIT is made up of sets of visual nodes and links representing nodes and links in a particular abstract graph.

A node in a diagram drawn in a particular language must be of a type defined to be part of the vocabulary of that language. It may also be given a textual label. A graphical or visual node may therefore be defined as follows:

VisualNode

$$
\begin{array}{l}
VisType : VisualNodeType \\
Label : TEXT
\end{array}
$$

Individual nodes will be placed in unique positions in 3-space which are defined by the positions of their centres on the x, y and z dimensions. It is therefore useful to define a position in 3-space as:

Position

$$X, Y, Z : \mathbb{Z}$$

A visual link must be of a type defined in the language and must always originate and end up at a visual node:

VisualLink

$$
\begin{array}{l}
VisType : VisualLinkType \\
Origin, Dest : VisualNode
\end{array}
$$

A diagram can now be defined as a graphical representation of an abstract graph (derived from the knowledge-based system of interest) in which abstract nodes and links are represented using graphical nodes and links drawn according to the specifications of the appropriate visual language vocabulary.

Diagram

$Graph$
$Vocabulary$
$NodeRep : NODE \nrightarrow VisualNode$
$NodePos : NODE \rightarrowtail Position$
$LinkRep : Link \nrightarrow VisualLink$

$\text{dom } NodeRep = \text{dom } NodePos = Nodes$
$\text{dom } LinkRep = Links$
$\forall l : \text{dom } LinkRep \bullet$
$\quad LinkRep(l).Origin = NodeRep(l.Origin) \wedge$
$\quad LinkRep(l).Dest = NodeRep(l.Dest)$
$\forall n : \text{ran } NodeRep \bullet$
$\quad n.VisType \in Vocabulary.NodeTypes$
$\forall l : \text{ran } LinkRep \bullet$
$\quad l.VisType \in Vocabulary.LinkTypes$

11.3.5 Visual Languages

In concordance with the definition presented in section 11.2, we may now define a visual language simply as a set of diagrams:

VisualLanguage

$Diagrams : \mathbf{P}\ Diagram$

Note that the definition of *Diagram* here is specific to ICDEDIT, in that it assumes the existence of a particular set of graphical primitives using which the vocabulary of a diagram must be constructed. This definition of *VisualLanguage* is therefore also specific to the context of ICDEDIT.

11.3.6 Primitive Operations

The framework set out above may now be used as a basis for the specification of primitive operations in 3-d ICD languages supported by ICDEDIT.

11.3.6.1 Changes in Layout

The simplest kinds of operations are those which lead only to changes in the layout of a diagram and not to any change in the structure represented or the vocabulary used.

A formal specification of this kind of operation may be given as follows:

$$
\begin{array}{|l}
\hline
\Delta Layout \\
\Delta Diagram \\
node? : NODE \\
\hline
\Xi Graph \\
\Xi Vocabulary \\
NodeRep' = NodeRep \\
LinkRep' = LinkRep \\
node? \in Nodes \\
\{node?\} \lhd NodePos' = \{node?\} \lhd NodePos \\
\hline
\end{array}
$$

This states that a change in layout is a change in a diagram in which the structure (seen as an abstract graph) and vocabulary remain unchanged. The mapping from nodes in the abstract graph to visual representations of nodes is unchanged and the mapping from links to visual representations of links is also unchanged. The positions of all nodes except *node?* are unchanged.

Note that in practice new positions of nodes resulting from changes in layout of diagrams in particular languages would be constrained by the specifications of those languages, just as their original positions had been.

Note also that nothing has been said here of links. Since links are defined with respect to nodes, their positions change only when the positions of their origin or destination nodes change.

11.3.6.2 Changes in Structure

Other operations on diagrams are related to changes in the structures or abstract graphs represented. The general form of such operations may be specified as:

$$
\begin{array}{|l}
\hline
\Delta Structure \\
\Delta Diagram \\
\hline
\Xi Vocabulary \\
\hline
\end{array}
$$

In other words, we may specify an operation resulting in a change in the structure of a graphical representation as a change in a diagram which leaves the vocabulary used in that diagram unaffected. There are a number of such operations, some of which are described below.

11.3.6.2.1 Deletions The operations of this kind which can be most simply specified are deletions. Deletions of links may be specified as:

$$
\begin{array}{|l}
\hline
DeleteLink \\
\Delta Structure \\
link? : Link \\
\hline
Graph'.Nodes = Graph.Nodes \\
Graph'.Links = Graph.Links \setminus \{link?\} \\
NodeRep' = NodeRep \\
NodePos' = NodePos \\
\{link?\} \lhd LinkRep' = \{link?\} \lhd LinkRep \\
\hline
\end{array}
$$

This states that the deletion of a link is a change in structure (as defined above) which leaves the set of nodes in the abstract graph and the way in which they are represented unchanged. The link in question is removed from the set of links in the abstract graph and is therefore no longer represented graphically. Graphical representations of other links are unchanged.

Node deletions are as follows:

DeleteNode

$$
\begin{array}{l}
\Delta Structure \\
node? : NODE \\
linkset? : \mathbb{P}\, Link \\
\hline
\forall l : Link \bullet l.Origin = node? \vee l.Dest = node? \leftrightarrow l \in linkset? \\
Graph'.Nodes = Graph.Nodes \setminus \{node?\} \\
Graph'.Links = Graph.Links \setminus linkset? \\
\{node?\} \lhd NodeRep' = \{node?\} \lhd NodeRep \\
\{node?\} \lhd NodePos' = \{node?\} \lhd NodePos \\
linkset \lhd LinkRep' = linkset \lhd LinkRep
\end{array}
$$

This describes the fact that the deletion of a node involves the removal of both that node and the set of links for which it was either the origin or the destination from the abstract graph. Graphical representations of other components are unchanged.

11.3.6.2.2 Additions The next operations to be specified are those involving the addition of nodes or links to the abstract graph and the corresponding augmentation of the graphical representation.

The addition of a new node may be described as follows:

AddNode

$$
\begin{array}{l}
\Delta Structure \\
node? : NODE \\
noderep? : VisualNode \\
nodepos? : Position \\
\hline
Graph'.Nodes = Graph.Nodes \cup \{node?\} \\
Graph'.Links = Graph.Links \\
NodeRep' = NodeRep \cup \{node? \mapsto noderep?\} \\
NodePos' = NodePos \cup \{node? \mapsto nodepos?\} \\
LinkRep' = LinkRep
\end{array}
$$

The operation of adding a node involves adding an abstract node to the set of nodes in the abstract graph and a representation of that node to the graphical representation of the graph as a whole. The set of links in the abstract graph and their representations remain unchanged.

The addition of a new link is slightly more complicated:

AddLink

$\Delta Structure$
$link? : Link$
$linkrep? : VisualLink$

$Graph'.Nodes = Graph.Nodes$
$Graph'.Links = Graph.Links \cup \{link?\}$
$NodeRep' = NodeRep$
$NodePos' = NodePos$
$LinkRep' = LinkRep \cup \{link? \mapsto linkrep?\}$
$linkrep?.Origin = NodeRep(link?.Origin)$
$linkrep?.Dest = NodeRep(link?.Dest)$

Once again, an abstract link is added to the set of links in the abstract graph and a representation of that link is added to the graphical representation of the graph as a whole while nodes and their graphical representations are unchanged. The final two lines of the schema specify the fact that the graphical nodes linked in the representation must correspond to the abstract nodes connected by the link in question in the underlying graph.

11.3.6.3 Changes in Appearance

The final kind of operation which may be performed on a diagram of the kind specified is one which causes a change in the visual representation of some component of the diagram while leaving the structure and layout of that diagram unchanged.

The general form of such operations may be specified as:

ΔAppearance

$\Delta Diagram$

$\Xi Graph$
$\Xi Vocabulary$
$NodePos' = NodePos$

Note once again that new properties of nodes or links resulting from operations of this kind being performed on diagrams in particular languages would be constrained by the specifications of those languages, just as their original properties had been.

Two operations which lead to changes in visual representation of this kind are described below.

11.3.6.3.1 Changing a Node The operation of editing a node may be specified as:

EditNode

$$\Delta Appearance$$
$$node? : NODE$$

$$LinkRep' = LinkRep$$
$$NodeRep'(node?).Label \neq NodeRep(node?).Label \vee$$
$$NodeRep'(node?).VisType.Height \neq NodeRep(node?).VisType.Height \vee$$
$$NodeRep'(node?).VisType.Width \neq NodeRep(node?).VisType.Width \vee$$
$$NodeRep'(node?).VisType.BodyColour \neq NodeRep(node?).VisType.BodyColour \vee$$
$$NodeRep'(node?).VisType.BorderColour \neq NodeRep(node?).VisType.BorderColour \vee$$
$$NodeRep'(node?).VisType.LabelColour \neq NodeRep(node?).VisType.LabelColour \vee$$
$$NodeRep'(node?).VisType.BorderStyle \neq NodeRep(node?).VisType.BorderStyle \vee$$
$$NodeRep(node?).VisType.BorderWidth \neq NodeRep(node?).VisType.BorderWidth \vee$$
$$NodeRep'(node?).VisType.LabelStyle \neq NodeRep(node?).VisType.LabelStyle$$

This states that at least one of a node's label, height, width, body colour, border colour, label colour, borderstyle, border width or label style must change as a result of an operation of type *EditNode*. Representations of links are left unchanged.

11.3.6.3.2 Changing a Link Similarly, we may state that at least one of a link's line style, colour, join type and arrow style must change as a result of an operation of type *EditLink*:

EditLink

$$\Delta Appearance$$
$$link? : Link$$

$$NodeRep' = NodeRep$$
$$LinkRep'(link?).VisType.Line \neq LinkRep(link?).VisType.Line \vee$$
$$LinkRep'(link?).VisType.Colour \neq LinkRep(link?).VisType.Colour \vee$$
$$LinkRep'(link?).VisType.Join \neq LinkRep(link?).VisType.Join \vee$$
$$LinkRep'(link?).VisType.Arrow \neq LinkRep(link?).VisType.Arrow$$

11.4 Specification of a Particular 3-d ICD Language

This section illustrates the way in which the scheme described above forms a basis for the specification of particular languages for knowledge representation which a future version of ICDEDIT might be configured to support.

It describes the specification of the particular language of which the representation developed in the second of the case studies was an example. It begins with a brief description of important characteristics of the knowledge-based system to be represented using this language. This is followed by an exposition of requirements for the graphical representation of relevant components of that system which were elicited from the subject during stage 2 of the study. A formal specification of the language developed is given, together with some examples of specialised operations for use with diagrams of the relevant kind. Figures showing the diagrams used in the study are shown in appendix D. Descriptions

of representations developed for the other studies (studies 1,3,4,5 and 6) are included in appendix G.

11.4.1 Knowledge-Based System Profile

Hardware platform: general purpose workstation

Software: Lisp environment

Architecture: mainly frame-based

Size: unknown

General Description:

The system as a whole was intended to assist expert communication network managers. Important entities in the part of the system under consideration were frames representing the logical and physical components of a telephone network. The aim of the graphical representation was to depict two independent types of relationship on the main components in the knowledge base, the first being inheritance, and the second containment. These two types of relationship operated orthogonally on the same set of objects.

11.4.2 Representation Requirements

Requirements on the representation specified by the subject in study 2 were as follows:

11.4.2.1 Vocabulary

The following types of entity were to be represented:

E1 logical components

E2 physical components

Specific requirements regarding the way in which these types of entities were to be represented were as follows:

E3 the same text style and border style should be used by nodes representing either type of component

E4 colour and border thickness should be used to discriminate between nodes representing logical and physical components

E5 nodes representing physical components should appear 'more real' so should be drawn in stronger colours and with thicker borders

Types of relation to be represented were:

R1 super/subclass

R2 physical containment

R3 referential containment

The physical containment relationship holds in the case where one physical component of a network actually contains another. The referential containment relationship holds in the case where one component contains a logical reference to another.

Requirements regarding the way in which these types of relationship were to be represented were:

R4 different types of link should be used to represent the different types of containment relationship

R5 the referential containment links should look less substantial than the physical containment links

R6 arrow heads should not be used

11.4.2.2 Connection Constraints

Constraints on the types of relationships which could hold between entities of particular types were as follows:

C1 all components are related by super/subclass relations

C2 only physical components may be related by the physical containment relationship

11.4.2.3 Composition Constraints

Finally, elements of the graphical vocabulary were to be composed or laid out according to the following principles or constraints:

P1 the super/subclass relationship should be shown in the y dimension

P2 the containment relationships should be shown in depth (*ie* in the z dimension)

11.4.3 Representation Specification

A language which would satisfy the requirements described above was developed by the designer as part of the case study. That language may be specified as follows.

11.4.3.1 Vocabulary

First, a range of colours for use in the specification was defined. Two similar colours differing mainly in saturation were chosen for use in representing the physical and logical components (see requirement N4). A dark colour which would show up well against either of these was chosen for use in labelling nodes and drawing links, and a pale shade of the colour used for the nodes was chosen for the background. Colours chosen may be identified by global constants as follows:

$$L2LightNodeCol, L2DarkNodeCol, L2OutlineCol, L2BackgroundCol : COLOUR$$

All nodes were of the same height and width, and were wider than they were high. We may therefore define two further global constants and an invariant on them as follows:

$$L2NodeHeight, L2NodeWidth : \mathbb{N}_1$$

$$L2NodeWidth > L2NodeHeight$$

All nodes had a number of common properties which were defined as follows:

L2CommonNodeType

VisualNodeType

$Height = L2NodeHeight$
$Width = L2NodeWidth$
$BorderColour = L2OutlineCol$
$LabelColour = L2OutlineCol$
$BorderStyle = solid$
$LabelStyle = capitalised$

Note that the border style and text style for use in labeling are here defined to be the same for all types as specified in requirement E3.

Nodes representing components of different types were distinguished by a difference in body colour and border thickness (see requirement E4):

L2LogicalComponentType

L2CommonNodeType

$BodyColour = L2LightNodeCol$
$BorderWidth = thin$

L2PhysicalComponentType

L2CommonNodeType

$BodyColour = L2DarkNodeCol$
$BorderWidth = thick$

Nodes representing physical components are made to seem 'more real' than those representing logical components (as specified in E5) by the use of thicker borders and a more saturated body colour.

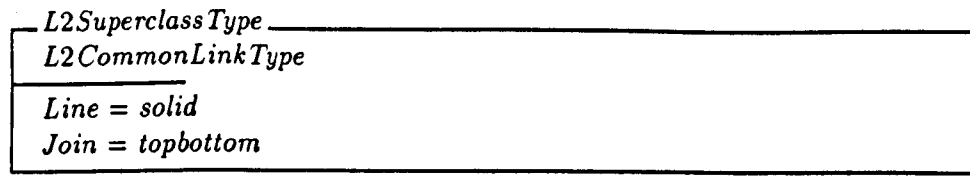
All links were the same colour, and none had arrow heads (see requirement R6):

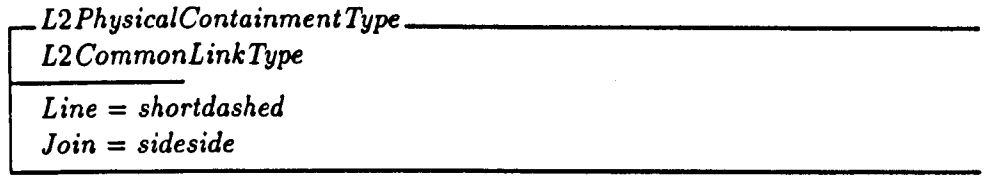
L2CommonLinkType

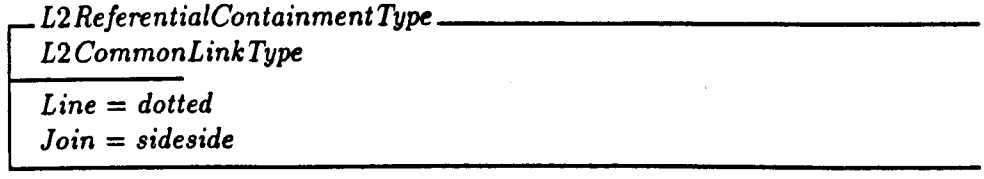
VisualLinkType

$Colour = L2OutlineCol$
$Arrow = noarrow$

Links representing the different kinds of relationship were distinguished by the type of line with which they were drawn and by the way in which they joined associated nodes.







Links representing containment relationships can be distinguished from those representing the super/subclass relationship by the fact that they join nodes in a side-side fashion, rather than from top to bottom. Links representing different types of containment relationship can be further distinguished by their use of different line types (requirement R4). Those representing physical containment are shown using stronger, more substantial dashes than those representing referential containment (requirement R5).

11.4.3.2 Language

The complete visual language may now be defined in terms of its graphical vocabulary and the various connection and composition constraints that must be satisfied by any diagram in that language.

Language2

VisualLanguage

$$
\begin{array}{l}
\forall\, d \in Diagrams \bullet \\
\quad d.Vocabulary.NodeTypes = \{L2LogicalComponentType, \\
\qquad L2PhysicalComponentType\} \wedge \\
\quad d.Vocabulary.LinkTypes = \{L2SuperclassType, \\
\qquad L2PhysicalContainmentType, L2ReferentialContainmentType\} \wedge \\
\quad d.Vocabulary.Colours = \{L2LightNodeCol, L2DarkNodeCol, \\
\qquad L2OutlineCol, L2BackgroundCol\} \\
\forall\, d \in Diagrams \bullet \\
\quad (\forall\, n : \text{ran } d.NodeRep \bullet \\
\qquad \exists\, l : \text{ran } d.LinkRep \bullet \\
\qquad\quad l.VisType = L2SuperclassType \wedge \\
\qquad\quad (l.Origin = n \vee l.Dest = n)) \wedge \\
\quad (\forall\, l : \text{ran } d.LinkRep \mid \\
\qquad l.VisType = L2PhysicalContainmentType \bullet \\
\qquad\quad (l.Origin).VisType = L2PhysicalComponentType \wedge \\
\qquad\quad (l.Dest).VisType = L2PhysicalComponentType) \\
\forall\, d \in Diagrams \bullet \\
\quad (\forall\, l : Link \mid \\
\qquad d.LinkRep(l).VisType = L2SuperclassType \bullet \\
\qquad\quad NodePos(l.Origin).Y > NodePos(l.Dest).Y) \wedge \\
\quad (\forall\, l : Link \mid \\
\qquad (d.LinkRep(l).VisType = L2PhysicalContainmentType \vee \\
\qquad d.LinkRep(l).VisType = L2ReferentialContainmentType) \bullet \\
\qquad\quad NodePos(l.Origin).Z < NodePos(l.Dest).Z)
\end{array}
$$

The first block of constraints describes the graphical vocabulary available for use in constructing diagrams in the visual language defined for case study 2. This vocabulary consists of two different types of node, three types of link and four colours.

The second block of clauses below the line describe the connection constraints which must hold for any diagrams in *Language2*. The first of the two clauses in this block states that all nodes must be either the origin or the destination of a link representing a superclass relationship (see requirement C1). The second states that physical containment relationships may only be represented as holding between physical components (as specified by requirement C2).

The last block of clauses below the line describe composition constraints or constraints on layout which must hold for any diagrams in *Language2*. These state that super-/subclass relationships must be represented such that the node representing the superclass is displayed higher than the node representing the subclass (see requirement P1), and that nodes representing components which either physically or referentially contain others must be shown in front of those representing the components contained (see requirement P2).

All the requirements expressed by the user have now been translated into formal constraints on the vocabulary which may be used in a diagram of the language described and the way in which elements of the vocabulary may be composed. The specification of the design can therefore be said to be complete in this respect.

11.4.3.3 Operations

Operations of the kind specified in section 11.3.6 can also be specialised for use with diagrams from particular languages.

Specifications of operations of type $\Delta Layout$ (for changing the layout of a diagram) or $\Delta Appearance$ (for changing the graphical representation of a diagram component) would remain unchanged. Specifications of deletion operations (*DeleteLink* and *DeleteNode*) would remain unchanged but specifications of operations involving the addition of nodes and links would need to be augmented.

Examples of specialisations of the *AddNode* operation are:

L2AddLogicalComponent

$$AddNode$$

$$noderep!.VisType = L2LogicalComponentType$$

and:

L2AddPhysicalComponent

$$AddNode$$

$$noderep!.VisType = L2PhysicalComponentType$$

An example of the specialisation of the *AddLink* operation is:

L2AddSuperclassLink

$$AddLink$$

$$linkrep!.VisType = L2SuperclassType$$

11.5 Discussion

Work on specifying the syntax of visual languages of various kinds has been done by a number of authors (see, for example, [12], [87], [67] and [76]). Different authors have proposed different approaches to specification depending on the kind of language and the context within which specifications are expected to be used.

As suggested above, the specification of the range of languages currently supported by ICDEDIT (described in section 11.3) is intended to be used as a basis for the development of further versions of a tool supporting a similar range of languages. It also forms a basis for developing specifications of particular languages in the manner described in section 11.4.

It is envisaged that such specifications of particular languages could in future be developed by specialised visual language designers. Such designers would first consult potential users to gather their requirements for the representation of particular kinds of abstract structure. They would then be able to draw on both their skill in visual language design and their appreciation of the user's requirements to develop a suitable language. Formal specification of the languages of interest is intended to support the development of a tool

which would permit visual language designers to define new languages on the basis of particular sets of constraints derived from user representation requirements as described above.

This process to visual language design is similar to that envisaged by Golin and Reiss [67] and also respects the need for a user-centred approach to design identified by Kamada and Kawai [76]. Further development of a visual language support tool such as ICDEDIT might provide an interface which would allow users to specify their own requirements for a particular visual language directly or graphically in the way that Borning's extension to ThingLab permitted the graphical definition of constraints on graphical representations [12].

The use of the Z notation in developing specifications of the kind described in this chapter was felt to be reasonably successful. It shared the advantages of all formal specification techniques of promoting clarity of thinking and facilitating the identification of errors and inconsistencies in the initial informal view. The use of Z in particular also means that certain properties of specifications (such as internal logical consistency) could be checked using standard Z support tools such as CADiZ [74]. Z specifications are sufficiently abstract to be hardware and software independent. Instead of specifications being relevant only to the particular environment for which they were constructed (as, for example, in the case of Borning's Smalltalk specifications), they form a potential basis for implementations in many different environments. Finally, the fact that Z uses both mathematical and logical constructs means that its expressive power is considerably greater than that of first order logic alone. It is possible to use the scheme described in section 11.3 as a basis for specifying both static and interactive features of a wide range of 3-d ICD languages as demonstrated by its use in the specifying each of the languages developed in the case studies (see appendix G).

The main disadvantage of using Z for the specification of 3-d ICD languages is that it is not obviously suitable for expressing the kind of heuristics or 'soft constraints' commonly used in laying out diagrams in those languages. Chapter 6 identified certain arrangements of nodes as causing problems with the accurate perception of relative depths in 3-d ICDs. Attempts must be made to avoid such arrangements where possible, but the 'soft constraints' imposed by such problems must, on certain occasions, be subordinated to the demands of the 'hard constraints' specified in the description of the relevant language. Thus, for example, the soft constraint or heuristic for aesthetic layout stating that 'the children of a parent node should ideally be displayed symmetrically on either side of the parent along an axis orthogonal to that in which degree of seniority in the hierarchy is displayed' might sometimes have been subordinated to a more important constraint (specified in the relevant language) stating that a further relationship was to be coded in terms of the relative positions of relevant nodes along that orthogonal axis. The Z notation is not ideally suited for the expression of such potential tradeoffs and further work would need to be done to investigate the possibilities of expressing both 'hard' and 'soft' constraints in an integrated manner. A further factor, which might in future prove to be a disadvantage, is that specifications written in Z might not form a suitable basis for permitting users to specify their own visual languages as described above.

Finally, it should be noted that while the use of Z in this context apparently holds considerable promise, further work is needed in order to determine the kind of scheme which would yield most benefits. There are many different ways of constructing specifications such as those described in this chapter using the Z notation and further work might

ideally investigate the relative merits of a range of schemes in the light of considerations such as those described above.

11.6 Summary

This chapter has presented a scheme for the formal specification of 3-d ICD languages for knowledge structure representation which is intended to act as a foundation for further work on the development of a tool such as ICDEDIT. The way in which this scheme permits individual 3-d ICD languages to be specified is illustrated in a worked example which describes the specification of a language developed for use in one of the case studies. Specifications have been written using the Z notation, and the advantages and disadvantages of using Z in the manner proposed in this chapter were briefly discussed.

Chapter 12

Discussion

12.1 Overview

The research described in this thesis has investigated human factors aspects of the use of 3-dimensional Interactive Connection Diagrams in the human-computer interface of tools for knowledge engineering.

A number of empirical studies were conducted. The main aim of studies carried out during phase one was to investigate whether users were able make effective judgements about relative depths of components in 3-d ICDs produced using the approach embodied in JIN, JINGLE and ICDEDIT. This was done in order to determine whether such 3-d ICDs constituted a reasonable medium on the basis of which to conduct further investigations into the meaningful use of depth-based spatial coding in graphical representations of knowledge structures for use in knowledge engineering.

A particular objective of this phase was to determine whether the use of a relatively unusual cue to depth (that of simulated rocking motion) would enhance the effectiveness with which such judgements could be made. This objective was met by carrying out a number of controlled experiments to evaluate the ability of subjects to make accurate judgements about relative depths in 3-d ICDs of various kinds undergoing various kinds of motion. These experiments were described in chapters 5 and 6. On the basis of the results of these experiments it was concluded that:

- users are able to make reasonably effective judgements about the relative depths of components in 3-d ICDs.

The aim of studies carried out during the second phase was twofold. The first objective was to estimate the utility in various knowledge engineering situations of tools incorporating 3-d ICDs of the kind supported by ICDEDIT. Information regarding utility gathered from the case studies was used to make recommendations concerning the direction of further work on the development of tools supporting the use of 3-d ICDs in knowledge engineering. The second objective was to evaluate the usability of depth-related features of ICDEDIT. A number of recommendations regarding ways in which such features could be implemented in future versions of ICDEDIT were made on the basis of this evaluation.

The case studies carried out in order to fulfill these last two objectives were described in chapter 7, and relevant findings were presented in chapters 9 and 8. Claims made on the basis of the findings of these studies were as follows:

- interface technologies based on the use of 3-d ICDs will, in some knowledge engineering situations, be more useful than those which employ only more conventional 2-d versions;
- depth-related features of ICDEDIT are usable but may be improved upon in future implementations.

The final section of work, whose results are reported in chapter 11 investigated the possibility of formalising specifications of the 3-d representations used in the case studies described above. As a result of this work, it was claimed that:

- 3-d ICD representations of knowledge structures for use in knowledge engineering can be formally specified using the Z notation.

This chapter presents an evaluation of the work reported earlier. This evaluation will consider both the success of the work itself (in terms of the general significance of its findings and its applicability in related domains) and its relation to other work with similar aims. The chapter ends with a discussion of some possible directions for future work and a short section presenting some overall conclusions.

12.2 Evaluation

One of the first projects to explore the use of three-dimensional graphical representations of large and complex abstract structures such as those at the heart of knowledge-based systems was the SemNet project. A tool which allowed users to view and manipulate 3-dimensional node and link representations of knowledge bases was developed as part of this project and a number of approaches to various issues were informally evaluated. Aspects of the problem discussed by Fairchild *et al.* in their description of the SemNet project [52] include techniques for the positioning of individual diagram components in 3-d space, strategies for coping with volume and complexity of information to be represented and the provision of functions to support browsing and navigation through the three-dimensional structure.

Until recently, there has been little further work on the use of three-dimensional representations of such abstract structures. In the last two years, there has, however, apparently been renewed interest in the area. Shum [128] describes how 'a number of researchers see 3-d displays as holding considerable promise as a future medium for human-computer interaction' and some examples of further experiments with the use of 3-dimensional representations have come to light. For example, Barnett *et al.* [5] have used 3-dimensional charts to represent parse spaces produced by Lucy, a knowledge-based system for understanding written English (see figure 12.1), and Card *et al.* describe the use of several different kinds of three-dimensional representation (including one similar to paradigms described in earlier chapters in its use of arrangements of nodes and links in 3-space) in the interface to PARC's new Information Visualiser [116] (see figure 12.2).

Issues involved in developing tools which will adequately support the use of such 3-dimensional representations in the human-computer interface have also begun to be discussed. A panel session at the recent INTERACT90 conference [41] discussed the use of 3-dimensional representations in tools for the development of large and complex software systems, and Shum [128] discusses the relevance of research on spatial cognition to the development of systems using a 'spatial metaphor'.

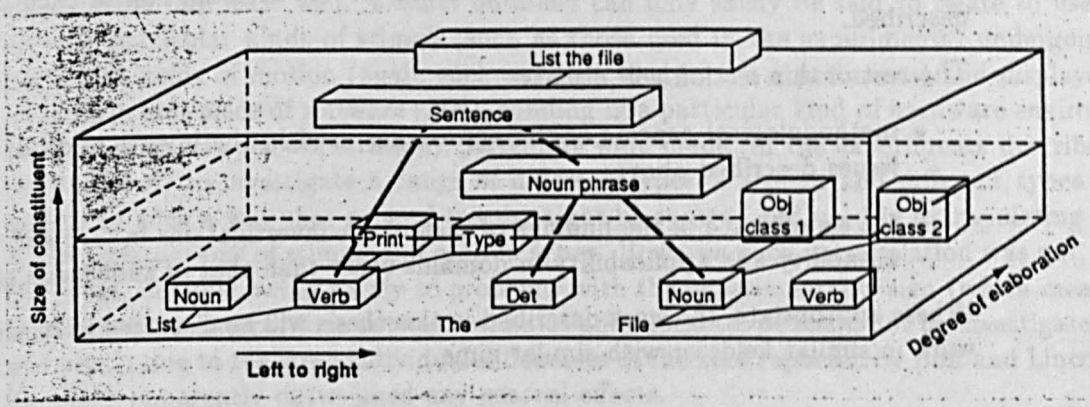


Figure 12.1: Chart of the Parse Space Produced by Lucy

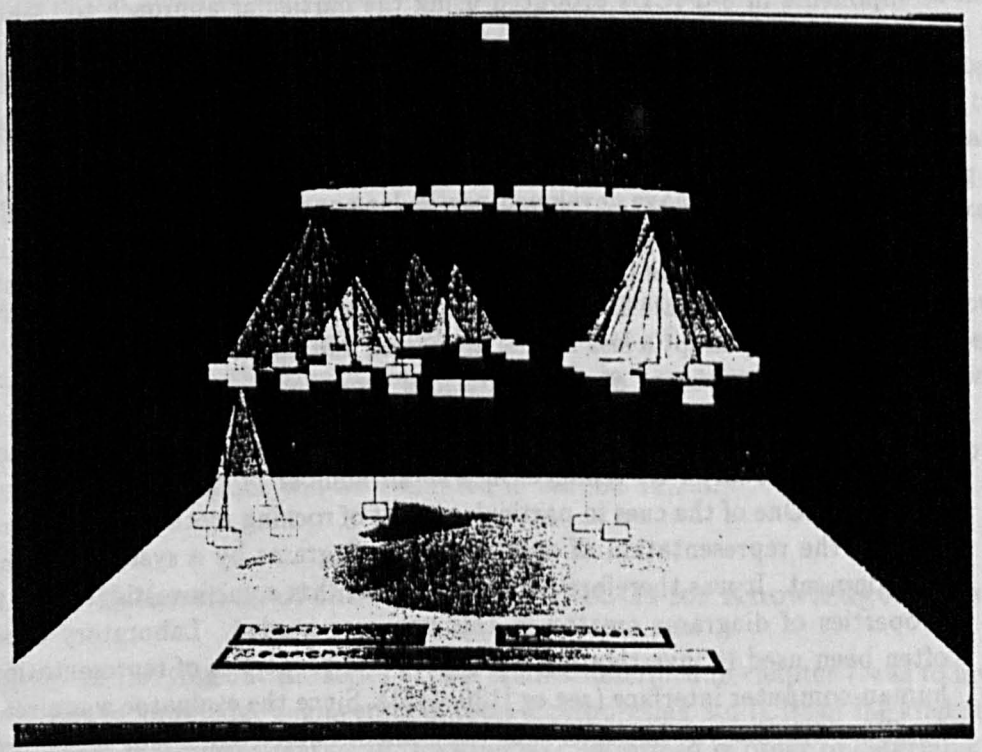


Figure 12.2: Cone Tree Used in the Information Visualiser

Of course, there are many aspects to the problem which might ideally be investigated and many possible approaches to any such investigation. The work described in this thesis has concentrated mainly on perceptual aspects of 3-d ICDs, questions of the utility in knowledge engineering of 3-d ICDs and some aspects of the usability of a particular tool which supports the use of 3-d ICDs. Some work has also been done on the possibility of formalising descriptions of 3-d ICDs for use in knowledge structure representation. Attention has been restricted to the use of such representations in knowledge engineering and particular approaches have been chosen for the investigation of each of the issues described.

The rest of this section will discuss:

- methodological aspects of the approaches used in the investigation of each of the issues described above
- the significance of the findings of each of the investigations in terms of their generalisability and applicability in domains other than that of knowledge engineering

Where appropriate, the work described in this thesis will be discussed in relation to other work in similar fields or with similar aims.

12.2.1 Investigation of Depth Cuing in 3-d ICDs

As stated above, the main aim of the experiments reported in chapters 5 and 6 was to investigate whether users were able to make effective judgements about relative depths of components in 3-d ICDs produced using the particular approach to creating an illusion of depth described in chapter 1. These investigations focussed mainly on perceptual aspects using response time as a measure of the cognitive effort required to make such judgements. Accuracy and subjective feelings of ease, confidence and comfort were also considered.

Numerous projects have developed particular approaches to creating the illusion of depth in various kinds of graphical representation and using various kinds of hardware [88, 73, 19, 149, 116, 52]. Some [149, 19] have used controlled experiments to investigate whether the illusion they create is in some sense reliable from the user's point of view. Others [73, 88, 116, 52] adopt a less formal approach, and apparently rely on their own intuitions regarding the reliability of representations produced.

The approach to creating the illusion of depth used in JIN, JINGLE and ICDEDIT is relatively unusual in its reliance on a small number of cues and its use of 2-d graphics hardware. One of the cues in particular - that of rocking motion - had not previously been used in the representation of node and link diagrams by a system running in a similar environment. It was therefore felt at the outset that some investigation of the perceptual properties of diagrams created in this way was needed. Laboratory experiments have often been used in investigating the perceptual properties of representations used in the human-computer interface (see eg [130, 145]). Since the evaluator was already reasonably familiar with the way in which such experiments were conducted, the choice of controlled experimentation as a method for the investigation of the effectiveness of depth cuing in JIN seemed natural.

Experiments of the kind described above should ideally yield results which are objective, rigorous, and generalisable across a useful range of situations. Some of the results

obtained from the experiments described in chapters 5 and 6 were indeed objective. Response time and accuracy were objective measures firstly of the extent to which the cues used in JIN provided effective support for users making judgements about the relative depths of components in 3-d ICDs, and secondly of the way in which the use of rocking motion impacted on the effectiveness of that support. A number of problems relating to the rigour of the experimental design have already been discussed in chapters 5 and 6, but it is perhaps with respect to issues of generalisability that the results of these experiments fare least well. Results obtained can only safely be said to relate to users viewing particular kinds of stimuli (such as those used in the experiments) undergoing particular kinds of motion (again such as those used in the experiments) and displayed by a particular piece of software (JIN), running in a particular kind of hardware environment (a Whitechapel workstation). Attempts were made (in the experiments described in chapter 6) to investigate a range of different types of stimuli and different types of motion in such a way that general trends might be spotted and used in extrapolating to other combinations of stimuli and motion types. However, such extrapolation was not, in the event, possible owing partly to problems with the experimental design (which meant that important interactions between stimulus and motion type could not be investigated) and partly due to the large individual differences of the kind reported by Boff and Lincoln [9] which apparently outweighed any general effects.

As a result of the work described in chapters 5 and 6, it was concluded that the question of the effectiveness of a particular approach to creating an illusion of depth in 3-d node and link diagrams is a hard one to investigate in such a formal manner. Many factors are obviously involved, and since the effects of many of these factors is unknown, it is hard to control for them in any rigorous manner. Even if such controls could reasonably be imposed, the thorough investigation of all relevant factors would require a large number of studies and huge amounts of laboratory time, and would mean that technology (and the market) would be likely to have progressed considerably during the time taken to conduct such studies. It is therefore recommended that future work on similar issues should adopt a less formal approach to such an investigation, possibly placing greater emphasis on the comments (such as those reported in chapters 5 and 6) of subjects having used the system of interest under a variety of conditions.

Despite the problems described, the results of the experiments did give some grounds for confidence that the approach to creating the illusion of depth was reasonably effective, and that it was worth using a similar approach in a subsequent tool for viewing and editing 3-d ICDs. Studies of the usability of this subsequent tool (reported in chapter 9) used a less formal approach to investigating the effectiveness of cues to depth. The success of this approach will be evaluated in section 12.2.3.

12.2.2 Estimation of the Utility of 3-d ICDs for Knowledge Engineering

One aim of carrying out the series of case studies described in chapter 7 was to investigate the utility in knowledge engineering of tools incorporating 3-d ICDs of the kind supported by ICDEDIT. An investigation of this kind was performed in order to determine whether further development of the existing prototype was justified and, if so, at which aspects of knowledge engineering activity potential future implementations of a tool incorporating 3-d ICDs would most usefully be directed.

As described in chapter 7, the concept of utility has not commonly been considered in

recent years. Taking the definition of the utility of an interface technology set out in chapter 7 ('the ability, capacity or power of of that technology to support the majority of users in carrying out the majority of tasks using the majority of underlying software tools in the majority of environments which they are likely to encounter'), we may, however, consider the approaches of a number of projects to investigating related aspects of particular tools or representations for use in the human-computer interface.

The SemNet project [52] adopted an informal 'exploratory' approach to investigating the problems inherent in attempting to provide the user with a 3-dimensional node and link representation of an abstract structure such as a knowledge base. Few explicit conclusions regarding the utility of such 3-d representations are drawn. In particular, there is no discussion of the relative merits of 3-d representations as opposed to 2-dimensional versions. Robertson *et al.* [116] present a brief argument in support of the claim that 3-d representations make more effective use of screen space than 2-d, but suggest that 'formal evaluation studies' should eventually be used to determine the benefits of using a 3-d layout.

Alternative approaches to defining issues relevant to the utility of environments or tools for visual programming and program visualisation are adopted by Glinert [64] and Eisenstadt *et al.* [48] who list desirable attributes and criteria against which such tools may be measured. Each of the lists is apparently the result of a thorough analysis of particular aspects of the problem. But neither say much about aspects of the user, task or physical environment (aside from the hardware environment) which are likely to have a significant influence on the utility of a tool in any real situation. Some of these aspects are discussed by Petre and Green [106] in the context of their investigation of the use of graphical notations in systems for computer-aided design for electronics, but their paper, in turn, makes little reference to the impact of aspects of the hardware and software environment.

The investigation of the utility in knowledge engineering of tools supporting the use of 3-d ICDs whose results are described in chapter 8 has attempted to take a broad view encompassing consideration of all aspects of the user, task, environment and underlying system which are likely to have a bearing on this issue. The relative utility of 3-d representations as compared with 2-d versions has also been explicitly discussed.

The success of the approach described was limited by a number of factors. As described in chapter 7, the prototypical nature of ICDEDIT meant that an investigation of the utility of 3-d ICDs in performing real knowledge engineering tasks could not be performed. The focus had instead to be on performance of 'atomic tasks' which meant that the opinions of subjects regarding the utility of 3-d ICDs in various kinds of knowledge engineering activity were formed in the absence of any real experience. The fact that subjects used ICDEDIT outside of their normal work environment also meant that judgements made might have been somewhat unrealistic. Finally, since two of the six subjects (subjects 1 and 3) had been interested in the project for some time before agreeing to take part, it is possible that their opinions may have been biased, or at least unrepresentative of those of the population of knowledge engineers as a whole.

Apart from the above limitations, it should also be noted that the method for estimation of utility proposed cannot claim to yield any objective measure of utility. Even if objective measures of utility in each aspect of a situation relevant to the case of 3-d ICD interfaces could be identified, utility of a system or interface technology with respect to such different aspects cannot be compared in any meaningful way: for example, the

benefits of using 3-d ICDs rather than 2-d in terms of the number of tasks which can be supported cannot be weighed against the benefits provided to different kinds of users. This should, however, not be seen as a disadvantage of the particular approach proposed: it is unreasonable to expect any approach to the evaluation of utility to yield such an objective measure since utility (or usefulness) will always be judged subjectively and in different ways by different potential users depending on their particular needs. The 'dimensions' of user, task, environment and system should be seen merely as prompts which will tend to encourage a broad-based approach to the investigation of utility. Comments made on particular points along each of these dimensions identified as relevant in the case of a particular system or interface technology may then be seen as as aids to potential users attempting to judge whether their particular needs are likely to be met by such a system.

An approach which would develop ideas presented in chapter 8 by facilitating a more objective assessment of some of the issues identified as relevant to the utility of 3-d ICDs for knowledge engineering was proposed in chapter 10. This approach involved collecting further empirical data from a controlled experiment and a series of scenario-based studies. The advantages and disadvantages of this approach were discussed in section 10.3. Briefly, it was suggested that while an experiment such as that proposed would support an evaluation of the overall utility of 3-d ICDs, questions concerning the relative utility of 2- and 3-d diagrams would best be answered using the results of scenario-based studies.

Apart from the limitations described above, it was felt that the interviews conducted as part of the case studies were relatively successful in permitting an estimation of the utility of tools supporting the use of 3-d ICDs in the field of knowledge engineering to be made. In an area such as knowledge engineering, the utility of a particular form of representation in the human-computer interface of relevant tools is likely to depend on a large number of factors (as indicated by the number of issues highlighted in work described above). The use of a contextual approach encouraged the consideration of a wide range of relevant issues. In comparison with the amount of work which might be wasted on developing tools which would not be useful to the knowledge engineer, the amount of effort required to carry out these case studies was very small. It seems, therefore, that this approach to conducting an initial investigation of utility while the tool to be developed is still at the stage of an early prototype is a valuable one. Furthermore, the results of such an initial investigation should provide useful information for the developers of future versions and enable work on the development and evaluation of further prototypes to be accurately targeted.

Finally, it should be noted that at least some of the findings of these studies (such as those relating to environmental and user factors) might be relevant to other projects developing tools to support the use of 3-d node and link representations in other areas of application. Further work would be needed to explore the relevance of the findings reported in chapter 8 in domains other than that of knowledge engineering.

12.2.3 Evaluation of the Usability of Features of ICDEDIT

A further aim of the case studies described in part III of this thesis was to investigate the usability of depth-related features of ICDEDIT. The focus of this investigation was on evaluating the effectiveness and satisfaction with which a group of knowledge engineers

could carry out a set of simple atomic tasks involving the use of depth information in a reasonably controlled environment. Usability of features of ICDEDIT not involved with the use of depth information was not considered.

The only other project to consider issues relating to the usability of 3-dimensional node and link diagrams in the context of knowledge engineering was the SemNet project [52]. Again, the approach to investigation of the relevant issues was exploratory and informal and concentrated on questions of how to provide the user with facilities for positioning individual components of a representation and for navigating and browsing through 3-dimensional structures. As SemNet was a fully working prototype, it was possible to address these issues in the context of a small number of real projects.

The approach to investigation of usability employed in the studies described above was more structured than that of the SemNet project in that it involved asking a number of subjects to carry out a range of tasks intended to provide information about a number of problems of interest. However, as the functionality of ICDEDIT was rather restricted, it was only possible to consider its usability in performing fine-grained (and therefore rather unrepresentative) atomic tasks. Again, the fact that stage 3 of a study took place in the relatively controlled environment of the evaluator's laboratory means that findings may have lacked ecological validity. The fact that two of the subjects had been interested in the project for some time before taking part in the case studies is not, however, so detrimental to the validity of findings regarding usability as it might have been for those relating to utility. In the case of subject 3, previous interest might well have had little impact on the effectiveness with which tasks were performed. Subject 1 was already an experienced user of ICDEDIT before participating in the case study and was therefore likely to have had fewer difficulties with exploiting the functionality of the system than other subjects. Unfortunately, since none of the other subjects were experienced users of the system, it was difficult to make any generalisations regarding differences between novice and expert performance. Future studies could explore this issue further.

At the time when the design for these studies was developed, techniques for contextual evaluation had not yet been systematically or completely presented as a methodology in the public literature [154]. Since that time, Wixon *et al.* have published a summary of the important aspects of such a procedure [154]. The procedure used in the case studies apparently conforms to most of their suggestions except that the studies reported above investigated the utility and usability of a real prototype whereas Wixon *et al.* describe working in the absence of any such prototype.

In conclusion, it was felt that the investigation of usability provided further useful information for the developers of future versions of a tool such as ICDEDIT for the viewing and editing of 3-d ICDs. Although some of the findings are relevant mainly to the development of tools using an approach very similar to that of ICDEDIT, others may have more general applicability. Once again, further work would be needed in order to explore the applicability of the findings reported in chapter 9 to the development of other tools.

12.2.4 Formal Specification of 3-d ICD Languages

As described in chapter 11, the objectives of developing a formal specification of the range of 3-d ICD languages currently supported by ICDEDIT was to lay the foundations for further developments of ICDEDIT and to provide a framework within which particular 3-d ICD languages could be specified.

The use of the Z notation in the specification of interactive visual languages is thought, at the time of writing, to be unique. The advantages and disadvantages of using the Z notation in this way were discussed in chapter 11. Briefly, it was suggested that Z was well-suited to jointly fulfilling both of the objectives described above. A Z specification of languages to be supported would form a valuable component of a specification for a further version of a tool such as ICDEDIT. Z also supported the specification of all the necessary presentational aspects of 3-d ICD languages, as well as interactive aspects which are a significant feature of such languages. The main limitation of the framework developed in Z was that it was not clear how so-called 'soft constraints' could be specified in such a way that they could be appropriately traded off against the hard constraints imposed by particular languages.

There have, it is thought, also been no other attempts to specify 3-dimensional node and link languages of any kind. Literature concerning other projects which make use of such languages (such as the SemNet [52] and Information Visualiser [116] projects) make no reference to such attempts. The approach proposed here has the advantage that the framework described in chapter 11 can easily be specialised for use in specifying 2-d node and link languages. It could also be extended to cover 3-d node and link languages based on different primitives (for example on nodes or links with different visual attributes) with relative ease, though it could not be applied to the specification of types of language other than those using nodes and links.

In summary, it seems that the work described in chapter 11 shows considerable potential and it is hoped that it will, in future, be developed further.

12.3 Further Development of Tools

A number of areas in which the research described in previous chapters could usefully be developed were mentioned in the previous section. This section discusses ways in which a tool such as ICDEDIT which allows users to view and edit 3-d ICDs could be developed in order that it might eventually be incorporated into an intelligent front end or presentation tool such as Mackinlay's APT [87] or Nong Tarlton and Tarlton's Pogo [141]. A presentation tool of this kind might then be integrated into a powerful general-purpose tool for knowledge engineering in such a way that 3-d ICDs were one of a range of representations available to the knowledge engineer for use at appropriate points in the knowledge-based system development lifecycle (although in this case, care would need to be taken that 3-d ICD representations integrated well with the other forms of representation used such that translation between, say, 2-d and 3-d versions could be performed automatically). This section considers: ways in which 3-d ICDs could be presented or displayed in such a tool; ways in which the layout of 3-d ICDs could be handled; and ways in which the high level functionality of ICDEDIT could be expanded.

12.3.1 Presentation of 3-d ICDs

The findings reported in chapters 5, 6 and 9 suggest that techniques used in presenting 3-d ICDs and in particular for creating an illusion of depth were relatively successful. These techniques could therefore be carried forward into further tools.

It was suggested in chapter 6 (section 6.5.2) that a default type of rocking motion might be provided in future tools and that the parameters of such a default type of motion should lie somewhere within the ranges investigated. An intelligent tool might ultimately be able to decide on what kind of motion would be most appropriate in any particular situation depending on the type or complexity of the ICD to be viewed or on knowledge of a particular user's preferences. Further work would however be needed to determine the way in which this might best be done. Although informal experiments with representations developed for use in the case studies and in the earlier experiments suggest that depth in some types of ICD is more successfully conveyed by some types of motion than others, it is not at present clear what attributes are relevant to the definition of an ICDs type, or what criteria should be used in an assessment of its complexity. Once such definitions are established, further investigation of the ways in which the type or complexity of an ICD affects the type of motion which should be used in viewing that ICD would be needed.

12.3.2 Layout of 3-d ICDs

3-d ICDs would ideally be laid out automatically by the kind of intelligent presentation tool envisaged. A considerable amount of work has been done on the development of heuristics for the aesthetic layout of 2-d node and link diagrams [140, 139] and Ding and Mateti [35] describe work on a system which will use formally specified heuristics to lay out 2-dimensional data structure diagrams. It seems likely that the aesthetic layout of 3-dimensional node and link diagrams or ICDs would be governed by a different, or at least larger set of heuristics than that needed for 2-dimensional diagrams. For example, the layout of a 3-d diagram might need to optimise positioning on nodes in space with respect to the amount of hiding resulting from particular arrangements: if there is too much hiding, much of the information displayed will be difficult for the user to get at; if, on the other hand, there is too little, a valuable cue to relative depth would be lost. The informal descriptions of criteria used in laying out ICDs for use as experimental stimuli could form a basis for further work on the development of a suitable set of heuristics as described in chapter 6 (section 6.5.3). The analysis of conditions which apparently lead to problems with the perception of relative depth described in section 6.3.6 is a further source of material on which such work could be based as the conditions identified as problematic in the context of the experiments might in general need to be avoided, or at least compensated for (by artificially increasing depth) in any automatically produced layouts.

Apart from providing an aesthetic layout, an intelligent system would ideally be able to make meaningful use of spatial coding in ways such as those suggested in the case studies. A formal specification of a particular 3-d ICD language such as that described in chapter 11 includes a definition of what Fairchild *et al.* refer to as 'mapping functions' [52] which could be used to compute the relative positions of representations of entities related in a particular way. Mapping functions suitable for use in displaying a particular structure could in the first instance be chosen by the visual language designer, or the user as suggested in chapter 11. (Note that in this case, users would need a suitable means of specifying what functions were to be used.) Ultimately, though, an intelligent tool might take over the task of 3-d ICD language design, just as Mackinlay's APT was able to take on the design of 2-dimensional displays of relational data [87]. Note, however, that as described above, the question of how heuristics for aesthetic layout (or

'soft constraints') should be integrated with or traded off against such 'hard constraints' or mapping functions is a subject for further research.

12.3.3 Further Functionality

A number of detailed recommendations regarding the functions which could be provided for manipulation of a 3-d ICD and for the placement of individual components within a 3-d ICD were made in chapter 9. Fairchild *et al.* [52] discuss of a number of approaches to some of these problems and also consider the provision of functions to support browsing and navigation through 3-d node and link structures. These suggestions and recommendations could be taken into account in the development of a tool of the kind described (though it should be noted that not all of the suggestions of Fairchild *et al.* may be appropriate in the development of tools supporting the use of 3-d ICDs of the kind considered here).

The kinds of user functions provided could also be specialised to meet the needs of particular tasks identified in chapter 8 as being those in which 3-d ICDs would be particularly useful. Extension of the kind of formal specification described in chapter11 would facilitate the specification and provision of such specialised facilities, though further work would be needed to determine what kinds of functions might best be provided to support particular kinds of task.

Finally, although the above discussion has concentrated mainly on aspects of the user interface, the development of an intelligent tool of the kind described would, of course, require a great deal of work on other aspects of the system. One of the major problems to be solved concerns the way in which both system and user should be allowed to manipulate 3-d ICD representations of a knowledge structure undergoing continual change. This problem is the subject of ongoing research at City.

12.4 Conclusions

This has been a fruitful area of research. 3-d ICDs have considerable potential as a medium in which to represent knowledge structures for use in knowledge engineering. It is therefore contended that the next generation of integrated tools for knowledge engineering could benefit considerably from the inclusion of one or more 3-d ICD editors or presentation tools tailored specifically for use at various points in the knowledge-based system development lifecycle. However, a considerable amount of work remains to be done before the goal of developing such integrated tools can sensibly be achieved. The discussion above has highlighted a number of areas in which such work is needed.

Finally, it should be noted that the potential utility of 3-d ICDs as a medium for use in the human-computer interface is not restricted to the domain of knowledge engineering, or indeed to that of knowledge-based systems. Work in the field of databases [110] as well as that on the Information Visualiser [116] suggests that tools incorporating a 3-d ICD component might be extremely useful in a number of other areas. Once again, further work is needed to investigate these possibilities.

12.5 Summary

This chapter has evaluated the work reported earlier, and the significance of the claims made as a result of that work. The evaluation has considered both the success of the work itself (in terms of the general significance of its findings and its applicability in related domains) and its relation to other work with similar aims. Directions for possible further research were briefly discussed, and some overall conclusions were presented.

Appendices

Appendix A

Pilot Experiment: Materials and Results

A.1 Stimuli

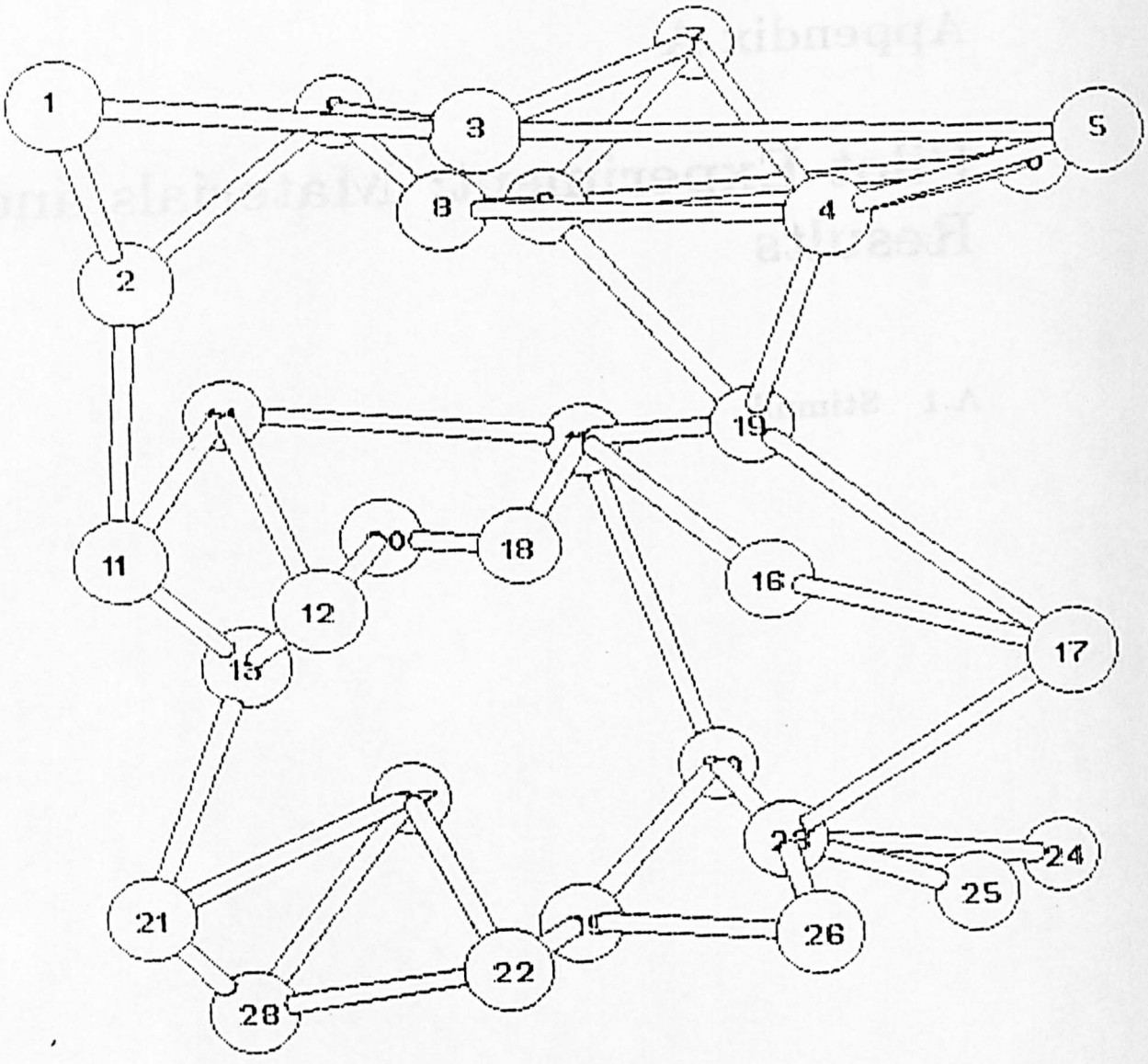


Figure A.1: Pilot Experiment: Stimulus 1

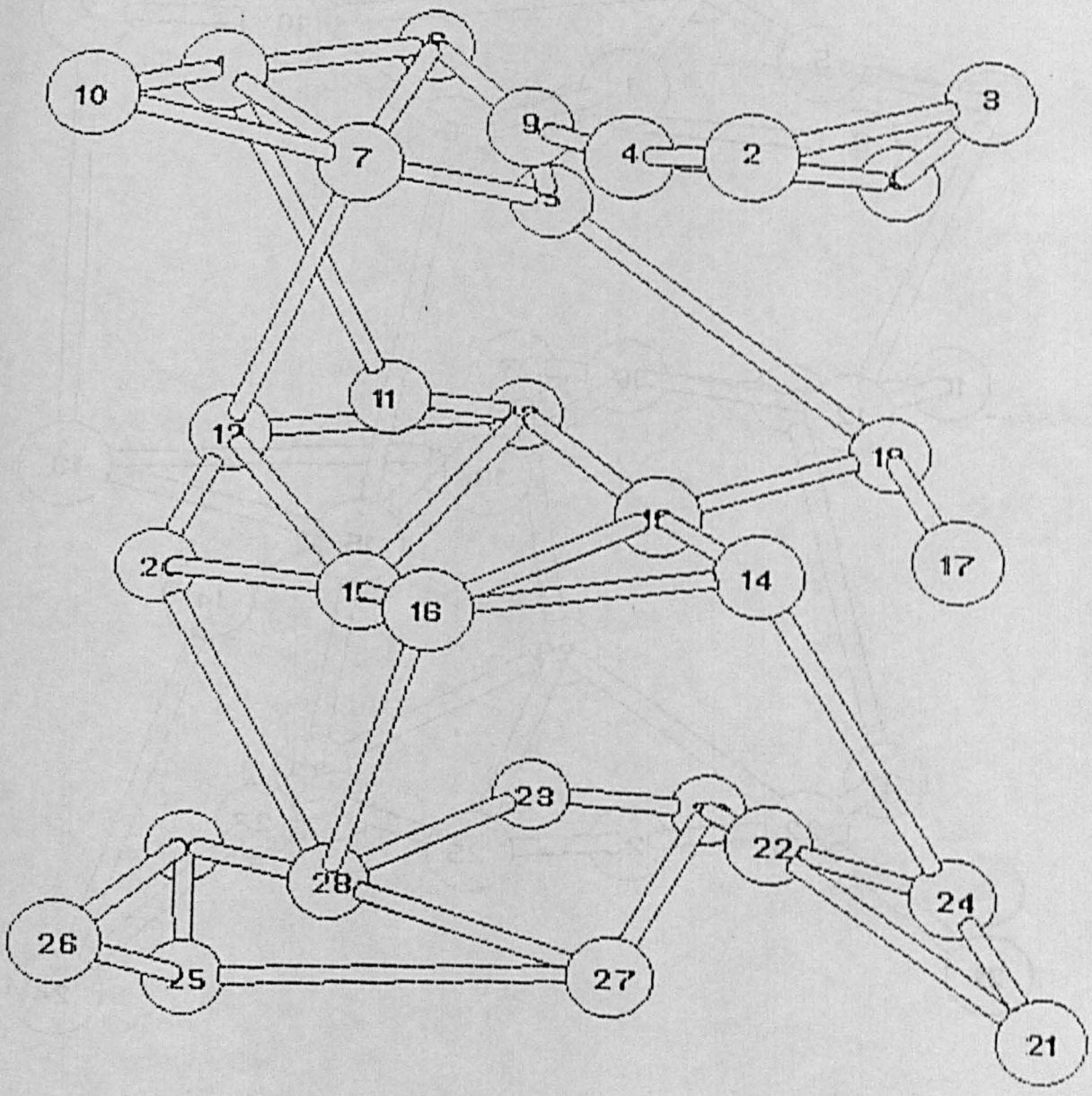


Figure A.2: Pilot Experiment: Stimulus 2

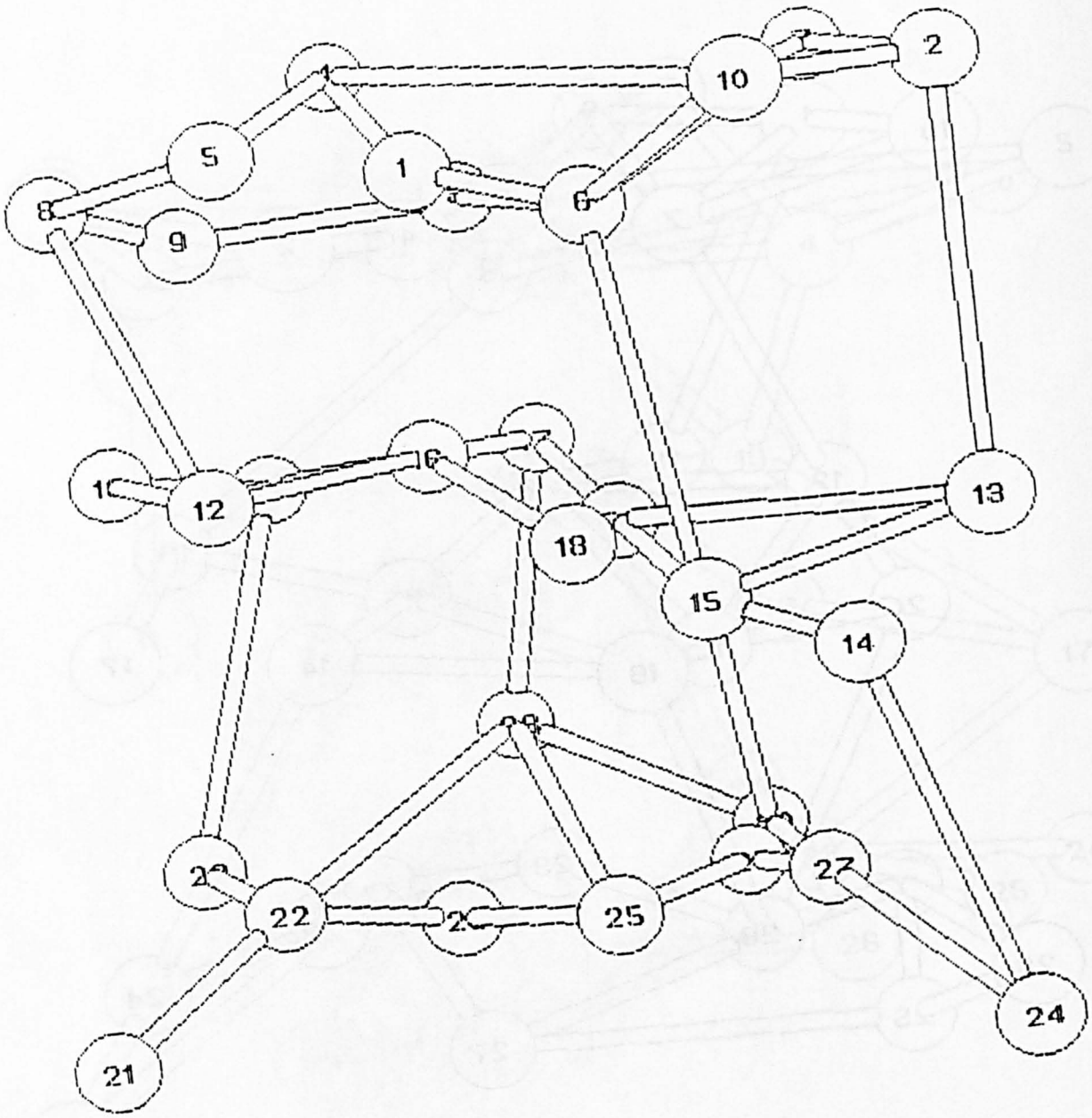


Figure A.3: Pilot Experiment: Stimulus 3

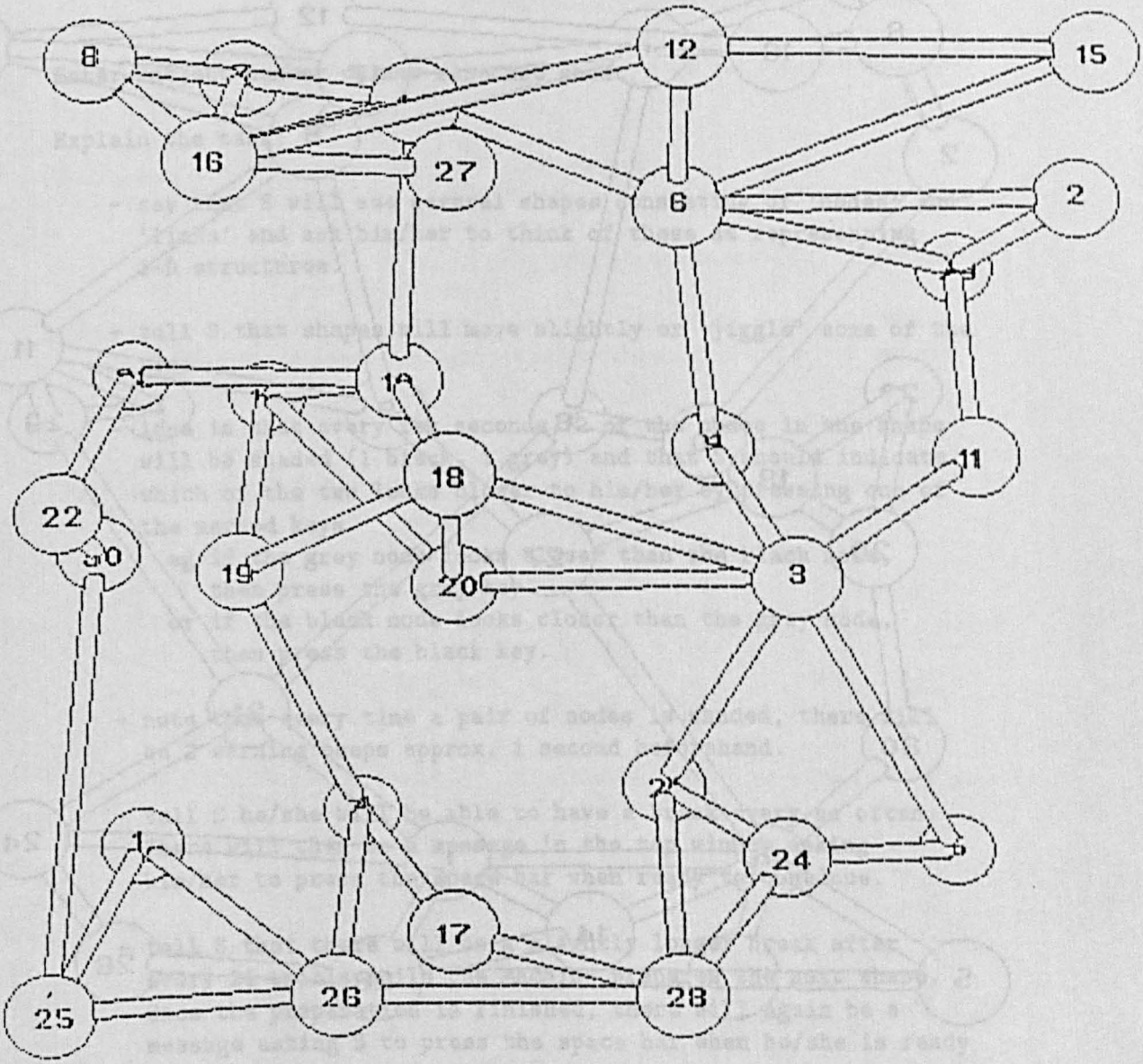


Figure A.4: Pilot Experiment: Stimulus 4

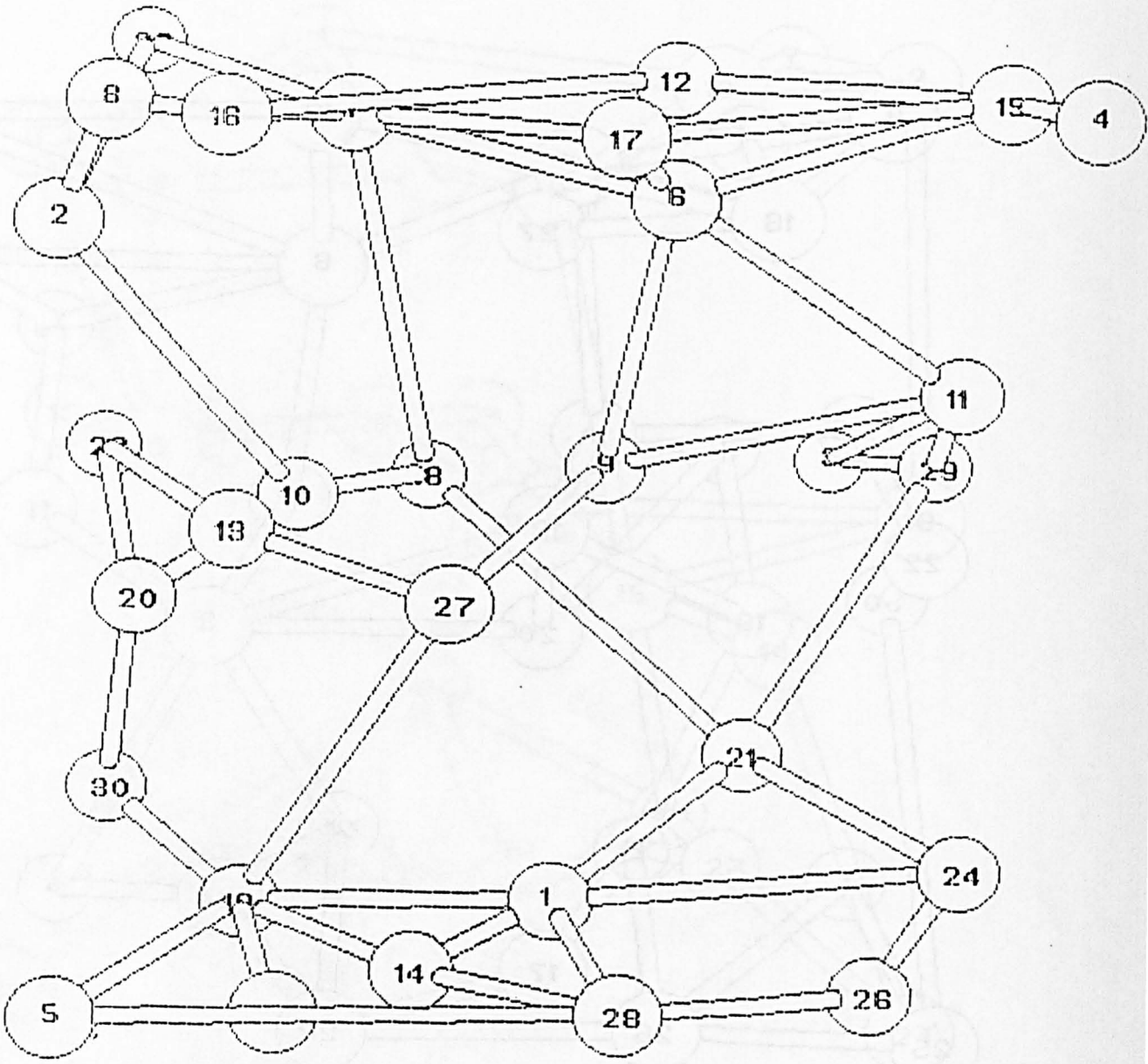


Figure A.5: Pilot Experiment: Stimulus 5

A.2 Protocol

For each subject:

Enter subject number on new response sheet.

Explain the task:

- say that S will see several shapes consisting of 'nodes' and 'links' and ask him/her to think of these as representing 3-D structures.

- tell S that shapes will move slightly or 'jiggle' some of the time.

- idea is that every few seconds, 2 of the nodes in the shape will be shaded (1 black, 1 grey) and that S should indicate which of the two looks closer to him/her by pressing one of the marked keys
  eg if the grey node looks closer than the black node, then press the grey key
  or if the black node looks closer than the grey node, then press the black key.

- note that every time a pair of nodes is shaded, there will be 2 warning beeps approx. 1 second beforehand.

- tell S he/she will be able to have a break every so often. There will then be a message in the top window asking him/her to press the space bar when ready to continue.

- tell S that there will be a slightly longer break after every 24 trials while the machine prepares the next shape. Once the preparation is finished, there will again be a message asking S to press the space bar when he/she is ready to continue.

- ask S always to respond reasonably accurately but to be as quick as he/she can. Ask S to guess which is closer if he/she really can't decide.

Enter subject number on the computer.

Change labels on keys if necessary.

Explain that first 24 trials are for practice and ask S to try them and ask any questions he/she likes during or after that block.

---------- S does first block ----------

Check that S understands everything.

Remind S to be as quick and accurate as possible.

---------- S does main part of experiment ----------

Ask S to fill in the response sheet.

Ask S if they would like a copy of their results. If so, use shell window to print out results file. Explain results if necessary.

Thank S for participating.

A.3 Questionnaire

SUBJECT RESPONSE SHEET

Subject no.: Age: Sex: M/F

1) How easy did you find the task when the shape was still ?

Very difficult Very easy

v v

1 2 3 4 5 6 7

2) How easy did you find the task when the shape was moving ?

Very difficult Very easy

v v

1 2 3 4 5 6 7

3) How confident were you of your accuracy when the shape was still ?

Very unconfident Very confident

v v

1 2 3 4 5 6 7

4) How confident were you of your accuracy when the shape was moving ?

Very unconfident Very confident

v v

1 2 3 4 5 6 7

5) How comfortable were you with the movement of the shape ?

Very uncomfortable Very comfortable

v v

1 2 3 4 5 6 7

6) Do you think the task with the moving shape would be easiest if the shape moved

Much slower Same speed Much faster

v v v

1 2 3 4 5 6 7

7) Do you think the task with the moving shape would be easiest if the shape moved

Much less distance Same distance Much greater distance

v v v

1 2 3 4 5 6 7

8) Can you suggest any other changes which would make it easier to work with the moving shape ?

9) Is there anything in particular that strikes you about the difference between working with a still shape and working with a moving shape ?

A.4 Results

The mean logged response times and ratings of perceived ease plotted in chapter 5. The remaining data relating to accuracy, confidence, comfort, preferred speed of motion and preferred angle of motion are tabulated below.

S = still shape
M = moving shape

	ERRORS		CONFIDENCE		COMFORT	SPEED	ANGLE
SUBJECT	S	M	S	M			
1	6	3	3	5	4	4	3
2	5	3	3	4	5	2	5
3	6	6	3	5	5	4	5
4	4	4	5	6	6	4	4
5	3	5	3	5	5	4	4
6	4	2	4	5	5	4	3
7	4	2	2	3	5	5	6
8	5	5	5	6	7	4	5
9	12	11	6	6	7	4	3
10	5	5	4	5	6	4	5
11	4	3	3	2	1	5	7
12	8	7	4	6	7	4	5

Appendix B

Further Experiments: Materials and Results

B.1 Stimuli

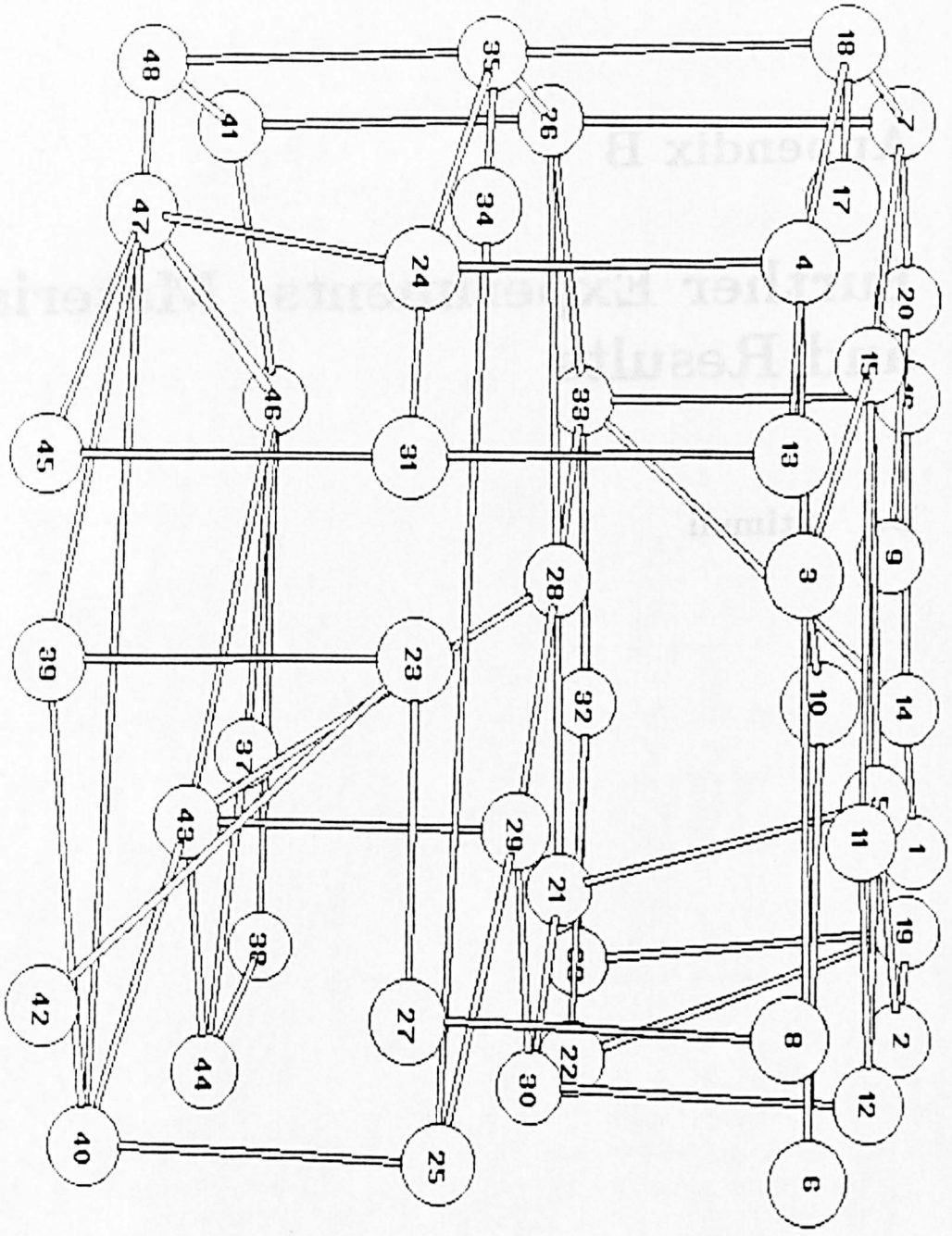


Figure B.1: Type I Stimuli: Layered (1)

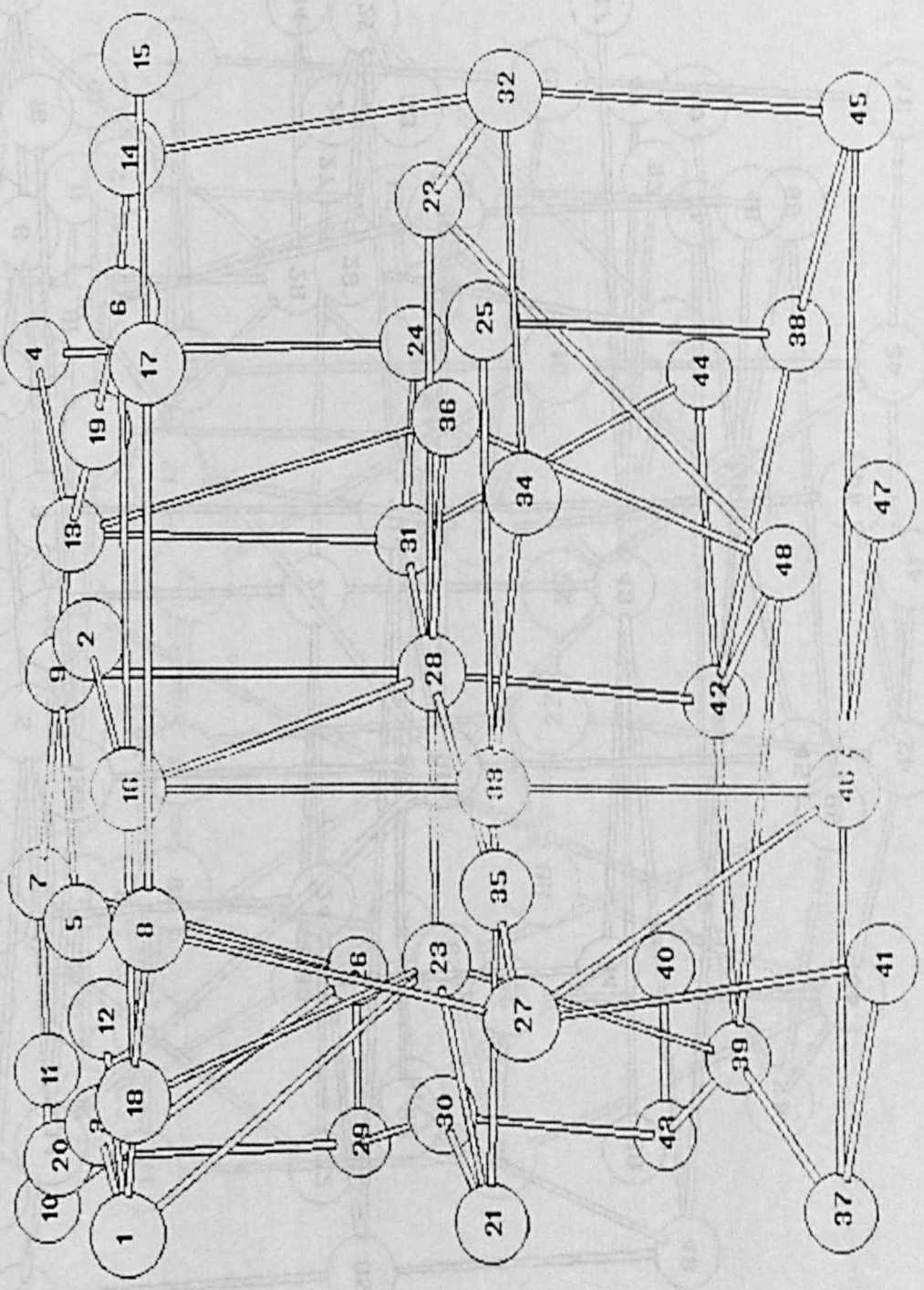


Figure B.2: Type I Stimuli: Layered (2)

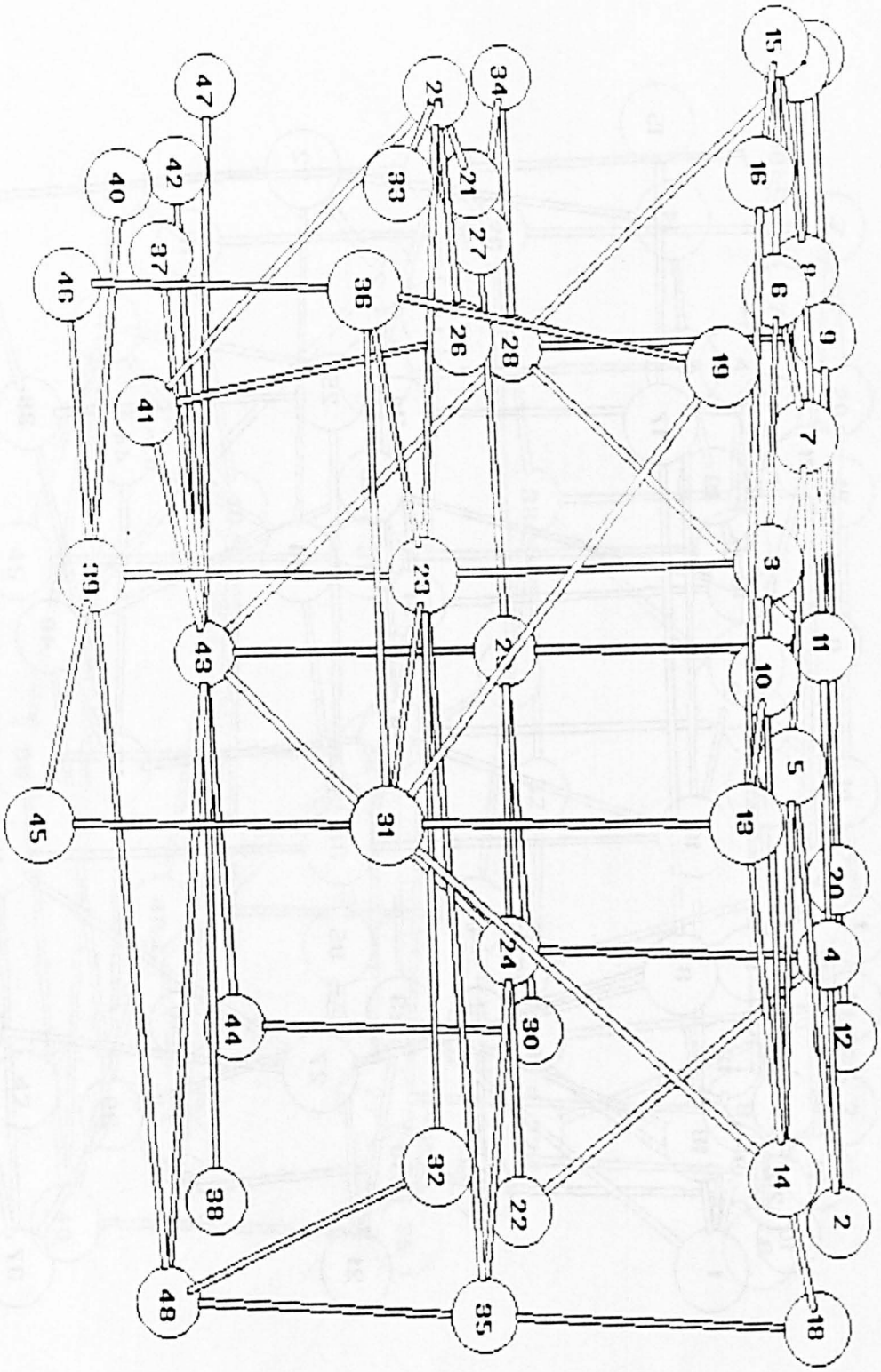


Figure B.3: Type I Stimuli: Layered (3)

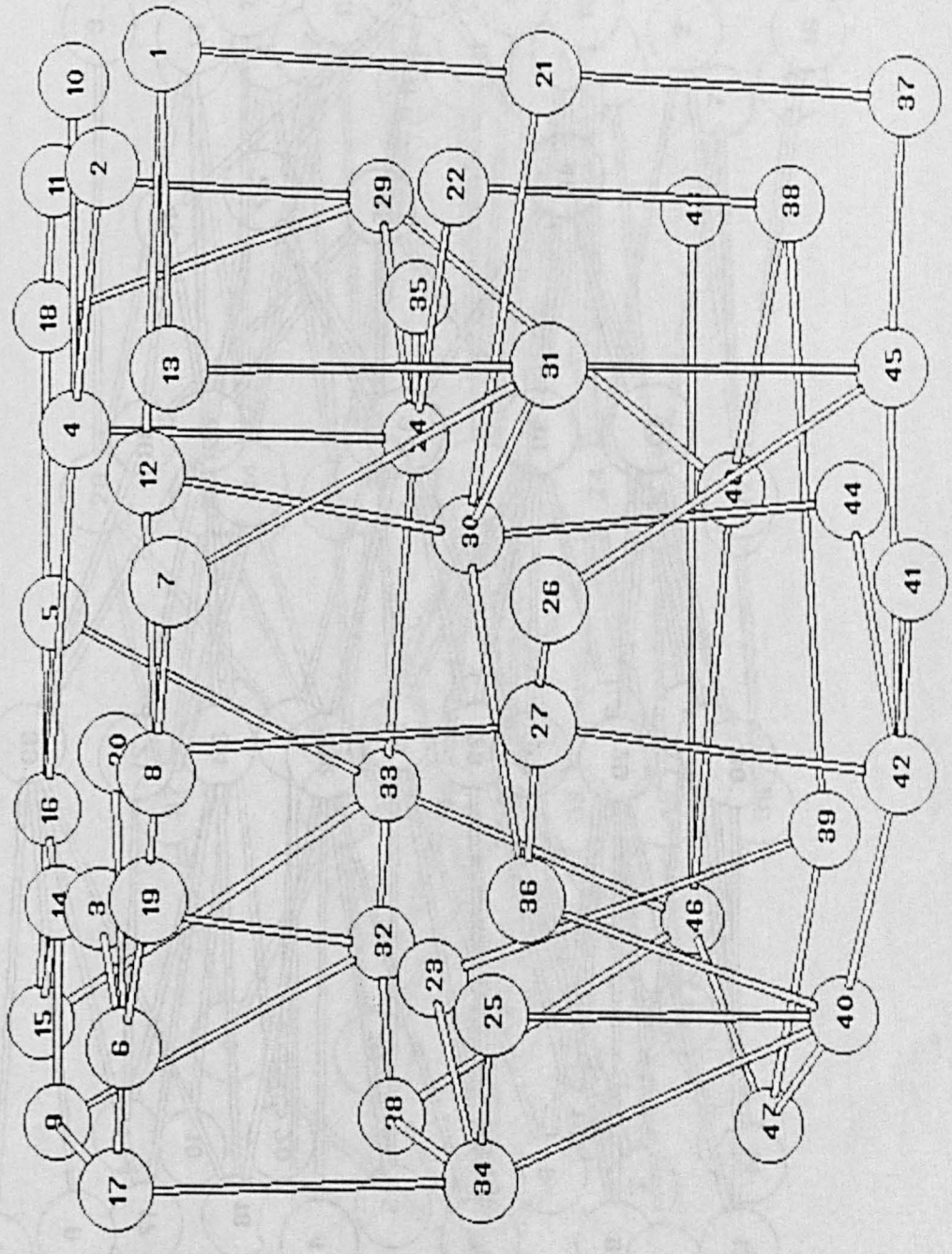


Figure B.4: Type I Stimuli: Layered (4)

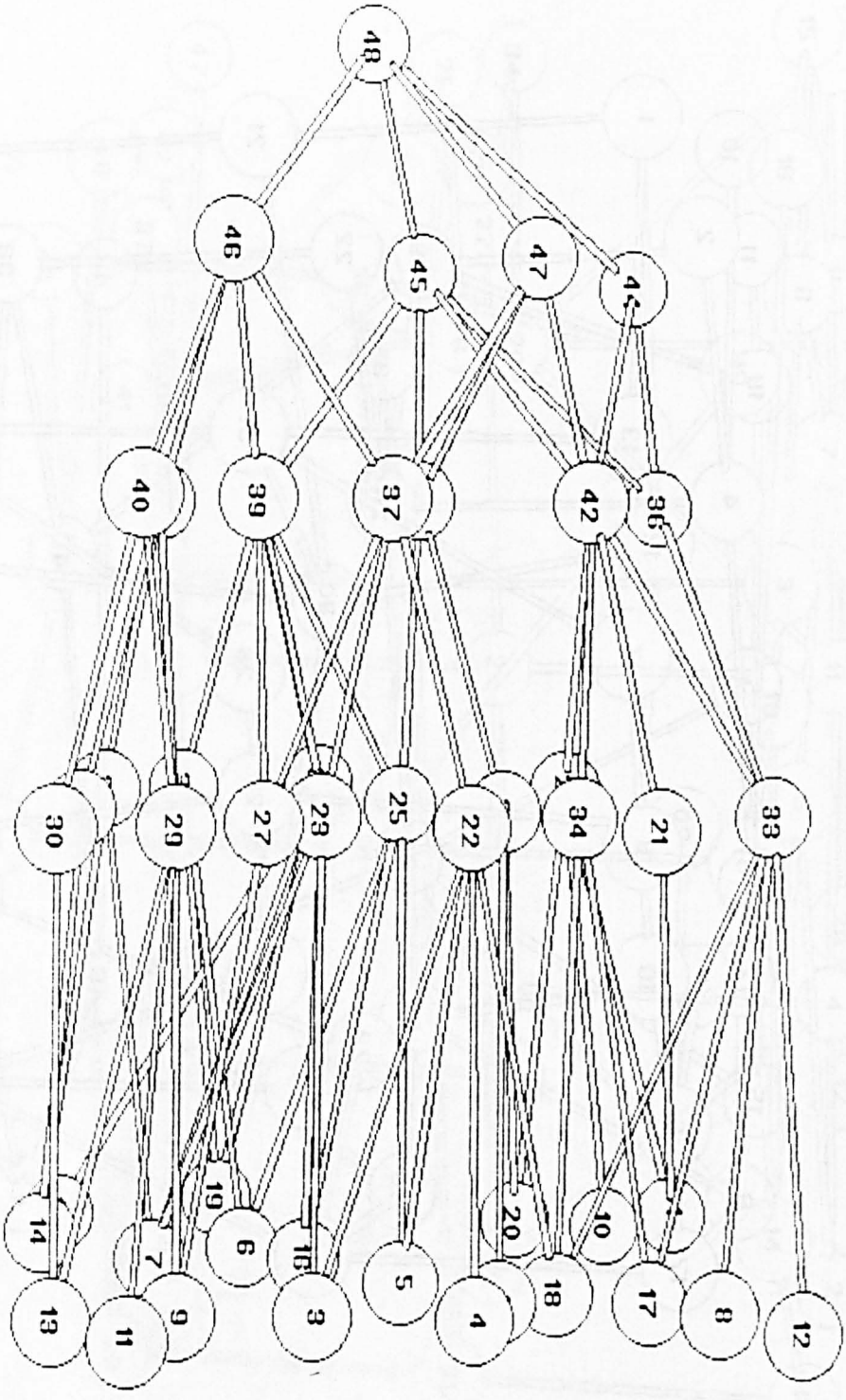


Figure B.5: Type II Stimuli: Hierarchic (1)

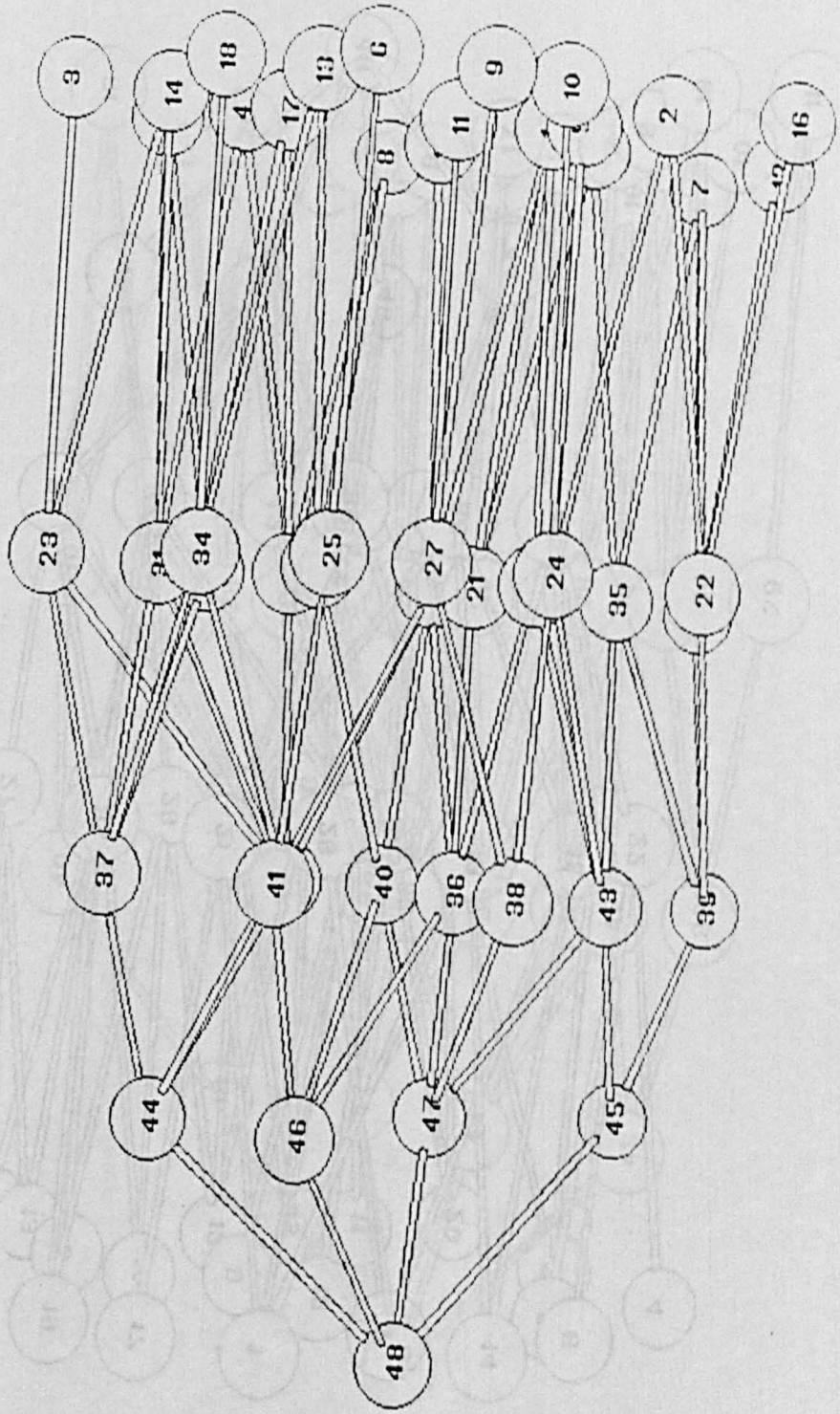


Figure B.6: Type II Stimuli: Hierarchic (2)

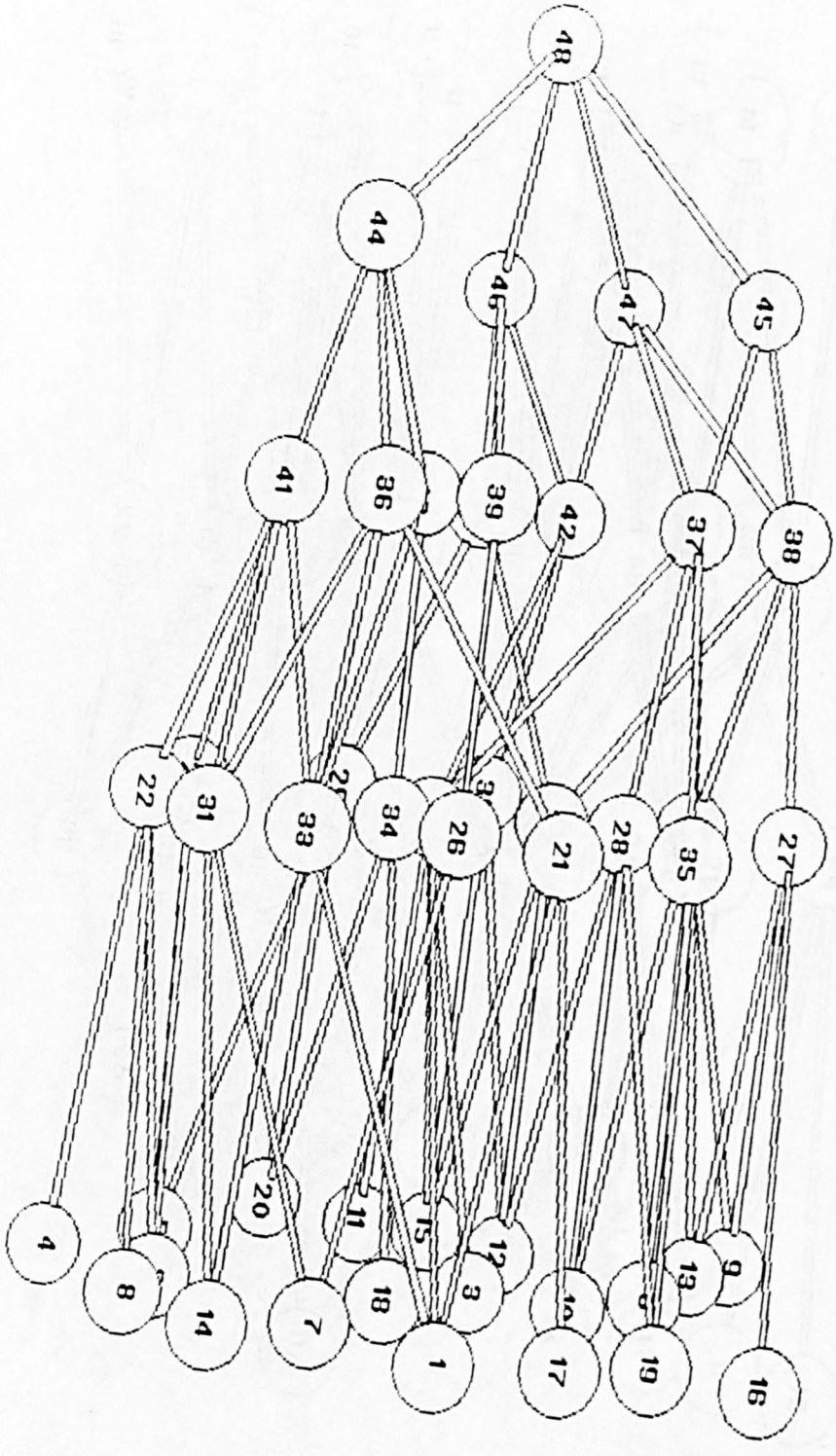


Figure B.7: Type II Stimuli: Hierarchic (3)

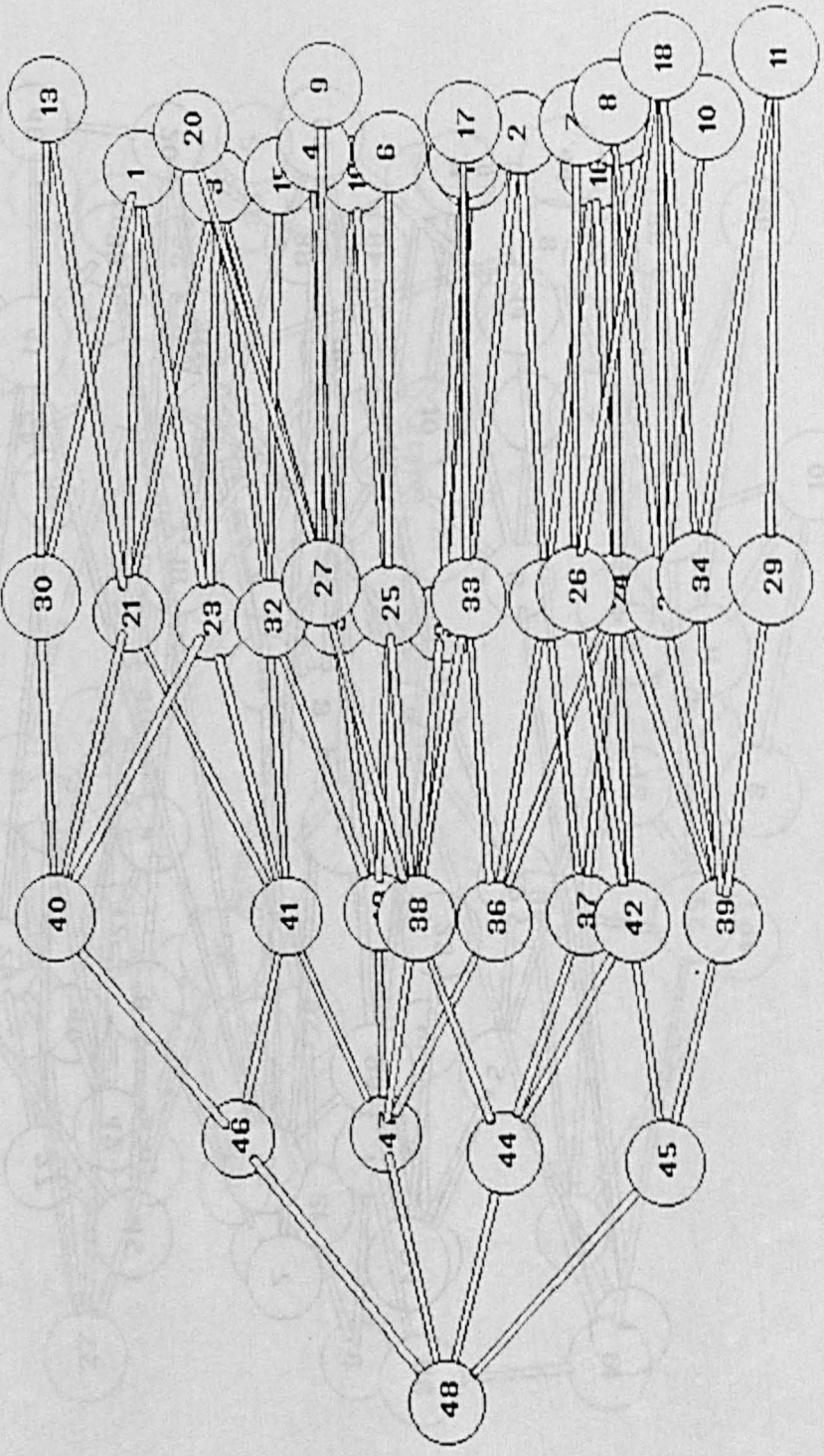


Figure B.8: Type II Stimuli: Hierarchic (4)

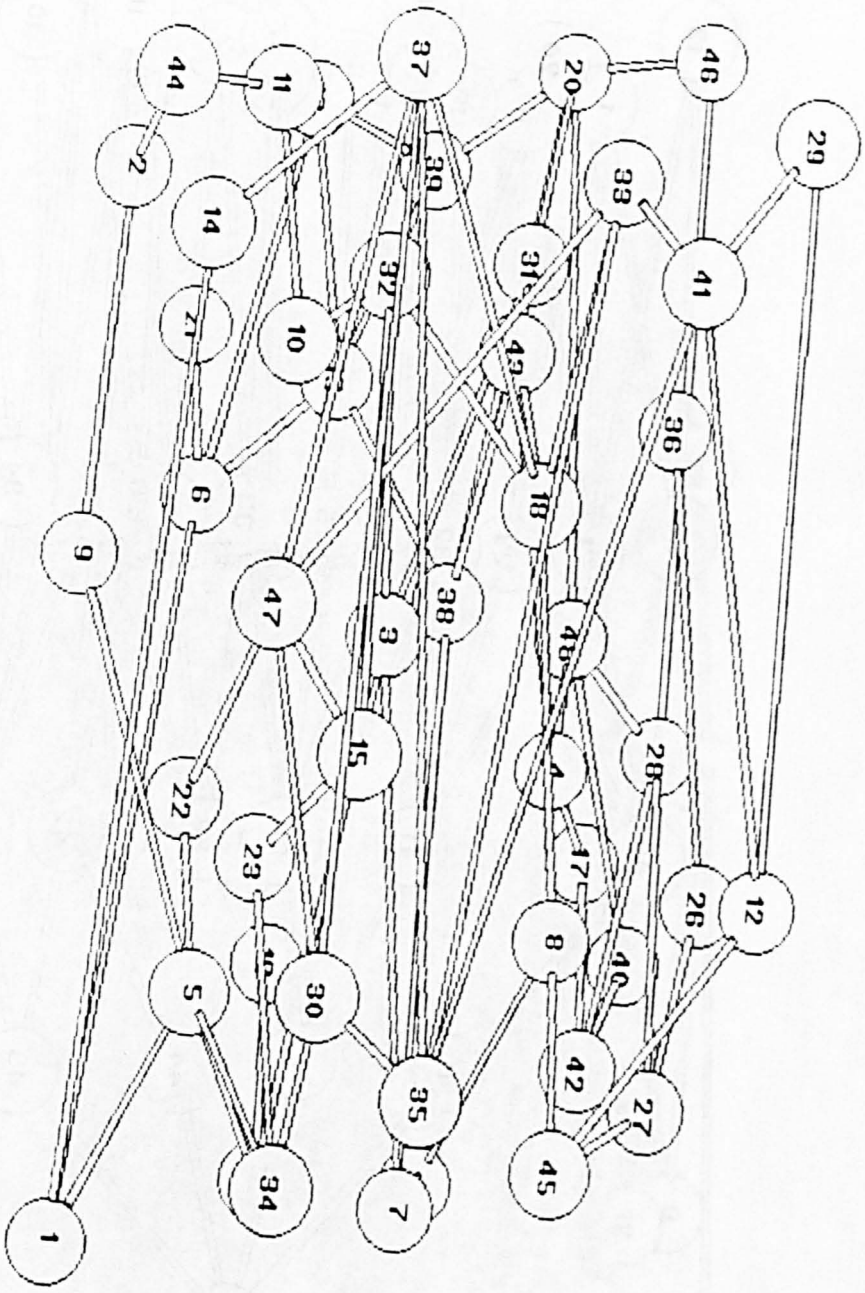


Figure B.9: Type III Stimuli: Random (1)

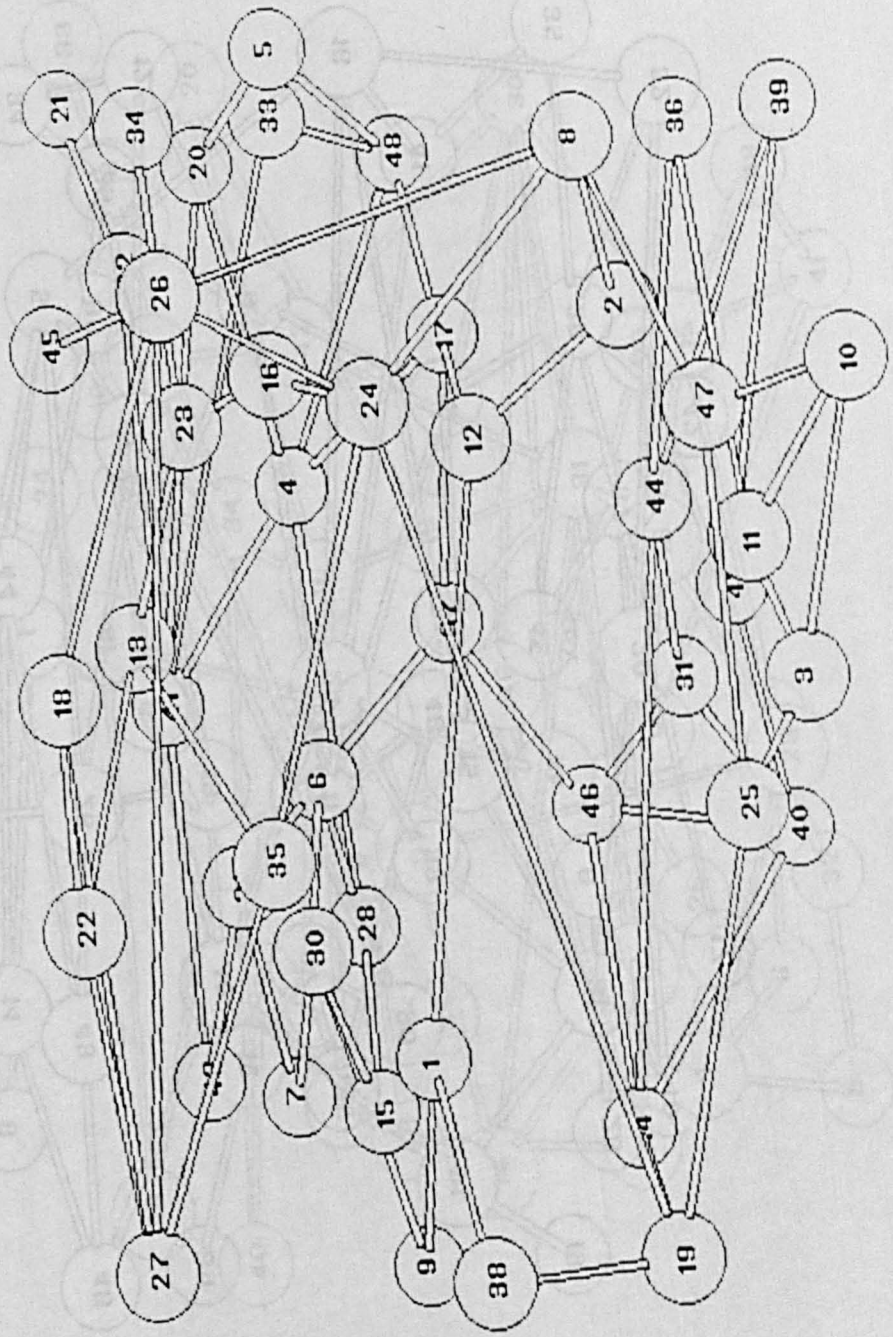


Figure B.10: Type III Stimuli: Random (2)

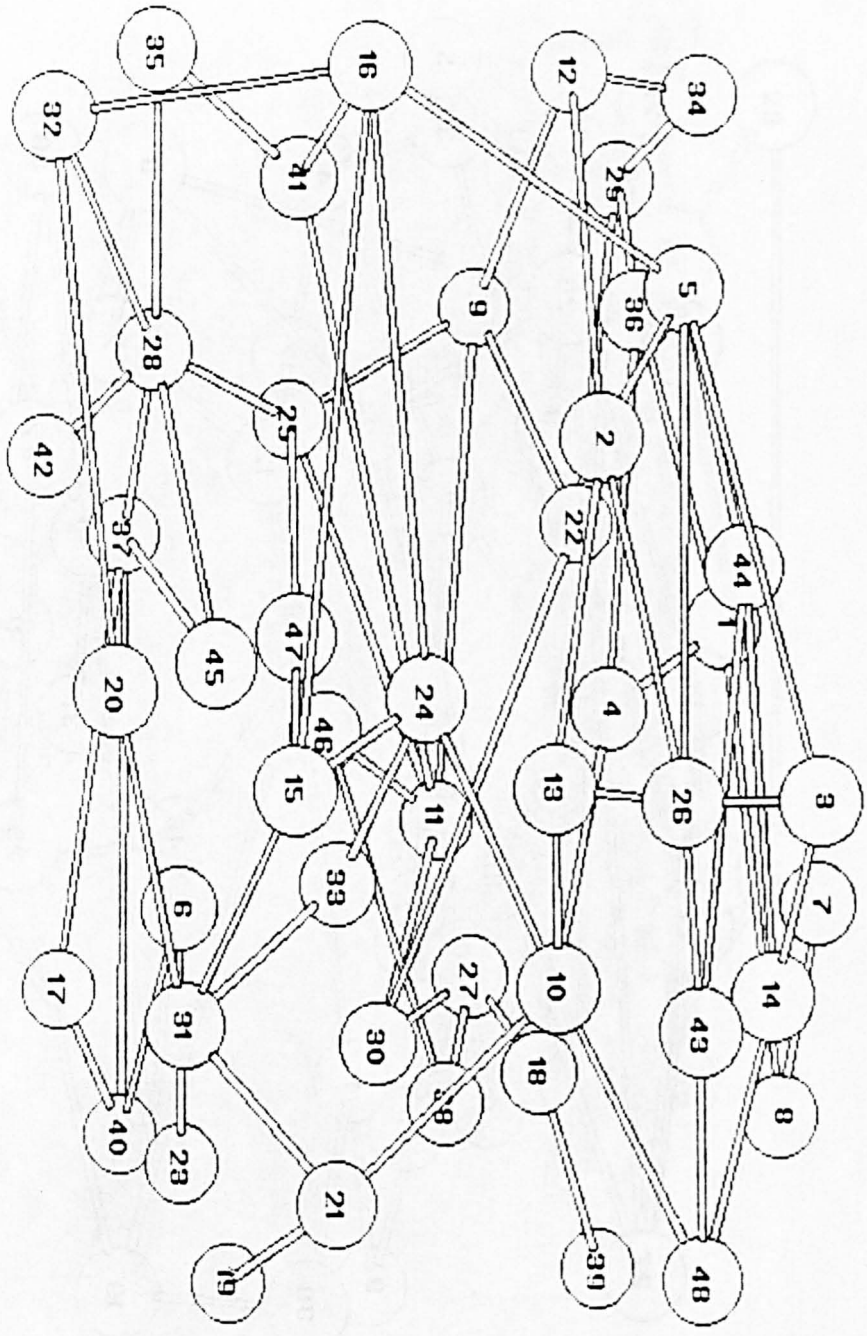


Figure B.11: Type III Stimuli: Random (3)

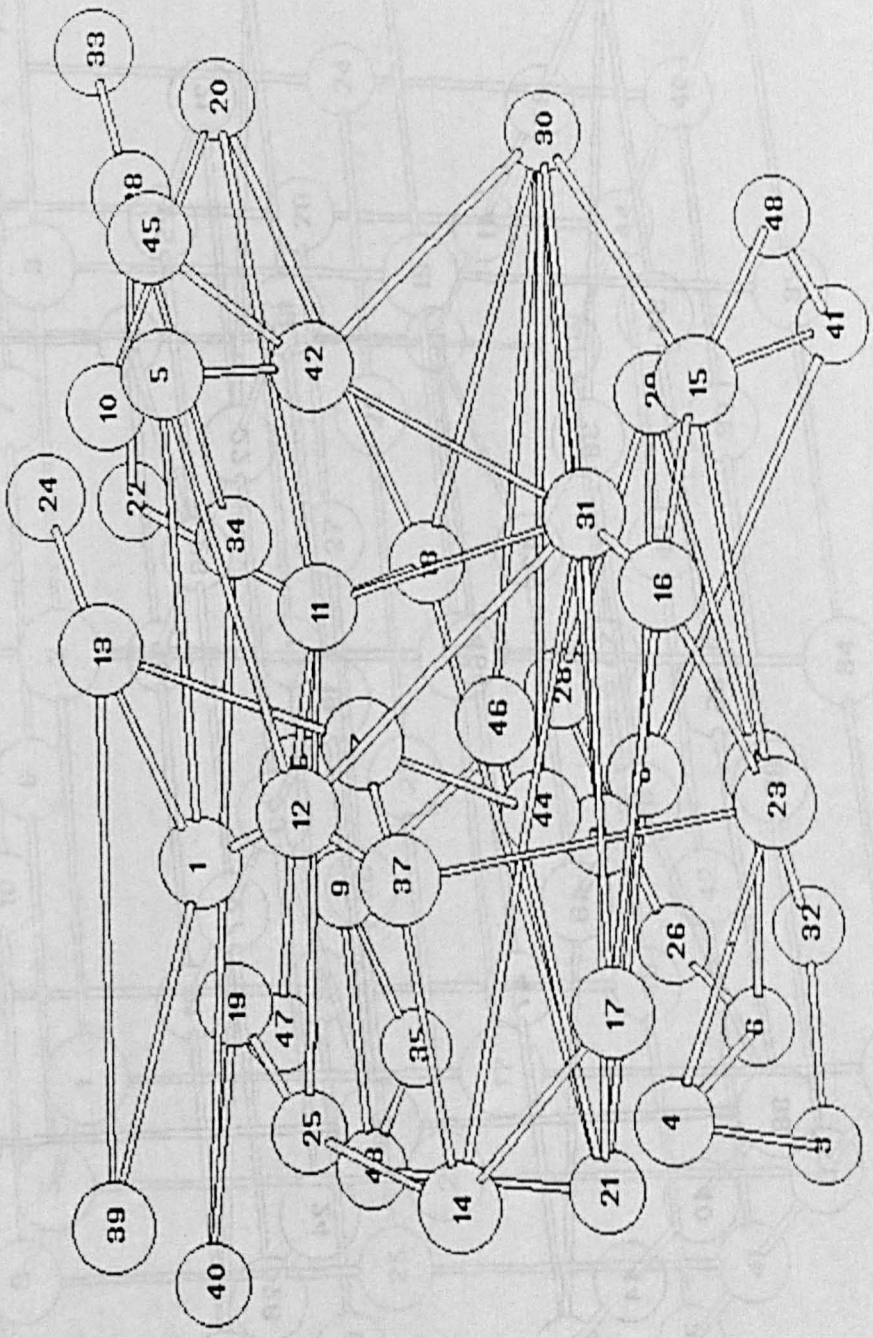


Figure B.12: Type III Stimuli: Random (4)

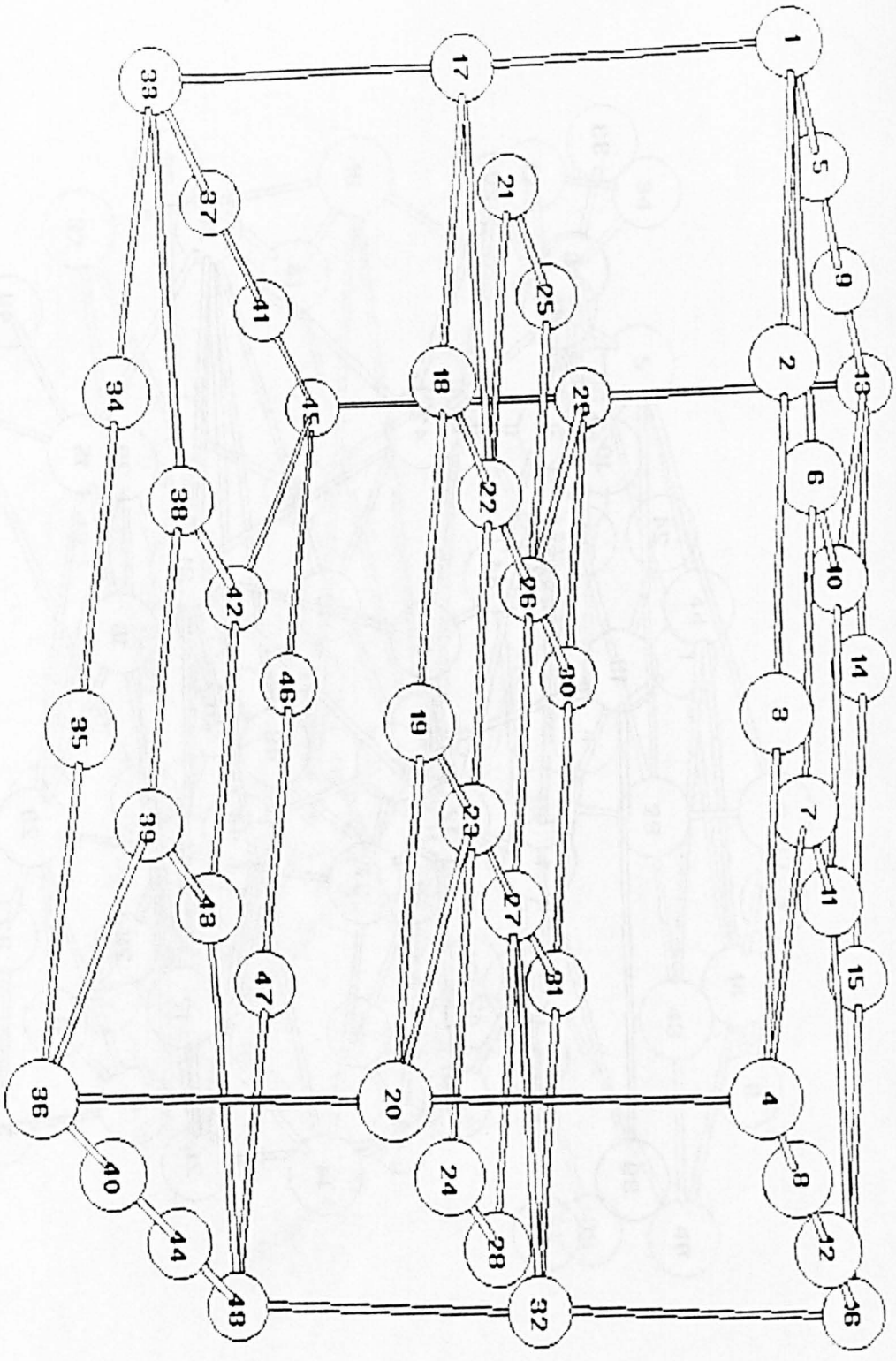


Figure B.13: Type IV Stimuli: Regular (1)

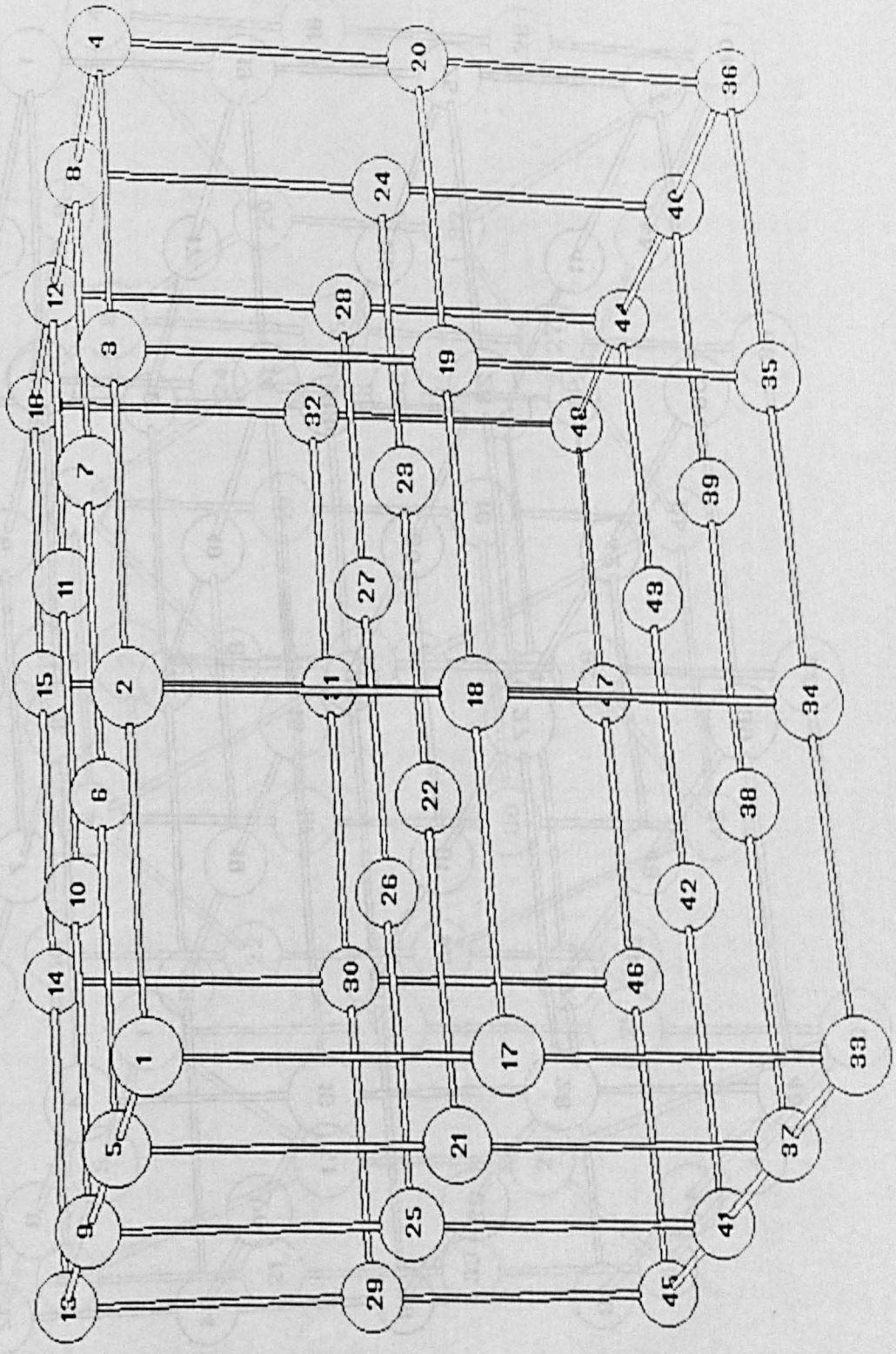


Figure B.14: Type IV Stimuli: Regular (2)

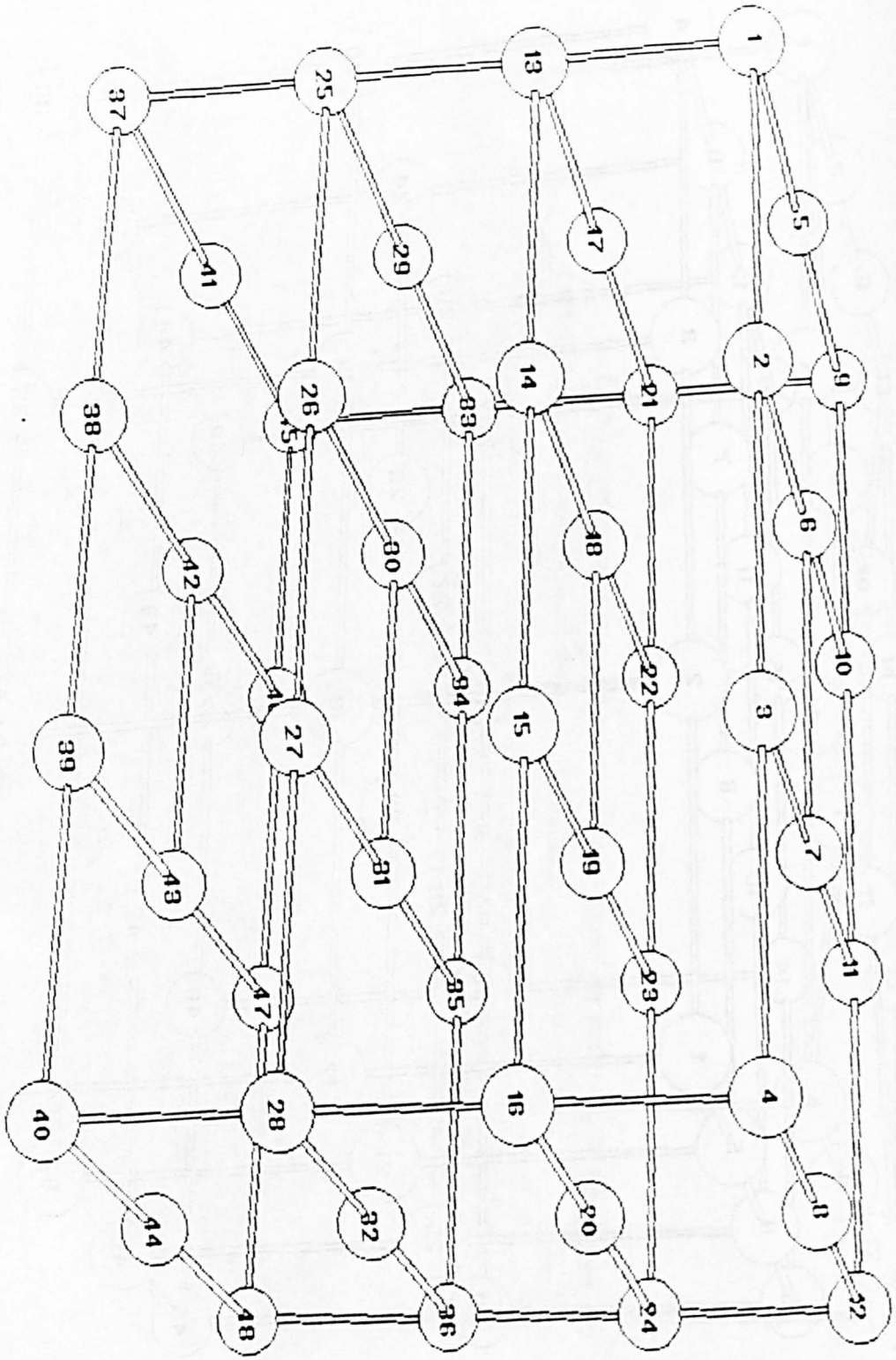


Figure B.15: Type IV Stimuli: Regular (3)

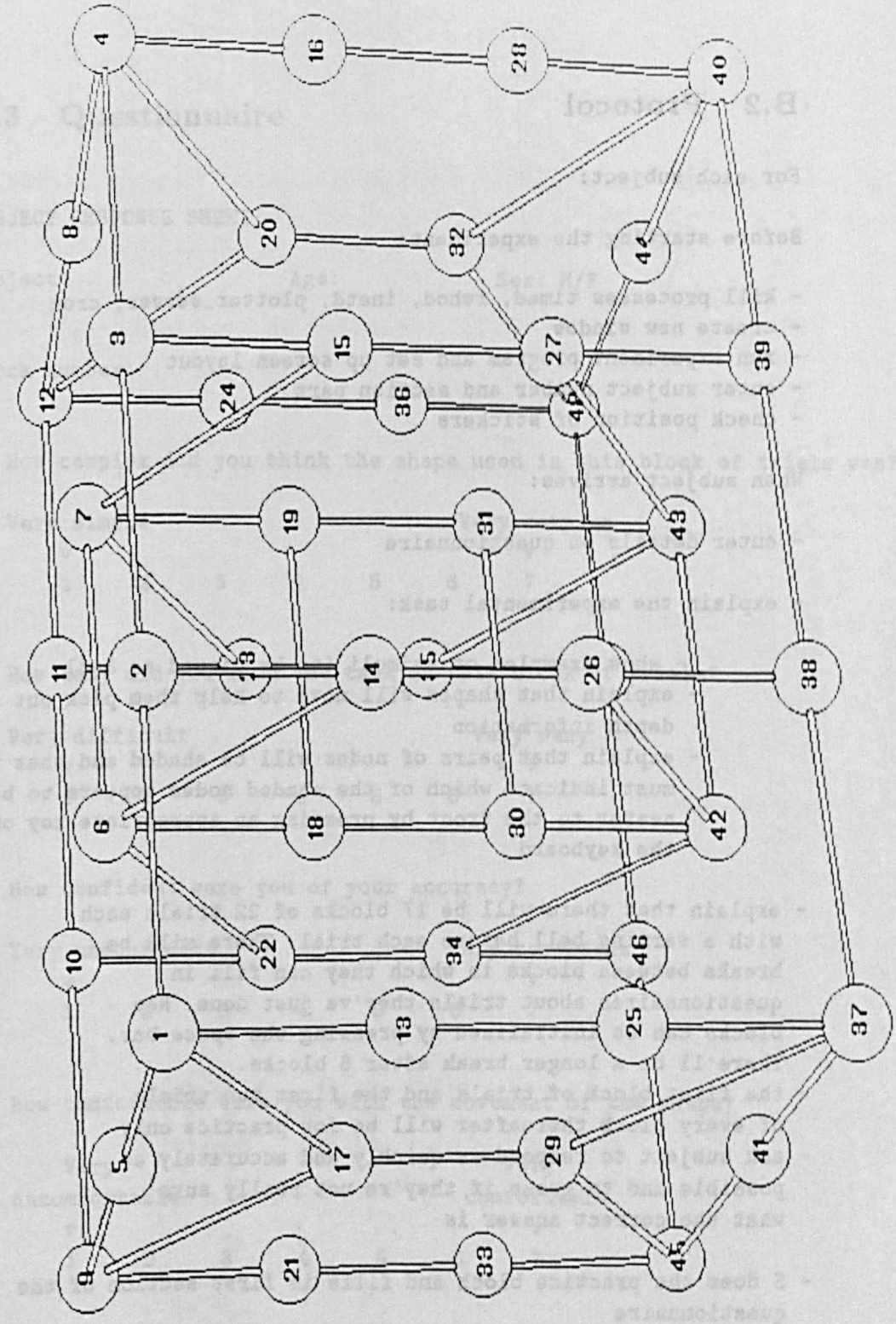


Figure B.16: Type IV Stimuli: Regular (4)

B.2 Protocol

For each subject:

Before starting the experiment:

- kill processes timed, rwhod, inetd, plotter_server, cron
- create new window
- run experiment program and set up screen layout
- enter subject number and section part
- check position of stickers

When subject arrives:

- enter details on questionnaire

- explain the experimental task:

    - show examples of stimuli (to be viewed as 3-d)
    - explain that shapes will move to help them pick out depth information
    - explain that pairs of nodes will be shaded and that they must indicate which of the shaded nodes appears to be nearer to the front by pressing an appropriate key on the keyboard

- explain that there will be 17 blocks of 22 trials each with a warning bell before each trial. There will be breaks between blocks in which they can fill in questionnaires about trials they've just done. New blocks can be initialised by pressing the space bar. There'll be a longer break after 8 blocks.
- the first block of trials and the first two trials of every block thereafter will be for practice only
- ask subject to respond as quickly and accurately as possible and to guess if they're not really sure what the correct answer is

- S does the practice block and fills in first section of the questionnaire
- check S understands, remind S to be as quick and accurate as possible
- S does first half of the experiment

- break if S wishes
- S does second half of the experiment
- ask S if they want a copy of their results file
- thank S for taking part

B.3 Questionnaire

SUBJECT RESPONSE SHEET

Subject: Age: Sex: M/F

Block number:

1) How complex did you think the shape used in this block of trials was?

Very simple Very complex

v v

1 2 3 4 5 6 7

2) How easy did you find the task in this block of trials?

Very difficult Very easy

v v

1 2 3 4 5 6 7

3) How confident were you of your accuracy?

Very unconfident Very confident

v v

1 2 3 4 5 6 7

4) How comfortable were you with the movement of the shape?

Very Very

uncomfortable comfortable

v v

1 2 3 4 5 6 7

5) Please note down any comments you have about the shape and the way it moved:

B.4 Results

B.4.1 Effect of Angle of Motion

B.4.1.1 Response Time Analysis

The results of carrying out an analysis of variance on logged response times from experiment 1 with subject (SUBJECT) as a random effect and angle of motion (JIGANG) as a fixed effect is shown below.

General Linear Models Procedure
Tests of Hypotheses for Mixed Model Analysis of Variance

Dependent Variable: RESPTIME

Source: JIGANG
Error: MS(JIGANG*SUBJECT)

DF	Type III MS	Denominator DF	Denominator MS	F Value	Pr > F
3	0.4397827645	33	0.2662541148	1.652	0.1964

Source: SUBJECT
Error: 0.9999*MS(JIGANG*SUBJECT) + 0.0001*MS(Error)

DF	Type III MS	Denominator DF	Denominator MS	F Value	Pr > F
11	27.404517651	33.00	0.2662447237	102.930	0.0001

Source: JIGANG*SUBJECT
Error: MS(Error)

DF	Type III MS	Denominator DF	Denominator MS	F Value	Pr > F
33	0.2662541148	3756	0.1385738637	1.921	0.0012

B.4.1.2 Data

The following tables show values of mean logged response times, total errors and median ease, confidence and comfort ratings for every subject by angle of motion. Values are ranked for each subject and total rankings for all subjects are shown.

	Angle of Rocking Motion			
	0.5	0.79	1.26	2
	MEAN	MEAN	MEAN	MEAN
	Log Response Time	Log Response Time	Log Response Time	Log Response Time
SUBJECT				
1	0.42	0.30	0.32	* 0.27
2	0.08	0.19	0.15	* 0.06
3	-0.32	* -0.44	-0.39	-0.43
4	0.55	0.55	* 0.52	0.57
5	-0.13	-0.24	-0.28	* -0.31
6	0.19	0.07	* 0.06	0.20
7	-0.03	-0.01	* -0.07	-0.05
8	* 0.45	0.50	0.60	0.52
9	0.36	0.35	* 0.28	0.36
10	* 0.20	0.25	0.21	0.30
11	-0.04	* -0.10	-0.09	0.02
12	0.51	0.37	* 0.31	0.36

* : subject minimum

Table B.1: Mean Logged Response Times by Angle of Motion: Experiment 1

	Angle of Rocking Motion							
	0.5		0.79		1.26		2	
	Total Errors		Total Errors		Total Errors		Total Errors	
SUBJECT								
1	2.5	9	2.5	9	4	15	1	7
2	1	6	3	11	4	12	2	9
3	1	15	4	19	3	18	2	16
4	2.5	14	1	9	4	19	2.5	14
5	2.5	11	2.5	11	4	18	1	6
6	2	13	1	5	3	14	4	15
7	2.5	10	2.5	10	4	15	1	9
8	2.5	7	2.5	7	1	3	4	8
9	1	9	2	10	4	20	3	11
10	3	12	2	10	4	16	1	8
11	1	10	3.5	14	3.5	14	2	11
12	2	5	1	4	4	11	3	6
Sum of ranks	23.5	(1)	27.5	(3)	42.5	(4)	26.5	(2)

Figures on the left in cells in the body of the table represent rank of total error rates for each subject:
1: least errors
4: most errors

A Friedman test showed the difference in rankings overall to be significant at p = 0.05. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.2: Total Errors by Angle of Motion: Experiment 1

	Angle of Rocking Motion							
	0.5		0.79		1.26		2	
	MEDIAN		MEDIAN		MEDIAN		MEDIAN	
	Ease Rating		Ease Rating		Ease Rating		Ease Rating	
SUBJECT								
1	2	5.5	3	5	1	6	4	3.5
2	3.5	4.5	1.5	5	1.5	5	3.5	4.5
3	2	5	2	5	2	5	4	3
4	2	4.5	3.5	4	1	6.5	3.5	4
5	2.5	4.5	1	5	2.5	4.5	4	3
6	3	4	3	4	1	6	3	4
7	3	4.5	4	4	1	6	2	5
8	2.5	5	2.5	5	1	5.5	4	3.5
9	2.5	5	4	4.5	1	5.5	2.5	5
10	2.5	4	4	2.5	1	5	2.5	4
11	1.5	5.5	1.5	5.5	3	4.5	4	4
12	1.5	5	3.5	4.5	1	6	3.5	4.5
Sum of ranks	29	(2)	33.5	(3)	17	(1)	40.5	(4)

Figures on the left in cells in the body of the table represent rank of median ease ratings for each subject:
1: rated easiest
4: rated least easy

A Friedman test showed the difference in rankings overall to be significant at $p = 0.005$. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.3: Median Ease Ratings by Angle of Motion: Experiment 1

	Angle of Rocking Motion							
	0.5		0.79		1.26		2	
	MEDIAN Confidence Rating		MEDIAN Confidence Rating		MEDIAN Confidence Rating		MEDIAN Confidence Rating	
SUBJECT								
1	2	5.5	3	5	1	6	4	4
2	2.5	5	2.5	5	2.5	5	2.5	5
3	1	5	2	4.5	3	4	4	3
4	3	4	3	4	1	6.5	3	4
5	2.5	4.5	2.5	4.5	1	5	4	3
6	3	6	3	6	1	6.5	3	6
7	2.5	5	4	4.5	1	5.5	2.5	5
8	3	3.5	3	3.5	1	4	3	3.5
9	4	4	2.5	4.5	1	6	2.5	4.5
10	2.5	4.5	2.5	4.5	1	5.5	4	4
11	2.5	5	1	5.5	4	4	2.5	5
12	2	5.5	4	4	1	6	3	4.5
Sum of ranks	30.5	(2)	33	(3)	18.5	(1)	38	(4)

Figures on the left in cells in the body of the table represent rank of median confidence ratings for each subject:
1: rated most confident
4: rated least confident

A Friedman test showed the difference in rankings overall to be significant at $p = 0.05$. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.4: Median Confidence Ratings by Angle of Motion: Experiment 1

	Angle of Rocking Motion							
	0.5		0.79		1.26		2	
	MEDIAN		MEDIAN		MEDIAN		MEDIAN	
	Comfort Rating		Comfort Rating		Comfort Rating		Comfort Rating	
SUBJECT								
1	2.5	6	2.5	6	2.5	6	2.5	6
2	3.5	4	1.5	4.5	1.5	4.5	3.5	4
3	3	3	3	3	1	4	3	3
4	3.5	5.5	3.5	5.5	2	6	1	6.5
5	3	4	3	4	1	5	3	4
6	2.5	7	2.5	7	2.5	7	2.5	7
7	1.5	5	1.5	5	4	4	3	4.5
8	2	6	3.5	5.5	1	6.5	3.5	5.5
9	2.5	7	2.5	7	2.5	7	2.5	7
10	3	4.5	2	5	1	5.5	4	3
11	3.5	6	1	7	2	6.5	3.5	6
12	3	4.5	3	4.5	3	4.5	1	5.5
Sum of ranks	33.5	(4)	29.5	(2)	24	(1)	33	(3)

Figures on the left in cells in the body of the table represent rank of median comfort ratings for each subject:
1: rated most comfortable
4: rated least comfortable

A Friedman test showed the difference in rankings overall to be not significant. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.5: Median Comfort Ratings by Angle of Motion: Experiment 1

B.4.2 Effect of Period of Motion

B.4.2.1 Response Time Analysis

The results of carrying out an analysis of variance on logged response times from experiment 2 with subject (SUBJECT) as a random effect and period of motion (JIGPER) as a fixed effect is shown below.

General Linear Models Procedure
Tests of Hypotheses for Mixed Model Analysis of Variance

Dependent Variable: RESPTIME

Source: JIGPER
Error: MS(JIGPER*SUBJECT)

DF	Type III MS	Denominator DF	Denominator MS	F Value	Pr > F
3	3.1163022437	33	0.2357783623	13.217	0.0001

Source: SUBJECT
Error: 0.9998*MS(JIGPER*SUBJECT) + 0.0002*MS(Error)

DF	Type III MS	Denominator DF	Denominator MS	F Value	Pr > F
11	33.698147656	33.01	0.2357586956	142.935	0.0001

Source: JIGPER*SUBJECT
Error: MS(Error)

DF	Type III MS	Denominator DF	Denominator MS	F Value	Pr > F
33	0.2357783623	3744	0.1153787403	2.044	0.0004

B.4.2.2 Data

The following tables show values of mean logged response times, total errors and median ease, confidence and comfort ratings for every subject by period of motion. Values are ranked for each subject and total rankings for all subjects are shown.

	Period of Rocking Motion			
	0.35	0.59	0.97	1.57
	MEAN	MEAN	MEAN	MEAN
	Log Response Time	Log Response Time	Log Response Time	Log Response Time
SUBJECT				
1	0.46	0.44	0.46	* 0.43
2	0.27	0.18	0.21	* 0.11
3	0.51	0.51	0.65	* 0.48
4	-0.01	-0.13	-0.07	* -0.24
5	-0.04	-0.09	-0.19	* -0.32
6	-0.27	-0.37	-0.35	* -0.53
7	0.42	0.40	0.37	* 0.36
8	0.15	0.14	* 0.09	0.11
9	-0.16	-0.26	-0.27	* -0.33
10	0.51	* 0.41	0.42	0.49
11	-0.21	-0.26	-0.30	* -0.48
12	0.14	0.11	0.14	* 0.04

* : subject minimum

Table B.6: Mean Logged Response Times by Period of Motion: Experiment 2

	Period of Rocking Motion							
	0.35		0.59		0.97		1.57	
	Total Errors		Total Errors		Total Errors		Total Errors	
SUBJECT								
1	4	11	1	2	2	9	3	10
2	2.5	5	2.5	5	1	4	4	8
3	3	7	2	4	1	3	4	14
4	2	3	3	4	1	2	4	12
5	1	9	2	11	3	13	4	18
6	1	3	2	5	4	14	3	9
7	3	7	1	3	2	5	4	13
8	3	10	2	6	1	5	4	13
9	1	4	2.5	10	4	12	2.5	10
10	2.5	9	2.5	9	1	7	4	10
11	2	3	1	2	3	13	4	15
12	3.5	8	1	4	2	6	3.5	8
Sum of ranks	28.5	(3)	22.5	(1)	25	(2)	44	(44)

Figures on the left in cells in the body of the table represent rank of total error rates for each subject:
1: least errors
4: most errors

A Friedman test showed the difference in rankings overall to be significant at $p = 0.005$. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.7: Total Errors by Period of Motion: Experiment 2

	Period of Rocking Motion							
	0.35		0.59		0.97		1.57	
	MEDIAN		MEDIAN		MEDIAN		MEDIAN	
	Ease Rating		Ease Rating		Ease Rating		Ease Rating	
SUBJECT								
1	3.5	3.5	1.5	4	3.5	3.5	1.5	4
2	1	6	2.5	5.5	4	4	2.5	5.5
3	1	5	2	4	3.5	3.5	3.5	3.5
4	2	5.5	1	6	3	4.5	4	3
5	2.5	5	1	5.5	2.5	5	4	4.5
6	1	5.5	3	4.5	4	3	2	5
7	3	4.5	1.5	5	4	4	1.5	5
8	2.5	5	1	5.5	2.5	5	4	3
9	2	6	2	6	4	5.5	2	6
10	2	6	2	6	4	5.5	2	6
11	2.5	4.5	1	5	4	2.5	2.5	4.5
12	1	6	3	5	2	5.5	4	4.5
Sum of ranks	24	(2)	21.5	(1)	41	(4)	33	(3)

Figures on the left in cells in the body of the table represent rank of median ease ratings for each subject:
1: rated easiest
4: rated least easy

A Friedman test showed the difference in rankings overall to be significant at $p = 0.05$. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.8: Median Ease Ratings by Period of Motion: Experiment 2

	Period of Rocking Motion							
	0.35		0.59		0.97		1.57	
	MEDIAN Confidence Rating		MEDIAN Confidence Rating		MEDIAN Confidence Rating		MEDIAN Confidence Rating	
SUBJECT								
1	3	3.5	1.5	4	4	3	1.5	4
2	1	6	2.5	5.5	4	5	2.5	5.5
3	1	4.5	2	4	3.5	3.5	3.5	3.5
4	1	6	2.5	5	2.5	5	4	4
5	4	4	1	5.5	2.5	4.5	2.5	4.5
6	2	5	2	5	4	2.5	2	5
7	3	4	1	4.5	3	4	3	4
8	2	5	2	5	2	5	4	4
9	1	6.5	2.5	4	4	3.5	2.5	4
10	4	5.5	2	6	2	6	2	2
11	1	5.5	2	4.5	4	3	3	4
12	1	6	4	3.5	3	4.5	2	5
Sum of ranks	24	(1)	25	(2)	38.5	(4)	32.5	(3)

Figures on the left in cells in the body of the table represent rank of median confidence ratings for each subject:
1: rated most confident
4: rated least confident

A Friedman test showed the difference in rankings overall to be not significant. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.9: Median Confidence Ratings by Period of Motion: Experiment 2

	Period of Rocking Motion			
	0.35	0.59	0.97	1.57
	MEDIAN	MEDIAN	MEDIAN	MEDIAN
	Comfort Rating	Comfort Rating	Comfort Rating	Comfort Rating
SUBJECT				
1	2 4	2 4	4 3.5	2 4
2	4 4	3 5.5	1.5 6	1.5 6
3	1.5 6	1.5 6	3 5	4 4.5
4	3 4.5	1.5 6	1.5 6	4 3
5	4 4.5	1 6.5	2 6	3 5
6	3.5 4	1.5 4.5	3.5 4	1.5 4.5
7	2.5 5.5	2.5 5.5	1 6	4 5
8	3 4	1 5	2 4.5	4 2
9	1 4.5	2.5 4	2.5 4	4 3
10	4 2	3 3.5	2 5.5	1 6.5
11	4 2	3 2.5	2 3.5	1 4
12	4 5.5	2 6	2 6	2 6
Sum of ranks	36.5 (4)	24.5 (1)	27 (2)	32 (3)

Figures on the left in cells in the body of the table represent rank of median comfort ratings for each subject:
1: rated most comfortable
4: rated least comfortable

A Friedman test showed the difference in rankings overall to be not significant. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.10: Median Comfort Ratings by Period of Motion: Experiment 2

B.4.3 Effect of Stimulus Type

B.4.3.1 Response Time Analysis: Experiment 1

The results of carrying out an analysis of variance on logged response times from experiment 1 with subject (SUBJECT) as a random effect and stimulus type (SHAPE) as a fixed effect is shown below.

General Linear Models Procedure
Tests of Hypotheses for Mixed Model Analysis of Variance

Dependent Variable: RESPTIME

Source: SUBJECT
Error: 0.9999*MS(SUBJECT*SHAPE) + 0.0001*MS(Error)

DF	Type III MS	Denominator DF	Denominator MS	F Value	Pr > F
11	27.421773739	33.00	0.2123720242	129.121	0.0001

Source: SHAPE
Error: MS(SUBJECT*SHAPE)

DF	Type III MS	Denominator DF	Denominator MS	F Value	Pr > F
3	0.2155220823	33	0.2123774046	1.015	0.3986

Source: SUBJECT*SHAPE
Error: MS(Error)

DF	Type III MS	Denominator DF	Denominator MS	F Value	Pr > F
33	0.2123774046	3756	0.1392263433	1.525	0.0279

B.4.3.2 Response Time Analysis: Experiment 2

The results of carrying out an analysis of variance on logged response times from experiment 2 with subject (SUBJECT) as a random effect and stimulus type (SHAPE) as a fixed effect is shown below.

General Linear Models Procedure
Tests of Hypotheses for Mixed Model Analysis of Variance

Dependent Variable: RESPTIME

Source: SUBJECT
Error: 0.9997*MS(SUBJECT*SHAPE) + 0.0003*MS(Error)

DF	Type III MS	Denominator DF	Denominator MS	F Value	Pr > F
11	33.699558889	33.01	0.3326377613	101.310	0.0001

Source: SHAPE
Error: MS(SUBJECT*SHAPE)

DF	Type III MS	Denominator DF	Denominator MS	F Value	Pr > F
3	0.9607803067	33	0.332709842	2.888	0.0502

Source: SUBJECT*SHAPE
Error: MS(Error)

DF	Type III MS	Denominator DF	Denominator MS	F Value	Pr > F
33	0.332709842	3744	0.1162515573	2.862	0.0001

B.4.3.3 Data

The following tables show values of mean logged response times, total errors and median ease, confidence and comfort ratings for every subject in each experiment by stimulus type.

	Stimulus Type							
	Layered		Hierarchic		Random		Regular	
	MEAN		MEAN		MEAN		MEAN	
	Log Response Time		Log Response Time		Log Response Time		Log Response Time	
SUBJECT								
1	4	0.41	2	0.29	1	0.26	3	0.34
2	2	0.08	1	0.07	4	0.19	3	0.15
3	3	-0.39	1	-0.42	3	-0.39	3	-0.39
4	3	0.56	4	0.64	2	0.51	1	0.48
5	4	-0.22	2	-0.25	3	-0.23	1	-0.27
6	1	0.10	3.5	0.15	2	0.12	3.5	0.15
7	4	0.05	1	-0.11	2	-0.07	3	-0.03
8	3	0.55	1.5	0.45	4	0.62	1.5	0.45
9	1	0.30	2	0.32	3.5	0.36	3.5	0.36
10	2.5	0.22	2.5	0.22	1	0.21	4	0.30
11	4	-0.01	1.5	-0.07	1.5	-0.07	3	-0.04
12	4	0.44	2	0.37	1	0.31	3	0.43
Sum of ranks	35.5	(4)	24	(1)	28	(2)	32.5	(3)

Figures on the left in each cell represent rank of mean logged response times for each subject:
1: responded most quickly
4: responded least quickly

Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.11: Mean Logged Response Times by Stimulus Type: Experiment 1

	Stimulus Type							
	Layered		Hierarchic		Random		Regular	
	MEAN Log Response Time		MEAN Log Response Time		MEAN Log Response Time		MEAN Log Response Time	
SUBJECT								
1	2.5	0.46	1	0.40	4	0.47	2.5	0.46
2	2	0.22	3	0.23	4	0.24	1	0.07
3	3	0.54	1	0.49	4	0.61	2	0.50
4	1.5	-0.19	3	-0.07	4	0.01	1.5	-0.19
5	2.5	-0.13	4	-0.05	2.5	-0.13	1	-0.35
6	4	-0.34	3	-0.38	2	-0.39	1	-0.43
7	3	0.38	2	0.35	4	0.52	1	0.30
8	1	0.07	4	0.18	3	0.13	2	0.10
9	1	-0.22	4	-0.30	3	-0.26	2	-0.24
10	3	0.46	4	0.54	1.5	0.42	1.5	0.42
11	3	-0.28	1	-0.40	2	-0.32	4	-0.25
12	4	0.14	3	0.11	1	0.09	2	0.10
Sum of ranks	30.5	(2)	33	(3)	35	(4)	21.5	(1)

Figures on the left in each cell represent rank of mean logged response times for each subject:
1: responded most quickly
4: responded least quickly

Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.12: Mean Logged Response Times by Stimulus Type: Experiment 2

	Stimulus Type							
	Layered		Hierarchic		Random		Regular	
	Total Errors		Total Errors		Total Errors		Total Errors	
SUBJECT								
1	3	11	1	7	3	11	3	11
2	3	11	1	7	4	12	2	8
3	1	13	3.5	20	2	15	3.5	20
4	3	16	1	9	2	14	4	17
5	3	12	1.5	10	1.5	10	4	14
6	2.5	11	1	7	2.5	11	4	18
7	3	12	1	7	2	11	4	14
8	2.5	5	4	11	1	4	2.5	5
9	2.5	13	1	10	4	14	2.5	13
10	2	9	1	5	4	17	3	15
11	3	12	2	11	1	10	4	16
12	1	2	3	8	2	6	4	10
Sum of ranks	29.5	(3)	21	(1)	29	(2)	40.5	(4)

Figures on the left in cells in the body of the table represent rank of total error rates for each subject:
1: least errors
4: most errors

A Friedman test showed the difference in rankings overall to be significant at $p = 0.05$. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.13: Total Errors by Stimulus Type: Experiment 1

	Stimulus Type							
	Layered		Hierarchic		Random		Regular	
	Total Errors		Total Errors		Total Errors		Total Errors	
SUBJECT								
1	2	6	3.5	12	3.5	12	1	2
2	2	4	4	9	3	6	1	3
3	2	3	4	15	3	10	1	0
4	2	3	4	9	3	7	1	2
5	2	12	4	17	1	6	3	16
6	2.5	9	2.5	9	4	10	1	3
7	3	9	4	12	2	7	1	0
8	3	10	4	13	2	8	1	3
9	1	5	3	9	4	15	2	7
10	2	7	4	15	3	12	1	1
11	2	6	4	13	3	10	1	4
12	2	4	4	13	3	8	1	1
Sum of ranks	25.5	(2)	44	(4)	37.5	(3)	13	(1)

Figures on the left in cells in the body of the table represent rank of total error rates for each subject:
1: least errors
4: most errors

A Friedman test showed the difference in rankings overall to be significant at $p = 0.001$. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.14: Total Errors by Stimulus Type: Experiment 2

	Stimulus Type							
	Layered		Hierarchic		Random		Regular	
	MEDIAN		MEDIAN		MEDIAN		MEDIAN	
	Ease Rating		Ease Rating		Ease Rating		Ease Rating	
SUBJECT								
1	2	5.5	2	5.5	2	5.5	4	4
2	3.5	4.5	1.5	5	1.5	5	3.5	4.5
3	3	4	4	3.5	1.5	5	1.5	5
4	4	4	1.5	5	1.5	5	3	4.5
5	4	3.5	1.5	5	1.5	5	3	4.5
6	1.5	4.5	4	3.5	1.5	4.5	3	4
7	4	4	1.5	5.5	3	4.5	1.5	5.5
8	3	4.5	4	4	1.5	5	1.5	5
9	3	5	3	5	3	5	1	5.5
10	1	4.5	2.5	4	4	3.5	2.5	4
11	4	4	1	6	2.5	4.5	2.5	4.5
12	3.5	4.5	1	5.5	3.5	4.5	2	5
Sum of ranks	36.5	(4)	27.5	(2)	27	(1)	29	(3)

Figures on the left in cells in the body of the table represent rank of median ease ratings for each subject:
1: rated easiest
4: rated least easy

A Friedman test showed the difference in rankings overall to be not significant. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.15: Median Ease Ratings by Stimulus Type: Experiment 1

	Stimulus Type							
	Layered		Hierarchic		Random		Regular	
	MEDIAN		MEDIAN		MEDIAN		MEDIAN	
	Ease Rating		Ease Rating		Ease Rating		Ease Rating	
SUBJECT								
1	1.5	4	1.4	4	3.5	3.5	3.5	3.5
2	1.5	6	3.5	4.5	3.5	4.5	1.5	6
3	2	4.5	3	3.5	4	2.5	1	6
4	4	3.5	2	4.5	3	4	1	5.5
5	3	5	3	5	3	5	1	6
6	3.5	4	3.5	4	1.5	5	1.5	5
7	4	4	3	4.5	1.5	5	1.5	5
8	3.5	4	2	4.5	3.5	4	1	5.5
9	1	4.5	4	3	2.5	3.5	2.5	3.5
10	2	6	2	6	4	5	2	6
11	2	4.5	1	5	4	3.5	3	4
12	3	5	2	5.5	1	6	4	4
Sum of ranks	31	(3)	30.5	(2)	35	(4)	23.5	(1)

Figures on the left in cells in the body of the table represent rank of median ease ratings for each subject:
1: rated easiest
4: rated least easy

A Friedman test showed the difference in rankings overall to be not significant. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.16: Median Ease Ratings by Stimulus Type: Experiment 2

	Stimulus Type							
	Layered		Hierarchic		Random		Regular	
	MEDIAN Confidence Rating		MEDIAN Confidence Rating		MEDIAN Confidence Rating		MEDIAN Confidence Rating	
SUBJECT								
1	1.5	5.5	1.5	5.5	3	5	4	4.5
2	2.5	5	1	6	4	4.5	2.5	5
3	2.5	4	1	4.5	2.5	4	4	3.5
4	3.5	4	1.5	4.5	3.5	4	1.5	4.5
5	2	4.5	2	4.5	4	4	2	4.5
6	2.5	6	2.5	6	2.5	6	2.5	6
7	4	4.5	1.5	5.5	3	5	1.5	5.5
8	1.5	4	4	3	3	3.5	1.5	4
9	4	4	2.5	4.5	2.5	4.5	1	5
10	4	4	1	5	2.5	4.5	2.5	4.5
11	4	3.5	1	6	2	5	3	4.5
12	3.5	4.5	1	5.5	3.5	4.5	2	5
Sum of ranks	35.5	(3)	20.5	(1)	36	(4)	28	(2)

Figures on the left in cells in the body of the table represent rank of median confidence ratings for each subject:
1: rated most confident
4: rated least confident

A Friedman test showed the difference in rankings overall to be significant at $p = 0.05$. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.17: Median Confidence Ratings by Stimulus Type: Experiment 1

	Stimulus Type							
	Layered		Hierarchic		Random		Regular	
	MEDIAN Confidence Rating		MEDIAN Confidence Rating		MEDIAN Confidence Rating		MEDIAN Confidence Rating	
SUBJECT								
1	1.5	4	1.5	4	4	3	3	3.5
2	1.5	6	3.5	5	3.5	5	1.5	6
3	2	4	3	3.5	4	3	1	6
4	3	4.5	1.5	5.5	4	4	1.5	5.5
5	2.5	4.5	2.5	4.5	4	4	1	5.5
6	4	4	3	4.5	2	5	1	6
7	3	4	3	4	3	4	1	4.5
8	3.5	4.5	1.5	5	3.5	4.5	1.5	5
9	2	4.5	4	4	2	4.5	2	4.5
10	3	5.5	1.5	6.5	4	5	1.5	6.5
11	2	4.5	1	5	4	3	3	4
12	4	4	3	5	1	6	2	4.5
Sum of ranks	32	(3)	29	(2)	39	(4)	20	(1)

Figures on the left in cells in the body of the table represent rank of median confidence ratings for each subject:
1: rated most confident
4: rated least confident

A Friedman test showed the difference in rankings overall to be significant at $p = 0.05$. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.18: Median Confidence Ratings by Stimulus Type: Experiment 2

	Stimulus Type							
	Layered		Hierarchic		Random		Regular	
	MEDIAN		MEDIAN		MEDIAN		MEDIAN	
	Comfort Rating		Comfort Rating		Comfort Rating		Comfort Rating	
SUBJECT								
1	1	6.5	3	6	3	6	3	6
2	1.5	4.5	1.5	4.5	3.5	4	3.5	4
3	1	4	4	2.5	2	3.5	3	3
4	1.5	6.5	4	4.5	1.5	6.5	3	5
5	1	4.5	3	4	3	4	3	4
6	2	7	4	6	2	7	2	7
7	1.5	5	4	2.5	1.5	5	3	4.5
8	2	6	4	5	2	6	2	6
9	2.5	7	2.5	7	2.5	7	2.5	7
10	4	3	2	5	2	5	2	5
11	4	5.5	2	6.5	2	6.5	2	6.5
12	2.5	5	4	4.5	1	5.5	2.5	5
Sum of ranks	24.5	(1)	38	(4)	26	(2)	31.5	(3)

Figures on the left in cells in the body of the table represent rank of median comfort ratings for each subject:
1: rated most comfortable
4: rated least comfortable

A Friedman test showed the difference in rankings overall to be not significant. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.19: Median Comfort Ratings by Stimulus Type: Experiment 1

	Stimulus Type							
	Layered		Hierarchic		Random		Regular	
	MEDIAN Comfort Rating		MEDIAN Comfort Rating		MEDIAN Comfort Rating		MEDIAN Comfort Rating	
SUBJECT								
1	1.5	4	3.5	3.5	1.5	4	3.5	3.5
2	1.5	6	4	4.5	1.5	6	3	5.5
3	2.5	5.5	2.5	5.5	4	4	1	6
4	4	4	2	5.5	1	6	3	4.5
5	1.5	6	3	5	4	4.5	1.5	6
6	2.5	4	2.5	4	2.5	4	2.5	4
7	2.5	5.5	4	5	2.5	5.5	1	6
8	3	4	1.5	4.5	1.5	4.5	4	3.5
9	1	4.5	4	3	3	3.5	2	4
10	2	5	1	5.5	3.5	4	3.5	4
11	4	2.5	1.5	3.5	1.5	3.5	3	3
12	3	5.5	1.5	6	1.5	6	4	5
Sum of ranks	29	(2)	31	(3)	28	(1)	32	(4)

Figures on the left in cells in the body of the table represent rank of median comfort ratings for each subject:
1: rated most comfortable
4: rated least comfortable

A Friedman test showed the difference in rankings overall to be not significant. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.20: Median Comfort Ratings by Stimulus Type: Experiment 2

	Stimulus Type							
	Layered		Hierarchic		Random		Regular	
	MEDIAN		MEDIAN		MEDIAN		MEDIAN	
	Complexity Rating		Complexity Rating		Complexity Rating		Complexity Rating	
SUBJECT								
1	1.5	3.5	3.5	5.5	3.5	5.5	1.5	3.5
2	3	3.5	3	3.5	3	3.5	1	2
3	2	3.5	3	4	4	5	1	2
4	2	3.5	4	5	3	4.5	1	1.5
5	2	4	3.5	5	3.5	5	1	2.5
6	2.5	5.5	4	6	2.5	5.5	1	2.5
7	2	5	3	5.5	4	6	1	4
8	2	5	4	6	3	5.5	1	3
9	3.5	5	2	4.5	3.5	5	1	1
10	2	4	3	5.5	4	6	1	2
11	2	4.5	4	6	3	5	1	1
12	3	5	3	5	3	5	1	2.5
Sum of ranks	27.5	(2)	40	(3.5)	40	(3.5)	12.5	(1)

Figures on the left in cells in the body of the table represent rank of median complexity ratings for each subject:
1: rated least complex
4: rated most complex

A Friedman test showed the difference in rankings overall to be significant at $p = 0.001$. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.21: Median Complexity Ratings by Stimulus Type: Experiment 1

	Stimulus Type							
	Layered		Hierarchic		Random		Regular	
	MEDIAN Complexity Rating		MEDIAN Complexity Rating		MEDIAN Complexity Rating		MEDIAN Complexity Rating	
SUBJECT								
1	3	5	2	3.5	4	5.5	1	2.5
2	2	4	3	5	4	6.5	1	2.5
3	2	4.5	3	5	4	6	1	2
4	2	4.5	3	5	4	5.5	1	2
5	2	4	3.5	5	3.5	5	1	3
6	2.5	5	2.5	5	4	6	1	2.5
7	2.5	5	2.5	5	4	5.5	1	4
8	2	5	4	6	3	5.5	1	2
9	2	5	4	7	3	6	1	3
10	3	5	2	3	4	5.5	1	2
11	3	3.5	1.5	3	4	5.5	1.5	3
12	2	5.5	3.5	6	3.5	6	1	3
Sum of ranks	28	(2)	34.5	(3)	45	(4)	12.5	(1)

Figures on the left in each cell represent rank of median complexity ratings for each subject:
1: rated least complex
4: rated most complex

A Friedman test showed the difference in rankings overall to be significant at $p = 0.001$. Figures on the right in cells in the bottom row represent ranks of sums of ranks.

Table B.22: Median Complexity Ratings by Stimulus Type: Experiment 2

Appendix C

ICDEDIT: A Brief Description

David Dodson

Computer Science Department, City University,
Northampton Square, London EC1V 0HB, UK

C.1 Introduction

"Connection diagrams" is a generic term for diagrams which show nodes interconnected by links. Figure C.1 is an example. Many varieties of connection diagram are used as direct-manipulation media, through which to specify, observe, control or modify whatever they can show. *"Interactive Connection Diagrams"* (ICDs) is a generic term for such media, analogous to "Interactive Graphics". There is growing interest in 3-d ICDs. As with other artificial realities, remarkably realistic implementations may become commonplace as graphics chips gain power. Yet little attention has been given to the possibility of useful implementations on workstations without real-time 3-d graphics hardware.

"2¾-d" ICDs are a slightly impoverished form of 3-d ICDs in which nodes carry no orientation information. Viewing controls can still allow the 3-d positional arrangements of diagram elements to be rotated relative to any axis, but nodes always appear the same way round. This allows 2-d graphics hardware and specialised display update methods to deliver acceptable responsiveness in many applications involving interactive changes to diagram content or layout. ("2¾-d" is mid-way between 3-d and 2½-d; in CAD/CAM terms 2½-d means 3-d without depth-affecting rotations available.)

ICDEDIT ("I-C-D-edit") is an experimental 2¾-d ICD graphics tool. This system, developed at City University, runs on a SUN 3/60C workstation under SunView. It allows nodes to be dragged around in 3-d by the mouse, with their links adjusting like elastic. Display update is facilitated by a diagram paradigm of flat rectangular nodes and single-pixel wide links. Figure C.1 is a screen dump from part of the system's diagram display panel.

User-controlled viewing parameters allow the 2¾-d diagram to be rotated, shifted, scaled and foreshortened by any amount, and allow it to rock to and fro with variable rocking-axis orientation, amplitude and frequency. This rocking motion, used to reinforce depth perception, is presented by rapidly cycling the display through each of three stored views,

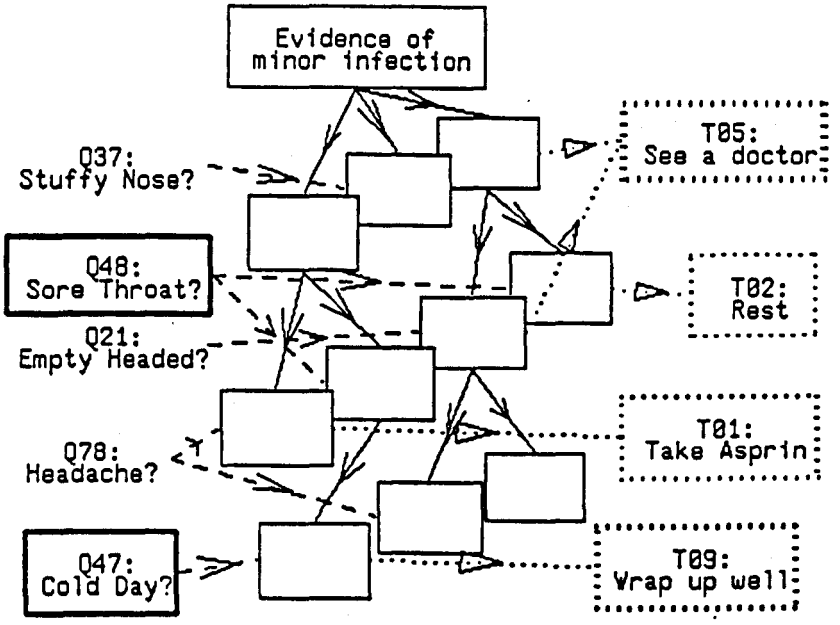


Figure C.1: A Small ICDEDIT Diagram

This diagram occupied only about 25% of the ICDEDIT diagram panel area, illustrating potential compactness. Hence the low image resolution.
The blank nodes form a hierarchy of disease categories. The dashed and dotted links represent evidence flows.
It is thought that the complexity of this diagram is probably maximal for potential end users, but low for knowledge engineers.

each with slightly different viewing angles. When viewing parameters are changed, redrawing the three stored views (using object depth-sorting) is not particularly rapid. However, local display update algorithms using raster operations allow acceptable responsiveness in displaying nodes being dragged in 3-d.

The system is not tailored for specific applications or domains, though it was designed with knowledge-based system applications in mind. The aim has been to gain understanding of techniques and design issues relevant to 3-d ICDs and to provide graphics support for further research and development. As an ICD Graphics Layer process, it can connect to and communicate with separate and possibly remote processes. User actions are translated into outgoing application-independent 'Abstract Diagrammatic Interaction' messages, and incoming messages are translated into into display updates. A separate ICD Intermediate Layer process is intended to manage semantic mapping and implement self-organising diagram dynamics.

C.2 The ICDEDIT 2¾-d Diagram Paradigm

The ICDEDIT diagram paradigm, then, is a four-colour 2¾-d ICD paradigm with rectangular nodes and single-pixel-wide links. The full specification is essentially as follows.

Straight-Line Links. Each ICDEDIT link follows an kinkless straight line path between two nodes, the norm in knowledge-based system applications of ICDs. This scheme has the merit of not requiring shape specification, at the cost of links tending to interfere with nodes in complex 2-d structures. Other widely observed advantages of straight links are that users can more easily comprehend the connectivity of multiply connected structures, and can visually scan along links faster and more reliably, quickly inferring continuity where small parts of links are obscured by nodes.

Link Types. Two main kinds of link are currently available. Left-to-right links join the mid-point of the right-hand side of one node to the mid-point of the left-hand side of a node centred further to the right. Top-to-bottom links analogously connect one node's bottom mid-point to another's top mid-point.

Visible Attributes of Diagram Elements. Each node is a 2-d right rectangle containing zero or more lines of text. Each can have a transparent or opaque body and can optionally have a border 1 to 3 pixels thick. A node's text can be in any of four colours, as can its body, if opaque. Node borders and links come in several alternative line types (solid, dotted, dashed etc.) and in any of the same four colours. Two alternate types of arrow mark are available on links. The four-colour palette can be edited by the user.

'Hidden' Nodes and Link Stubs. Any node, as well as possibly being obscured by some other(s), can optionally be 'hidden', retaining its position but becoming invisible. Any link from an viewed node (a non-'hidden' node in the view pyramid) to a non-viewed node takes the form of a short link 'stub' on the former node, pointing in the direction of the latter. Non-viewed nodes can be viewed via clicking on link stubs which point to them. Links between pairs of non-viewed nodes are not displayed.

Rocking Motion. Rocking motion is optional. Any axis of rotation parallel to the screen can be selected, as can any amplitude of rotation and any frequency between ⅓ Hz and about 10Hz. To achieve rocking motion, the colour map cycles the diagram display through three slightly different views, each of which is maintained in its own pair of bit planes. To yield an approximately sinusoidal motion in which each view is shown for one-third of the time, the 'middle' view's appearances are twice as frequent but half as long as either of the 'side' views' appearances.

There are also useful optional display modes for:

- Annotating nodes to show their location in world coordinates;
- Streamlined node dragging with approximate display update, which can subsequently be tidied by redrawing.

This paradigm has the property that semantic conventions introduced for specific diagram applications are readily expressed in self-explanatory 'keys' within the diagram paradigm of the tool.

C.3 The ICDEDIT User Action Paradigm

The ICDEDIT user action paradigm is only sketched here as the findings to be presented relate mainly to diagram paradigms. Dodson (1990b) gives further details.

Mouse Actions. Mouse buttoning conventions in ICDEDIT are fixed as follows:

- The left button provides *'actuation'*, much as in SunView Actuating a node or link sends a message recording the actuation to any connected processes, which should respond accordingly, if appropriate.
- The middle button provides node *'positioning'*. Existing nodes can be dragged with this button down. By default, dragging is parallel to the (vertical) screen, except that whilst the shift key is down, it occurs parallel to the (horizontal) mouse pad. This provides an acceptable means of 3-d dragging. Clicking with this button other than on a node creates a new node with default parameters as set by a control panel. This node is positioned at the point at which the normal to the screen through the locator intersects a plane lying at 45 deg to the screen and intersecting the screen along a horizontal line through the centre of the diagram display panel. This allows the initial depth of the new node to be controlled by its initial vertical position. Whilst the button is held down, dragging can then take place as for an existing node. This allows a new node to be positioned in 3-d with a single click-down, drag, click-up sequence.
- The right button provides a generalised *'menu'* function, including selection of a node (or link) as the referent of a node (or link) edit panel.

Control Panels. The main control panel contains:

- Control panel buttons for loading diagrams, storing diagrams, establishing communication with another process, view control, colour palette control, etc.. Each of these calls up a control panel overlay for the relevant function.
- Display style selectors, a refresh button for redrawing from scratch, and a slider controlling rocking motion frequency.
- A node editing region, with alternate control-panels for the default attributes for new nodes and for the attributes of the current node. .The latter panel also has buttons for node deletion, for node 'hiding', and for the creation of links between nodes.
- A link editing region, with alternate control-panels for the default attributes of new links and for the attributes of the current link. The latter panel also has buttons for link deletion and for the revelation of hidden nodes pointed to by link stubs.

All control panels operate according to the normal SunView mouse action conventions, which are consistent with those listed above.

Semantic Feedback. Messages are sent to any attached process detailing all user actions on the diagram and any changes to its viewing or colour parameters. Semantic feedback is dependent on receiving system action messages in response, which can change the diagram (and thus even cancel the user's action). As a result, semantic feedback, once implemented, may well take longer than the 0.1 seconds needed to approach perceptual

immediacy. (As yet, we have not linked ICDEDIT to anything that can provide feedback, other than another ICDEDIT process back-to-back.)

This lack of immediacy could be partly alleviated by allowing a rich variety of diagram element types with built-in interactive behaviours to be constructed from suitable primitives. Thus, for example, attempts to manipulate diagram element attributes beyond variable limit parameters could be met with instant resistance. Such element types could be constructed interactively as well as being programmable and communicable between processes. Beyond this, it is conjectured that momentary delay in 'deeper' semantic feedback will often usefully help the user understand the origin of responses in a layered interface architecture.

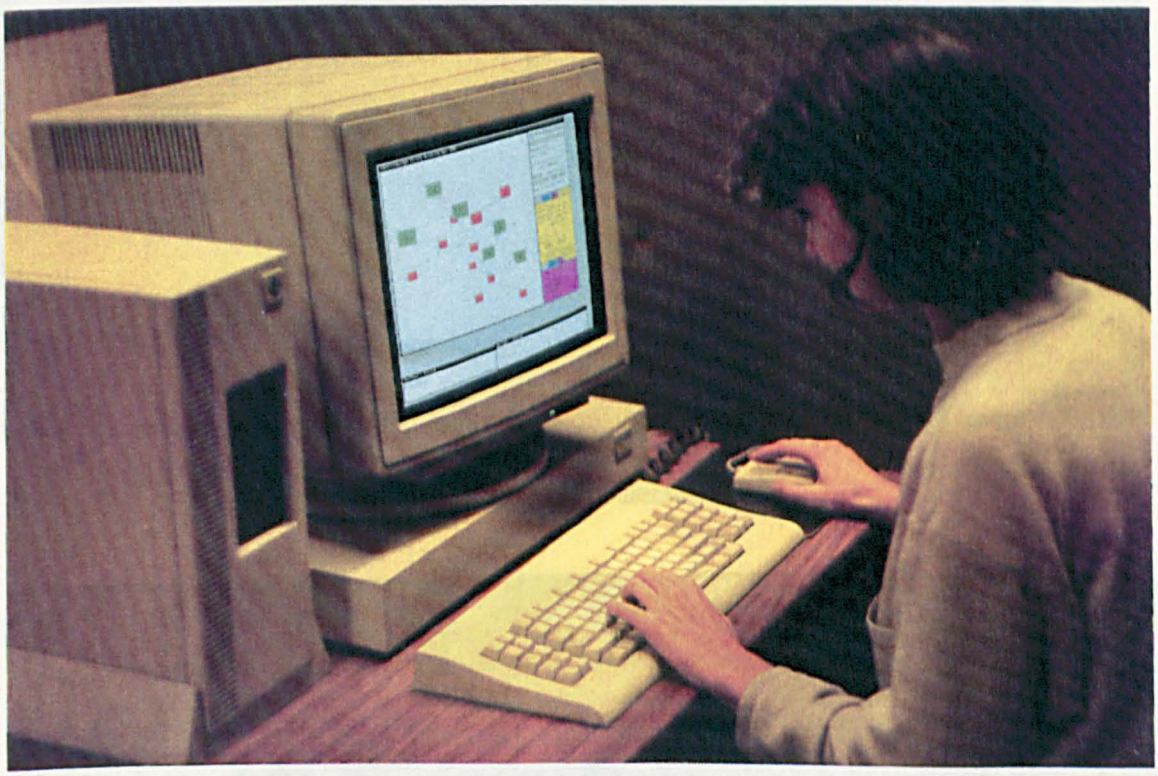


Figure C.2: ICDEDIT in Use

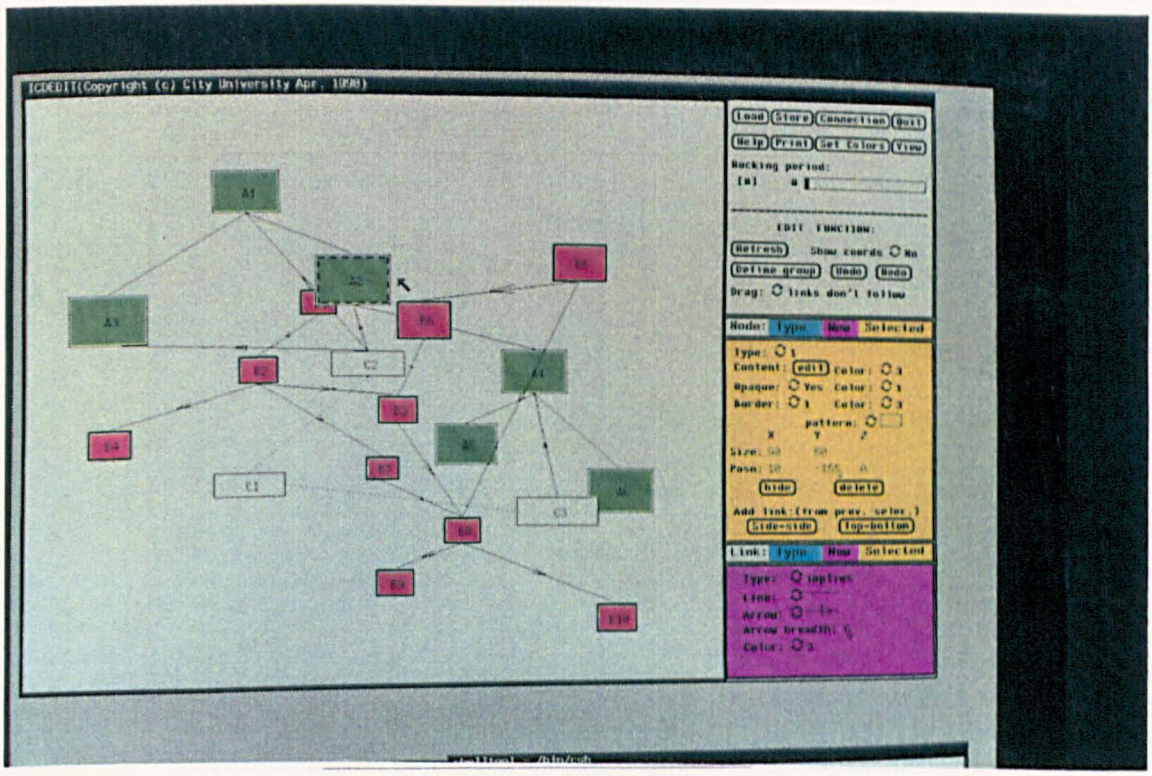


Figure C.3: Node Selected

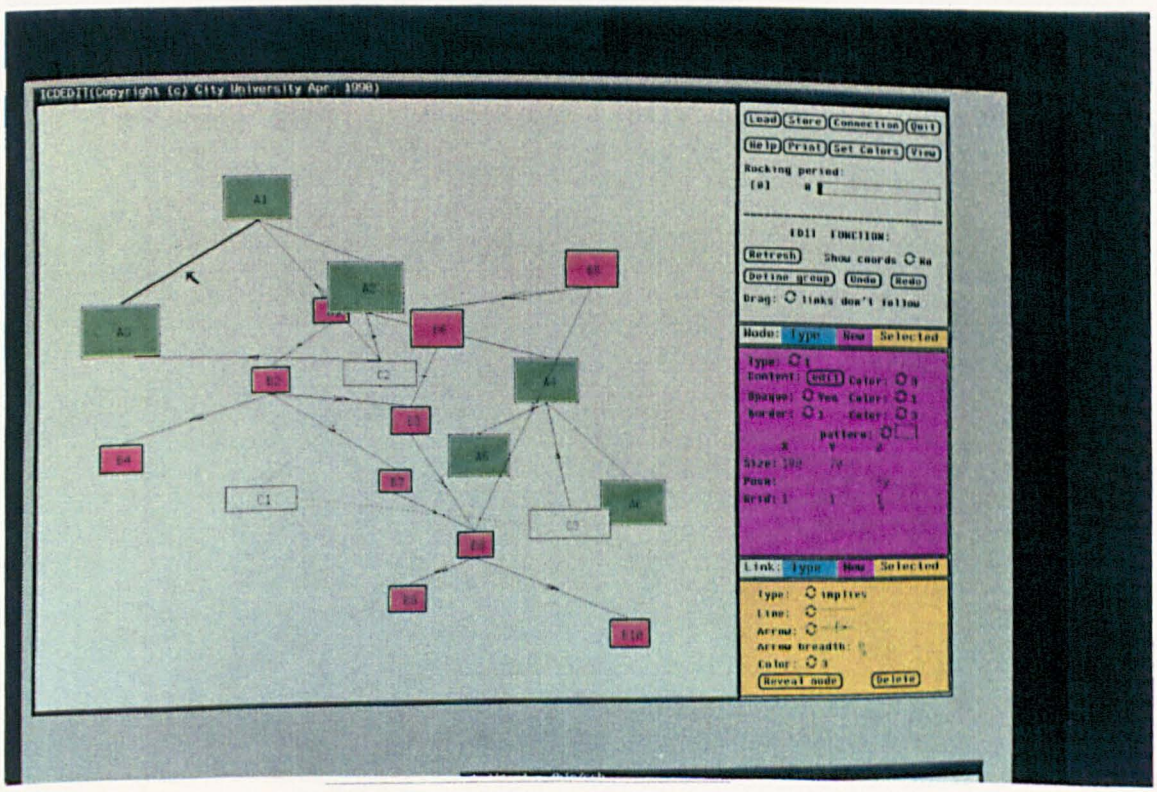


Figure C.4: Link Selected

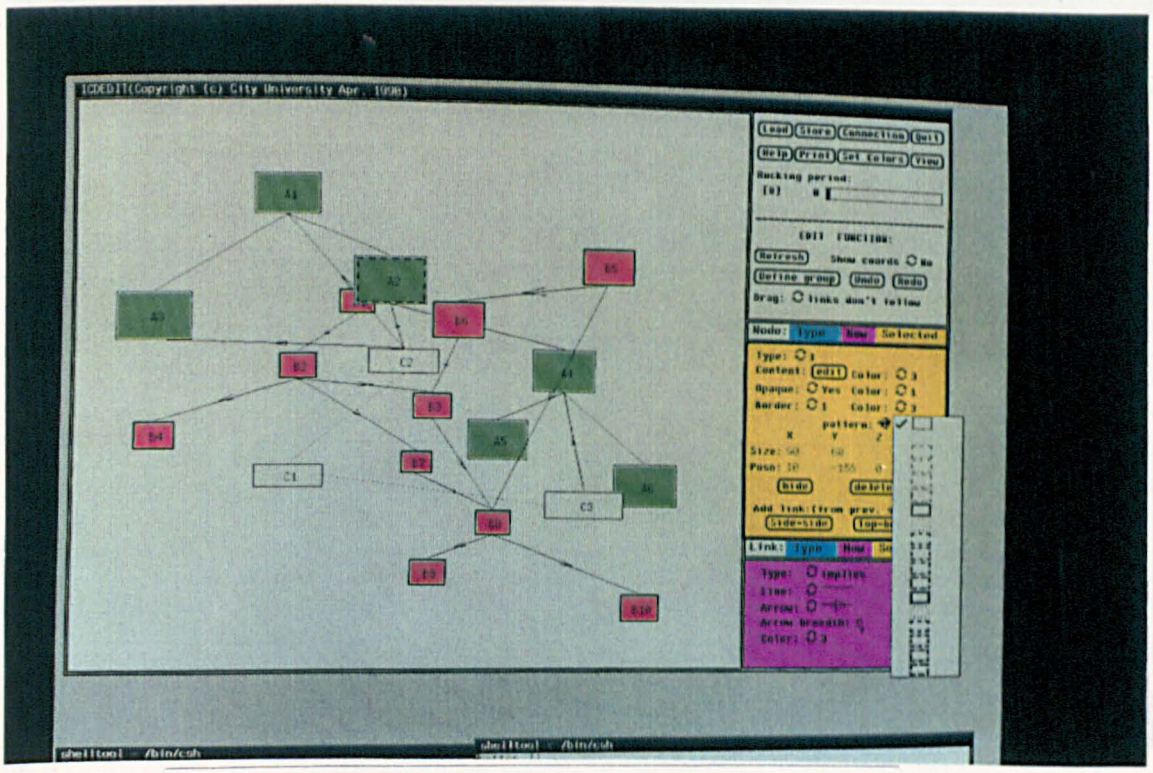


Figure C.5: Selected Node Border Edit Menu Raised

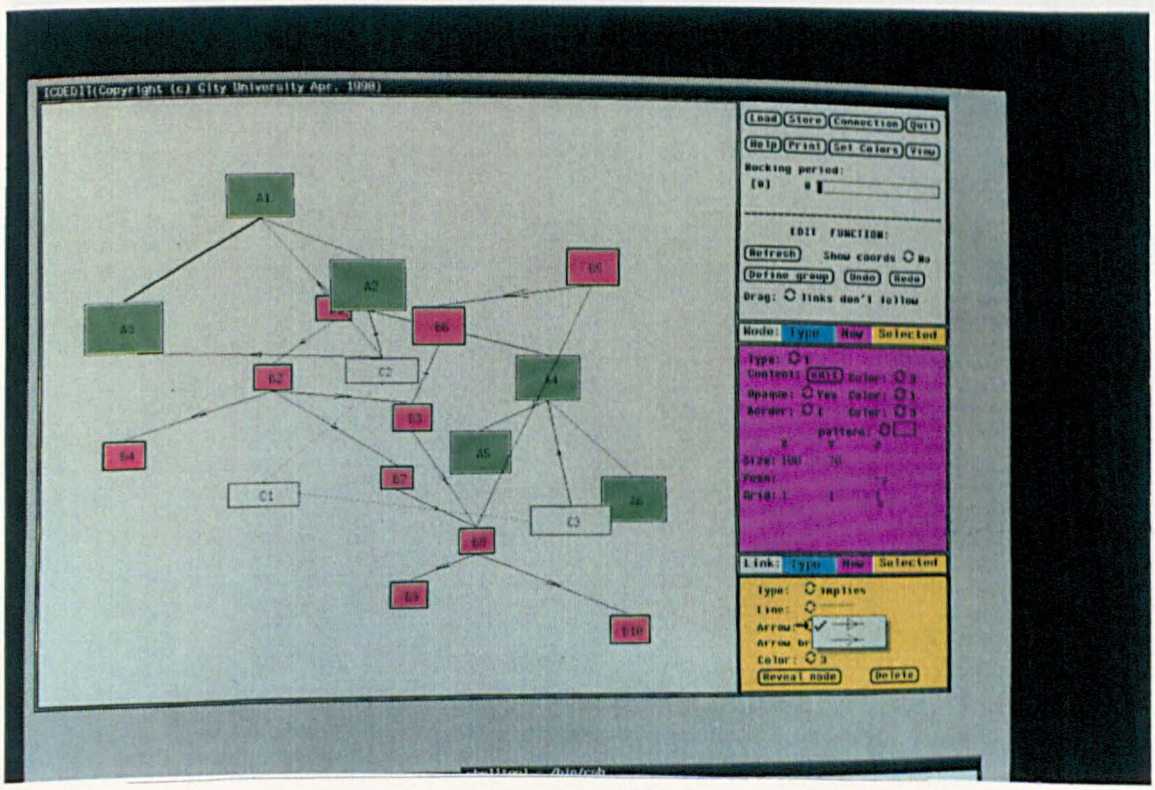


Figure C.6: Selected Link Arrow Head Edit Menu Raised

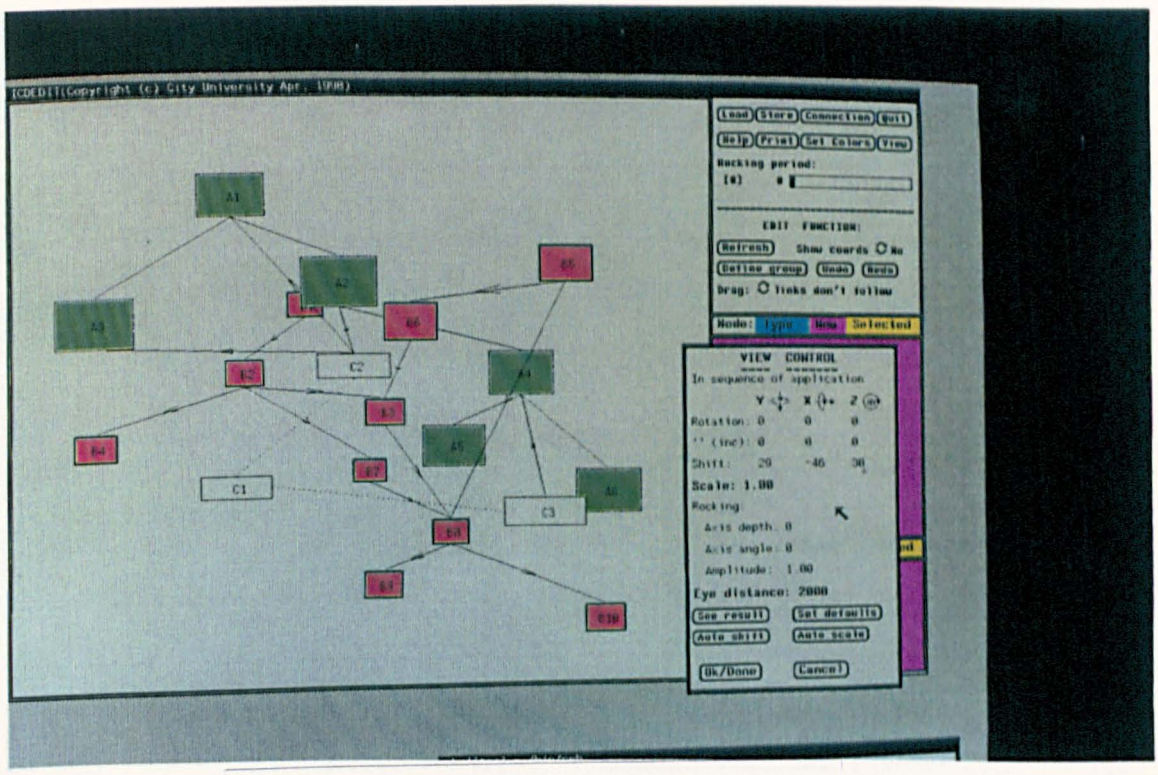


Figure C.7: View Control Panel Raised

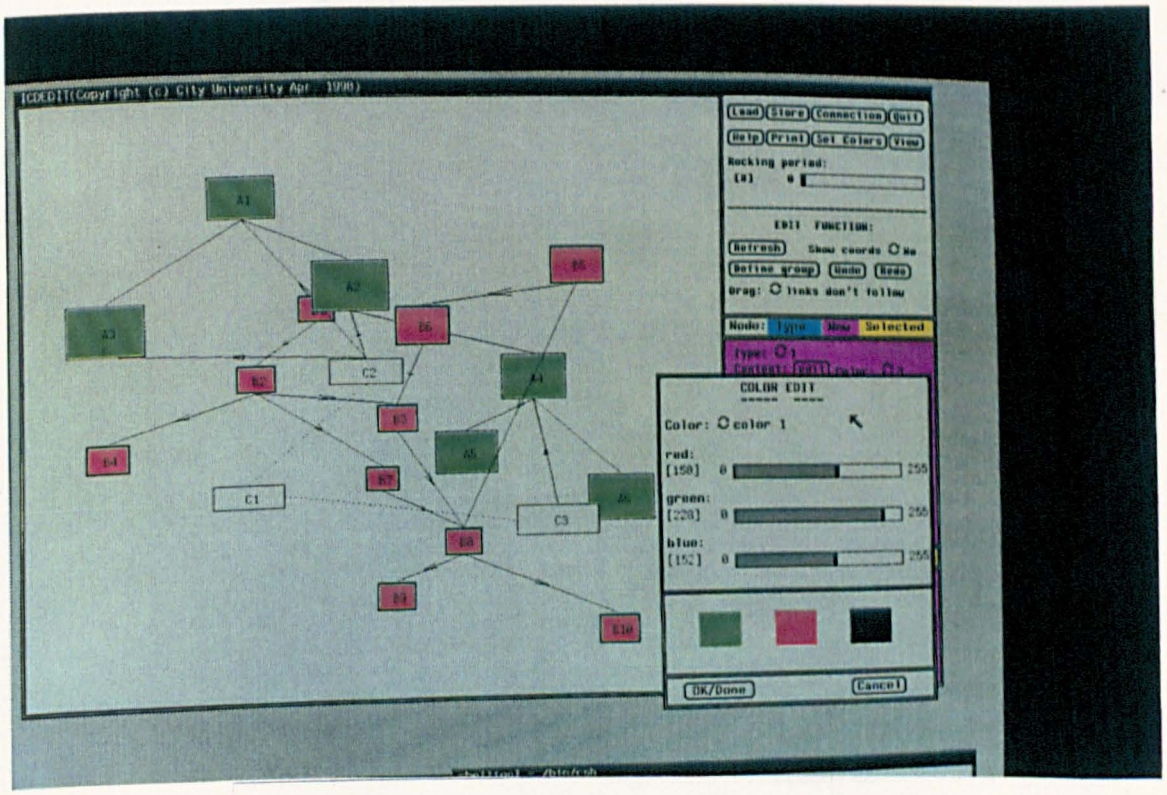


Figure C.8: Colour Edit Panel Raised

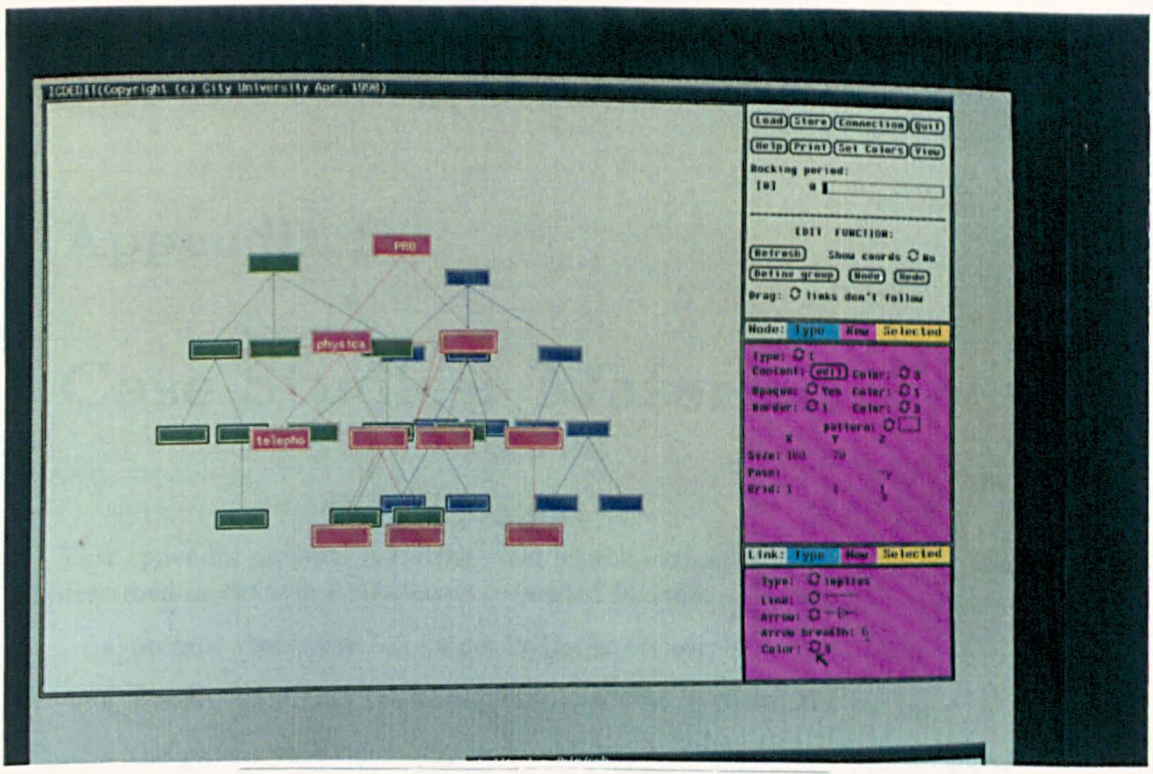


Figure C.9: Frontal View of a 3-d ICD

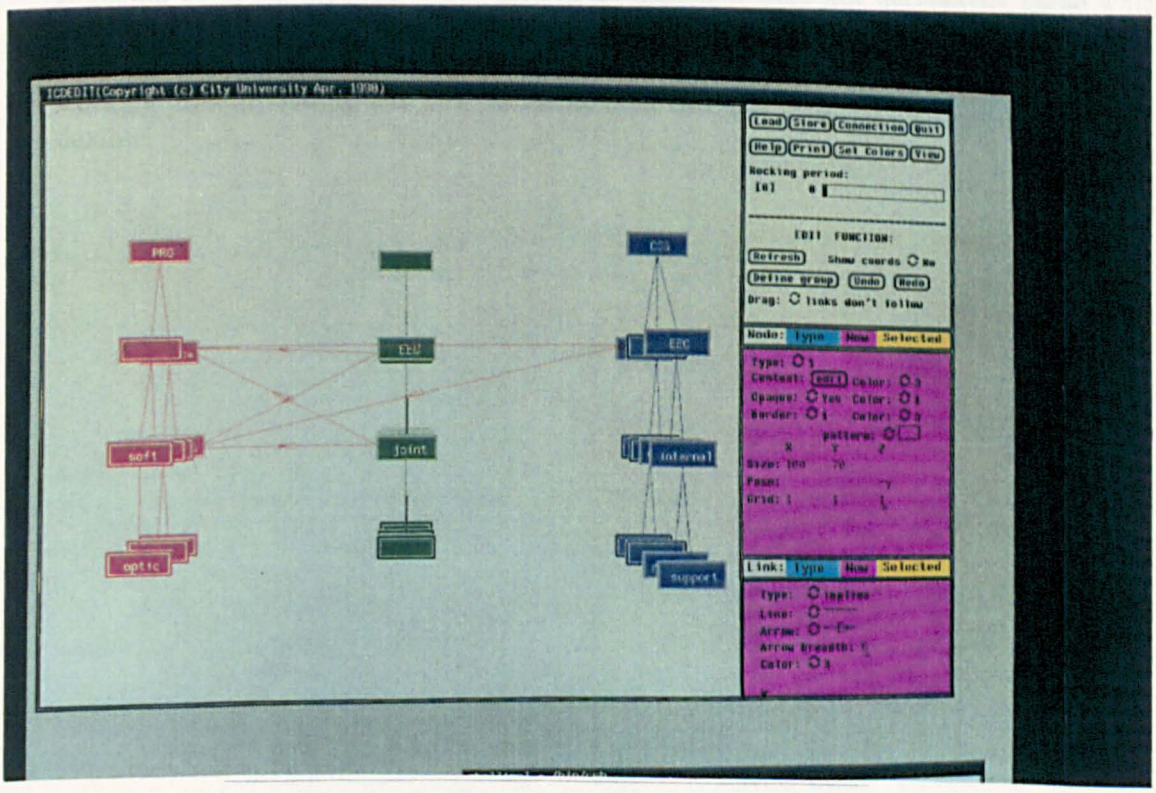


Figure C.10: Side View of the Same 3-d ICD

Appendix D

Case Studies: Materials

This appendix presents materials used by the evaluator in carrying out the case studies described in chapter 7. Materials presented include:

- prompt sheets used as guides in the interviews
- a sheet describing the vocabulary available in ICDEDIT
- the generic task taxonomy from which specialised taxonomies for particular studies were derived
- the graphical representations of knowledge structures used in each of the studies

It should be noted that the words used in an interview were not necessarily those which appear on the prompt sheets. Since the interviews were semi-structured, the wording of the questions needed to be flexible in order to allow for deviations from the plan. The wording used in reading out task instructions in the final stages of a study was similarly flexible.

D.1 Stage 1

D.1.1 Interview Prompt Sheet

INTRODUCTION
==========

Hello, my name is Sara Jones, I'm a research student in the department of Computer Science at City University.

I'm in the final year of a PhD concerned with investigating new forms of interface for tools for knowledge engineering:

- I'm interviewing a number of people in the field and will be carrying out a number of case studies at various institutions including your own

Each case study consists of 3 stages:

- in stage 1 (which is what we're doing now), I will ask you for some background information about your organisation or institution, your own professional experience and your likes and dislikes with respect to existing tools for knowledge engineering. I will also ask you for some suggestions regarding recent projects in which you have been involved and which you feel might provide suitable material for the rest of the study
- in stage 2, I will return with some preliminary ideas for the representation of the material you suggest using and we will discuss how these might be further developed
- in stage 3, you will come to City to use a tool we have there called ICDEDIT to perform some simple tasks using the representation developed

Interviews will be taped and task performance during stage 3 videoed subject to your agreement.

The findings of these studies will be reported in such a way that all comments are anonymous, and any references to specific hardware, software or organisations will be removed.

I'd like to stress that this interview is intended to be flexible: please feel free to interrupt, ask questions, or discuss anything you feel is relevant that I haven't brought up.

Do you have any questions?

BACKGROUND INFORMATION

Organisational Involvement in Knowledge Engineering

Length of involvement:
- number of years
- number of projects

Current involvement:
- number of people
- number of projects

Methods for Knowledge Engineering

Does your organisation use any structured methods for knowledge engineering?

How many people are typically involved in the development of a single knowledge-based system?

- how much time do members of the project team spend working together/ alone?
- how do members of the project team communicate?

What roles do people involved in the development of a knowledge-based system play?

- what are their professional skills
- what is their position within the organisation
- how much time do they contribute

Where does work take place?

What hardware and software is commonly used?

PROFESSIONAL EXPERIENCE, PREFERENCES REGARDING TOOLS

Personal Involvement in Knowledge Engineering

Length of involvement

Previous background

Knowledge engineering projects involved in:

- what type of system were you building
- what knowledge representation paradigms were you using
- what (if any) development method did you use
- what was your role
- what tools did you use

Experience of Tools for Knowledge Engineering

What percentage of your time (while involved in knowledge-based system development) do you spend using computer-based tools?

- what other tools do you use and why?

List computer-based tools

- of which you have extensive experience
- of which you have some experience
- which you have seen working

Of the tools you have used

- which features did you particularly like or dislike and why?
- what did you think of the graphical representations they used (if any)

Given a free choice, what tools or facilities would you most like to have available to you?

Reactions to the Possibility of Using 3-d Node and Link Diagrams

What are your initial reactions to the possibility of using 3-d node and link diagrams for knowledge engineering?

- can you see any way in which they might be useful?

What are your initial reactions to the possibility of using special-purpose hardware such as a helmet or spectacles for viewing 3-d diagrams?

- would you feel comfortable using it?
- would it be feasible in your normal environment?
- would it be feasible considering your normal activities?

PROJECT DETAILS
==========

How big is/was the project of interest?

- size of project team
- number of person years

What is/was your role in the project?

What is/was the purpose of the system?

- how big was it / is it now / is it intended to be?
- what knowledge representation paradigms does it use?
- what hardware and software does it use?

What methods are/were used in development?

What sort of representations of the knowledge were used during development?

- do you have any examples of the kind of representations used?

- for each representation:
    - what does it mean?
    - what are the important things shown?
    - who uses it?
    - what for?

Do you have any suggestions for 3-d representations of any of the knowledge structures used?

- what are the important components of the structure to be represented?
- have you any specific suggestions regarding the way in which they should be represented?

END
===

Thankyou for your time.

I will go away and develop some initial ideas for 3-d ICD representations of the knowledge structures we have discussed.

D.2 Stage 2

D.2.1 Interview Prompt Sheet

The aim of this session is just to check I have a correct understanding of what we talked about in stage 1, and to discuss some suggestions regarding a 3-d ICD representation for the knowledge structure we decided on.

Firstly, I have some questions about the knowledge structure:

- (ask questions)

This sheet should give you some idea of the graphical vocabulary available in ICDEDIT:

- (talk through vocabulary shown in figure \ref{fig:ICDEDITvocab})

Here are some examples of representations developed for previous studies:

- (talk through examples where possible)

Now I'd like to discuss some ideas for a representation of your knowledge structure:

- do I have a correct list of the important components to be represented?
- in the last session, you suggested that components might be represented as follows:

- (remind subject of any suggested codings)

- do you have any further suggestions?
- what do you think of the following suggestions

- (discuss any further suggestions from the experimenter regarding aspects of the representation not covered by the subject's own suggestions)

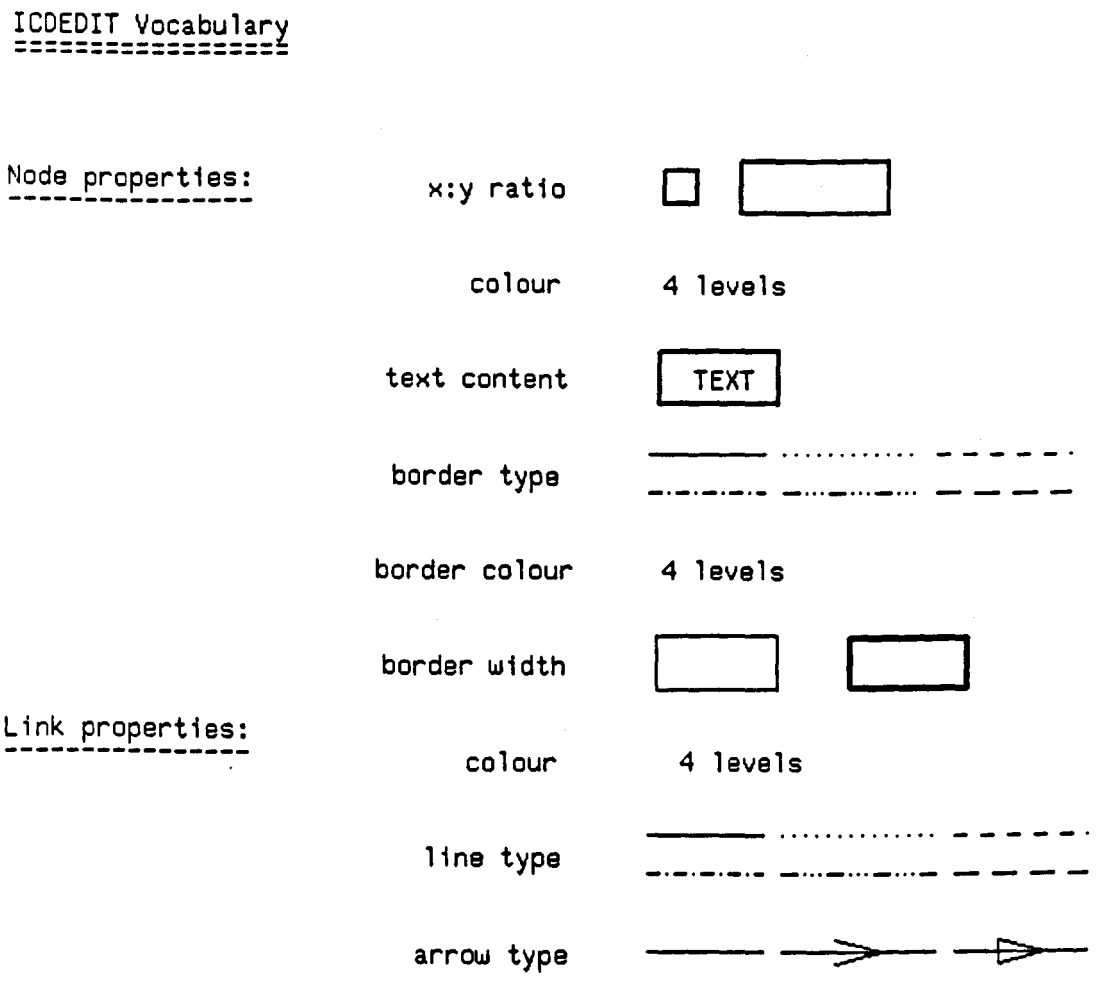


Figure D.1: ICDEDIT Vocabulary

D.3 Stage 3

D.3.1 Interview Prompt Sheet

INTRODUCTION
==========

Hello, thankyou for coming.

The aim of today's session is to see whether the representation we've developed seems as though it might be useful, and whether it is easy to use under ICDEDIT.

- ICDEDIT is a tool for manipulating 3-d ICD's of the kind we've discussed which runs on a Sun workstation
- at the moment it is just a graphical editor and is not connected to any real knowledge-based systems
- it is being used as a prototype for evaluating how useful and usable the kind of 3-d ICDs it supports are

The structure of the session will be as follows:

- first I will introduce you to the functionality of ICDEDIT by demonstrating some simple operations using a pre-prepared ICD
- you will then have as long as you like to try working with the same ICD yourself
- once you feel fairly confident with ICDEDIT, we will go through a set of simple tasks using an example of the kind of representation we developed for your knowledge-based system
- finally, I will ask you for your comments on the success or otherwise of the representation and on the tool itself

Please feel free to ask any questions or make any comments at any time.

Please do not worry about any problems you may have, they are the system's fault not yours!

TRAINING
========

Stress that this is a prototype so some functions have still not been implemented and there are still a few bugs.

General
=======

Diagrams can be edited using a combination of direct manipulation and form-filling (using the panels on the right).

Many of the operations are specified in terms of the x, y and z dimensions. The x axis runs from the left of the screen to the right, the y axis runs from bottom to top, and the z axis runs into the screen.

Mouse
=====

The mouse buttons work as follows:

- the lefthand button is used to activate:
  it is used mainly to activate functions behind buttons in the control panels

- the middle button is used to move nodes in the diagram:
- to move a node in the x-y plane, click over that node and hold down the button while moving the mouse to position the node as required;
- to move a node in the x-z plane, click over that node and hold down both the mouse button and the shift key while moving the mouse from left to right (to move the node in x) or forwards and backwards (to move the node in z)
- the middle button may also be used to create new nodes by clicking outside of any existing nodes

- the righthand button is used to select components of the diagram (see below) and also to select items from the menus hidden behind the circular icons in the control panels

Global Functions
================

The 'Set Colors' function allows you to change any of the colours defined for this representation.

- you may also use it to remind you which colour is which (as colours are referred to numerically in the control panels)

Rocking motion may be set in action using the 'Rocking period' slider:

- a rocking period of 0 means that the diagram will be static
- otherwise, the lower the period, the faster the motion will be

Viewing Functions

Clicking on the 'View' button will bring up a control panel allowing you to define various 3-d transformations on the representation, as well as setting some rocking motion parameters.

- to specify a 'Rotation' about any of the 3 axes, click on the relevant field, delete its current contents, enter a figure representing the extent of the desired rotation in terms of degrees and press return

- to specify a further 'Incremental Rotation' in addition to one already specified, click on the relevant field in the following line and enter a suitable value as above

- to specify a 'Shift' along any of the axes, enter a figure representing the extent of the desired shift in terms of pixels in the appropriate field as above

- to 'Scale' the diagram, enter the desired value in the appropriate field and press return (1.00 is the default scale, so values commonly used here are between 0.3 and 2.5 or 3.0)

- to set the amount of perspective, enter a value in the 'Eye distance' field:

- the default value for eye distance is 2000 and corresponds to a 'normal' distance between the viewer's eye and the virtual structure of 2000 pixels;

- smaller values (down to perhaps 500) can be used to exaggerate the perspective
- larger values can be used to lessen its effects, for example, a value of 99999 will effectively set the eye-point at infinity and lead to a planar projection

- parameters of the rocking motion may be set by entering values in the 'Axis depth', 'Axis angle' and 'Amplitude' fields:

    - Axis depth is specified in terms of pixels and refers to the depth of the axis about which a virtual structure is rocked relative to the screen
    - Axis angle is specified in terms of degrees and refers to the angle made by the axis about which a structure is rocked with respect to the x axis
    - Amplitude is specified in terms of degrees and refers to the amplitude of motion, that is, the amount of rotation used at the farthest point in the cycle of motion

- the 'See result' button allows you to see the result of any of the transformations you have specified

- the 'Auto shift' button allows you to request that the diagram should be placed in the centre of the panel

- the 'Auto scale' button allows you to request that the diagram should be scaled such that it just fills the panel

- the 'Set defaults' button allows you to request a combination of both of the above

It is advisable not to perform any editing operations while the View panel is activated.

- the 'Ok/Done' button allows you to close down the View panel while leaving the transformations you have just specified in place

- the 'Cancel' button allows you to close down the View panel and return to the view of the diagram you had before it was activated

Edit Functions

The 'Refresh' button allows you to clear the screen of any junk that might accumulate.

The 'Show coords' field allows you to turn display of node co-ordinates on and off.

The three 'Drag' options allow you to specify whether links should appear to follow nodes when they are dragged.

- note that this affects the speed at which nodes can be dragged

'Node' and 'Link' panels can be used to define sets of attributes which will hold for any new nodes and links specified, or to change attributes of selected components.

- yellow panels are used to specify attributes of new components
- pink panels are used to change attributes of selected components

In the 'Node' and 'Link' panels:

- the 'Content' button activates a panel allowing you to specify text which is to appear within a node

- the first 'Color' field allows you to define the colour of the text

- the 'Opaque' field allows you to specify whether a node should be opaque or transparent

- the second 'Color' field allows you to define the colour of a node's body

- the 'Border' field allows you to specify how thick the border of a node should be (in pixels)

- the third 'Color' field allows you to define the colour of a node's border

- the 'Pattern' field allows you to specify which of the six available line types should be used to draw a node's border

- the 'Size' fields allow you to specify the width and height

of a node in terms of pixels covered (by a node at depth z = 0) in the x and y directions

- the 'Posn' field allows you to:

    - specify a z position for new nodes
    - specify a position for a selected node in x, y and z

- the 'Grid' field allows you to specify increments (in pixels) in terms of which node positions should be aligned in x, y or z

- nodes may be deleted by selecting them and clicking on the 'delete' button in the selected node panel

- nodes may be hidden (so that they no longer appear on the screen and links attached to them are shown as stubs attached to the nodes at their opposite ends) by selecting them and clicking on the 'hide' button in the selected node panel

- nodes may subsequently be revealed by selecting the stub of a link to which it is attached and activating the 'Reveal node' button in the selected link panel

- links may be added by selecting the two nodes they are to connect (in order if the link is to be directional) and activating either the 'Side-side' or the 'Top-bottom' buttons in the selected node panel

    - using the 'Side-side' button causes the link to be drawn between the vertical sides of the two nodes
    - using the 'Top-bottom' button causes the link to be drawn between the horizontal sides of the two nodes

- the 'Line' field (in the 'Link' panels) allows you to specify which of the six available line types should be used to draw a link

- the 'Arrow' field allows you to specify whether a link should be drawn with an open arrow head or a closed one

- the 'Arrow breadth' field allows you to specify the width of the arrow head in terms of pixels

    - note that a width of zero signifies that no

arrow head is to be drawn

- the 'Color' field allows you to specify what colour should be used to draw a link

- a link may be deleted by selecting it and activating the 'delete' button in the selected link panel

Have you any questions?

Please try working with the tool using this diagram until you feel confident that you will be able to carry out a few simple tasks using a diagram of the kind we developed in stage 2.

TASK SESSION
==========

We're now going to try performing some simple tasks using the representation we developed.

- I have a list of tasks for you to perform
- I will read them out one by one and I'd like you to try and do them, talking me through what you do as you go
- please tell me about any problems you have with the software, or in particular with the representation
- please tell me when you are happy that you have completed the task
- I will remain silent while you perform individual tasks, except to remind you to tell me what you are doing

Do you have any questions?

Here is a key to remind you of the representational scheme we developed:

- (explain meaning of the key)

Now I will load in a simple diagram of the type described for you to work with

- (load abstract diagram and run through tasks)

Thankyou, that's the end of the tasks.

DEBRIEFING
==========

The Representation
==================

How do you think the representation we developed has turned out?

- has anything turned out differently to the way you expected?
- is anything missing?
- does anything seem to represented in a misleading way?
- is there anything you'd like to change now you've seen the diagrams?

Use of 3-d
==========

What did you think about using 3-d diagrams?

- were you happy with the way depth information was used in the representation developed?
  - was it as you expected?
  - how do you think it compares with 2-d?
- how useful do you think 3-d diagrams might be for knowledge engineers?
  - how easy did you find 3-d diagrams to work with?
  - how easy did you find it to tell which nodes were further towards the back of the diagram than others?
  - what do you think the advantages of using 3-d diagrams of this kind might be?
  - what do you think the disadvantages might be?

The Tasks
=========

What kind of knowledge engineering tasks do you think this kind of diagram might best be used to support?

- how might it be used in this way?
- what kind of extra functionality do you think might be needed?

The Tool
========

How did you find using the tool?

- how easy did you find it?
- did you like it / enjoy it / find it frustrating?

Are there any other general comments you'd like to make about the tool itself?

END
===

D.3.2 Generic Task Taxonomy

```
1. Finding Elements
===================

Find elements of the diagram described in terms of:

        - graphical criteria:

                - single nodes described in terms of:
                        - node content
                        - node appearance
                                - node body colour
                                - node border colour
                                - node label colour
                                - node border type
                                - node border width
                        - node position
                                - absolute (in terms of co-ordinates)
                                - relative to other nodes

                - single links described in terms of:
                        - link appearance
                                - link line style
                                - link colour
                                - link arrow head type
                        - link position
                                - absolute (in terms of co-ordinates
                                  of end points)

                - groups of nodes described in terms of:
                        - node appearance
                                - node body colour
                                - node border colour
                                - node label colour
                                - node border type
                                - node border width
                        - node position
                                - absolute (in terms of co-ordinates)
                                - relative to other nodes

                - groups of links described in terms of:
                        - link appearance
                                - link line style
                                - link colour
                                - link arrow head type
                        - link position
                                - absolute (in terms of co-ordinates)
```

- language criteria:

    - single entity described in terms of:
        - appearance
        - relations with other entities
        - position
            - absolute (in terms of co-ordinates)
            - relative to other entities

    - groups of entities described in terms of:
        - appearance
        - relations with other entities
        - position
            - absolute (in terms of co-ordinates)
            - relative to other entities

    - single relation described in terms of:
        - appearance
        - position with respect to entities

    - groups of relations described in terms of:
        - appearance
        - positions with respect to entities

2. Deleting Elements

Delete elements described in terms of content or appearance (other criteria already tested under 1):

- single nodes

- single links

3. Editing Existing Elements

Edit components of the diagram in terms of:

- graphical criteria:

- edit single nodes:
    - node content
    - node appearance
        - node body colour
        - node border colour
        - node label colour
        - node border type
        - node border width
    - node position
        - absolute (in terms of co-ordinates)
        - relative to other nodes

- edit single links:
    - link appearance
        - link line style
        - link colour
        - link arrow head type
    - link position
        - absolute (in terms of co-ordinates of end points)

- language criteria:

    - edit single entity:
        - appearance
        - relations with other entities
        - position
            - absolute (in terms of co-ordinates)
            - relative to other entities

    - edit single relation:
        - appearance
        - position with respect to entities

4. Adding New Elements

Add to the diagram new components described in terms of:

- graphical criteria

- language criteria

D.3.3 Graphical Representations

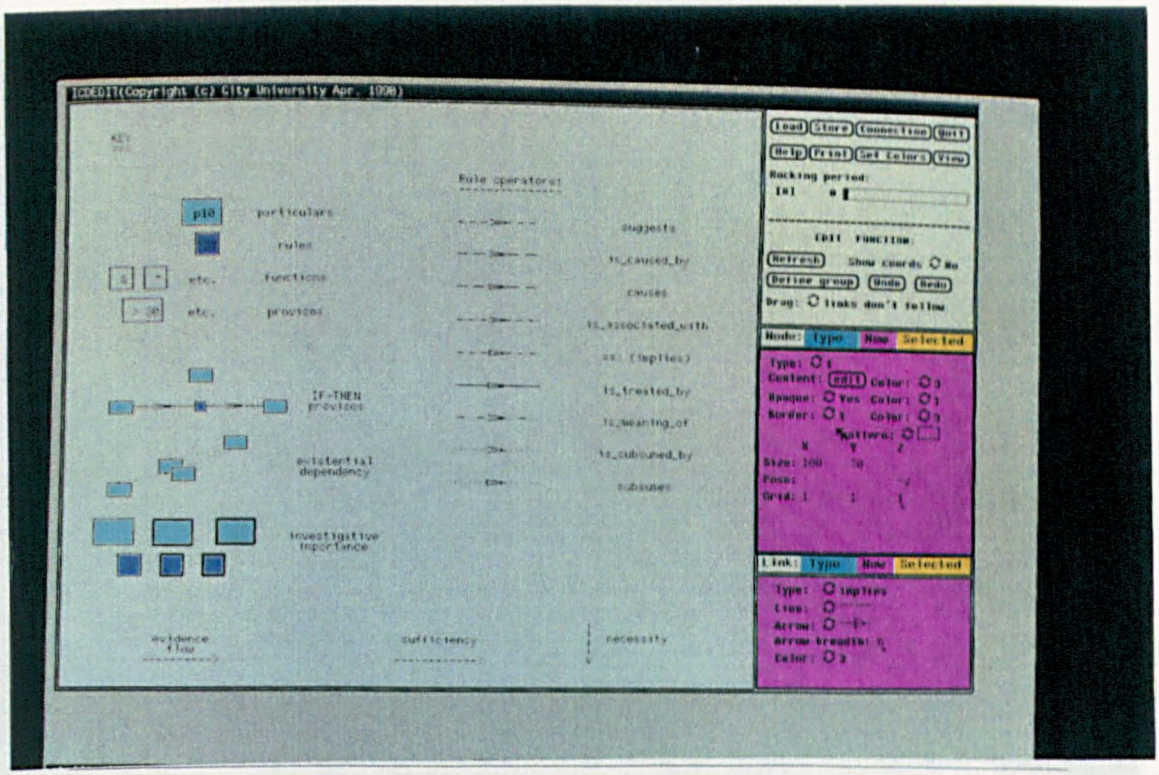


Figure D.2: Key Describing the Visual Language Used in Case Study 1

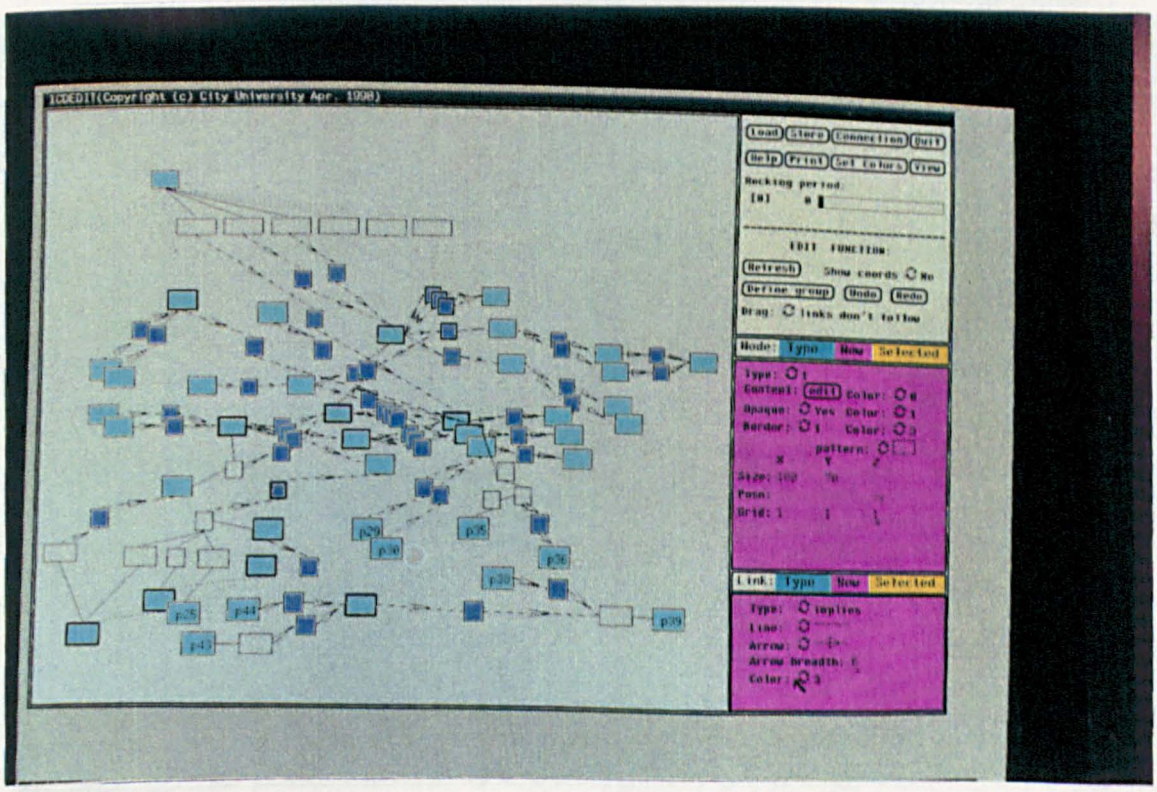


Figure D.3: Actual 3-d ICD Representation Used in Case Study 1

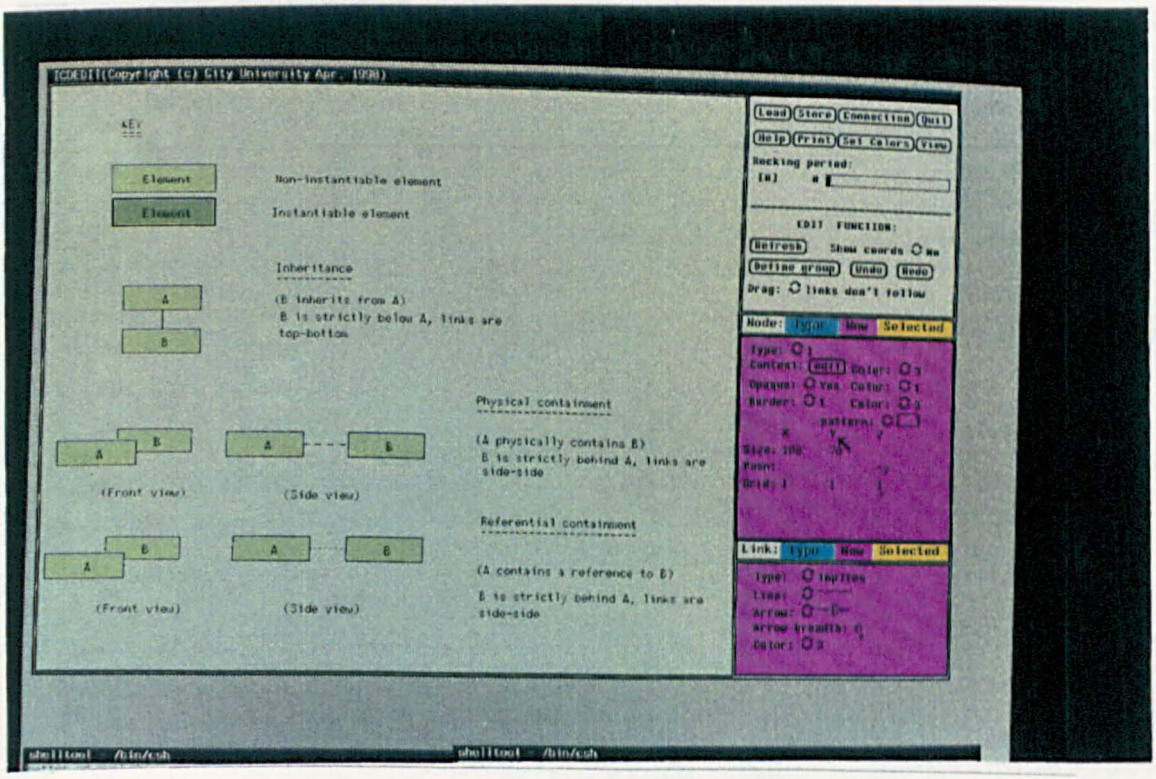


Figure D.4: Key Describing the Visual Language Used in Case Study 2

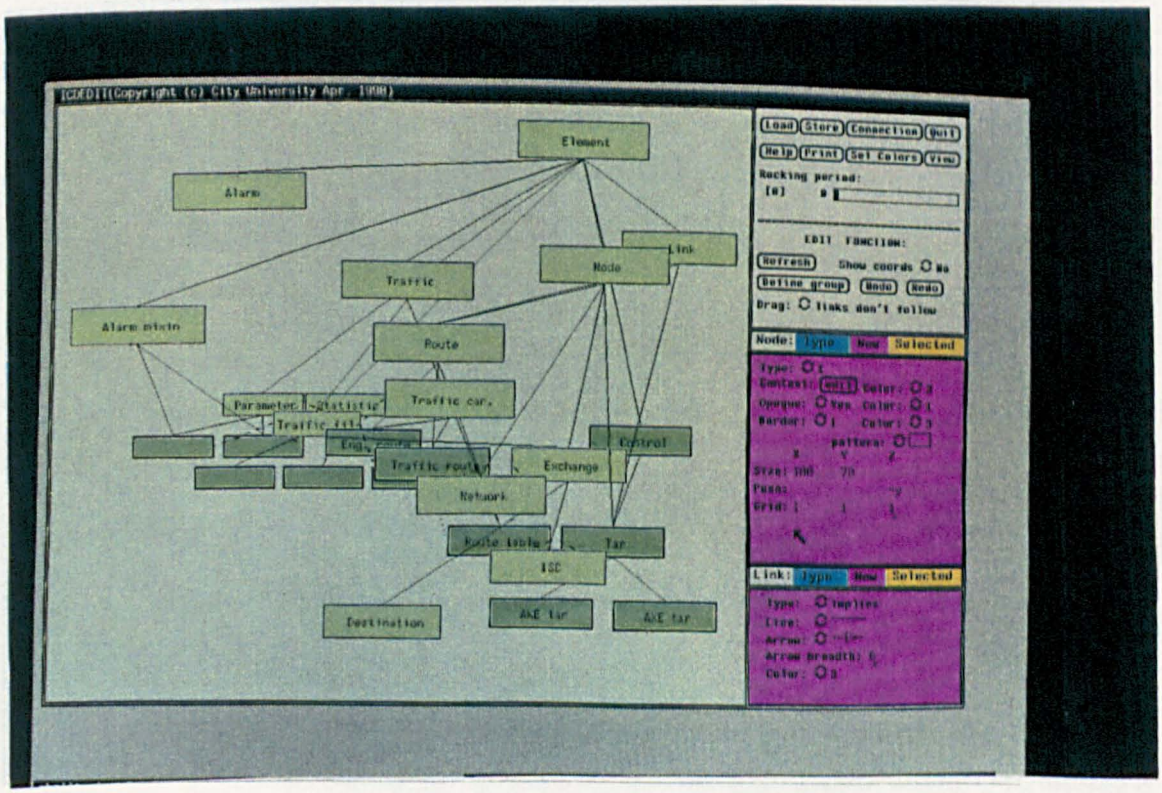


Figure D.5: Actual 3-d ICD Representation Used in Case Study 2

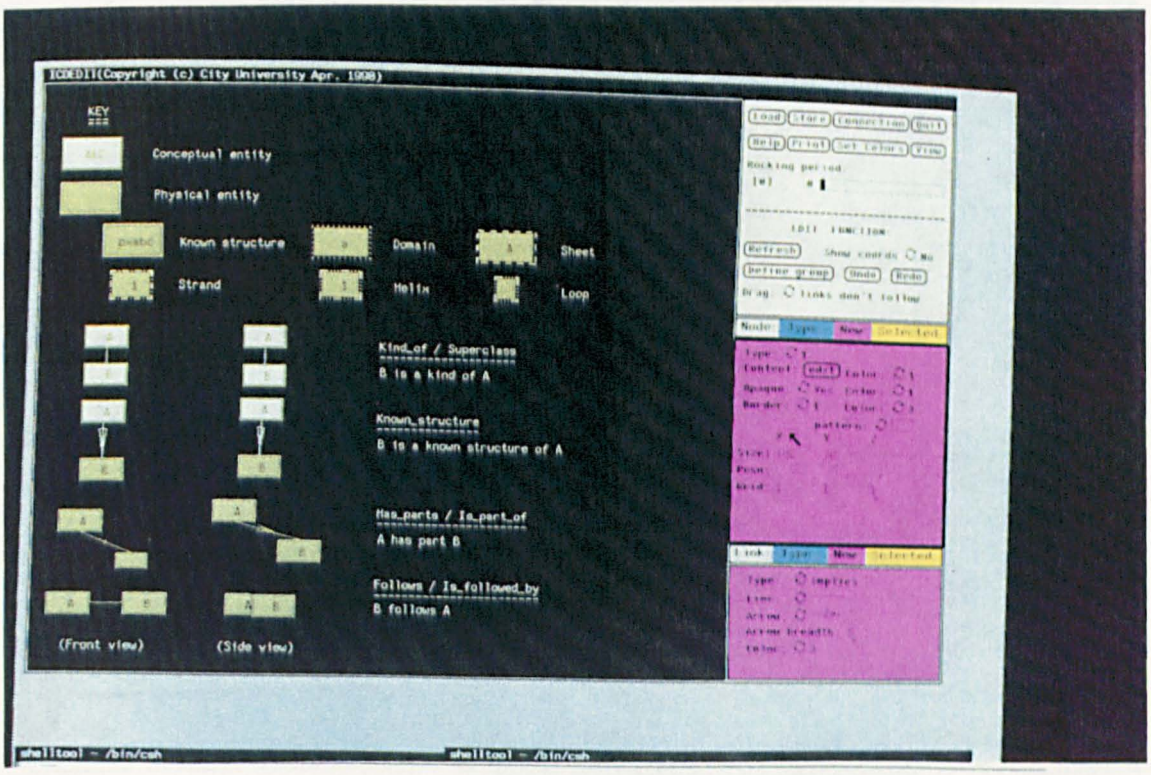


Figure D.6: Key Describing the Visual Language Used in Case Study 3

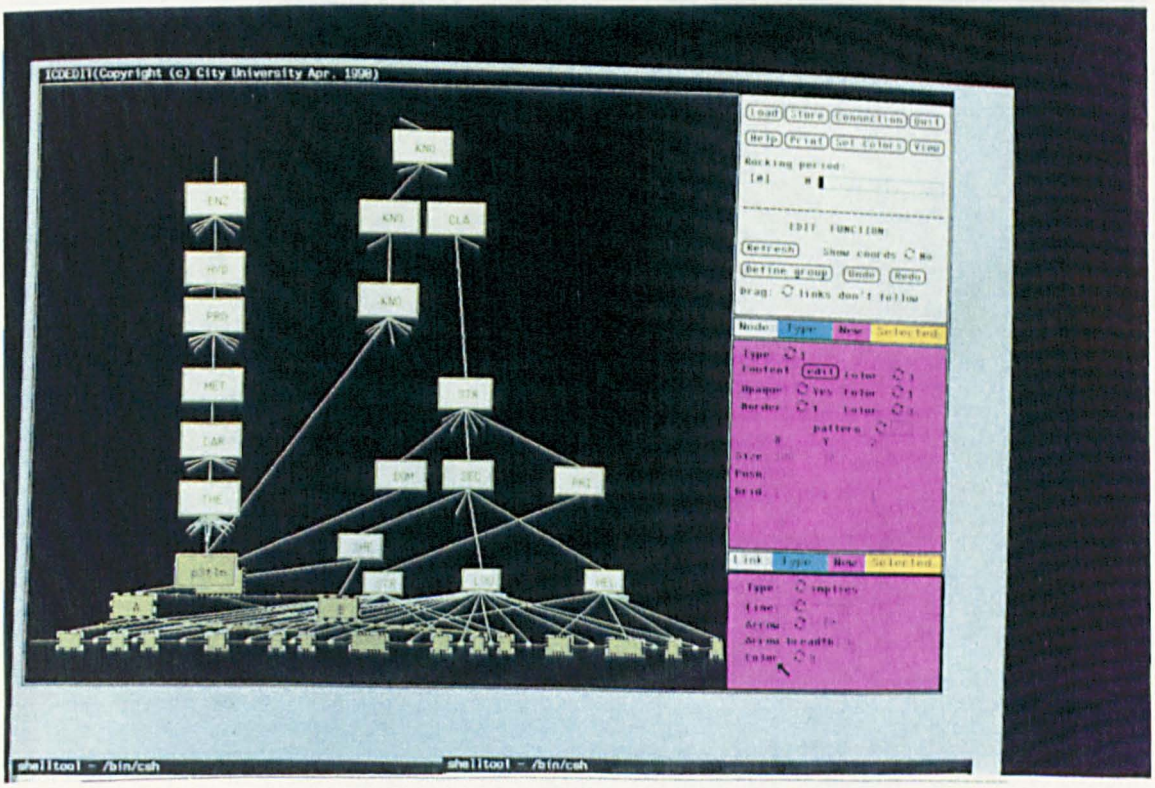


Figure D.7: Actual 3-d ICD Representation Used in Case Study 3

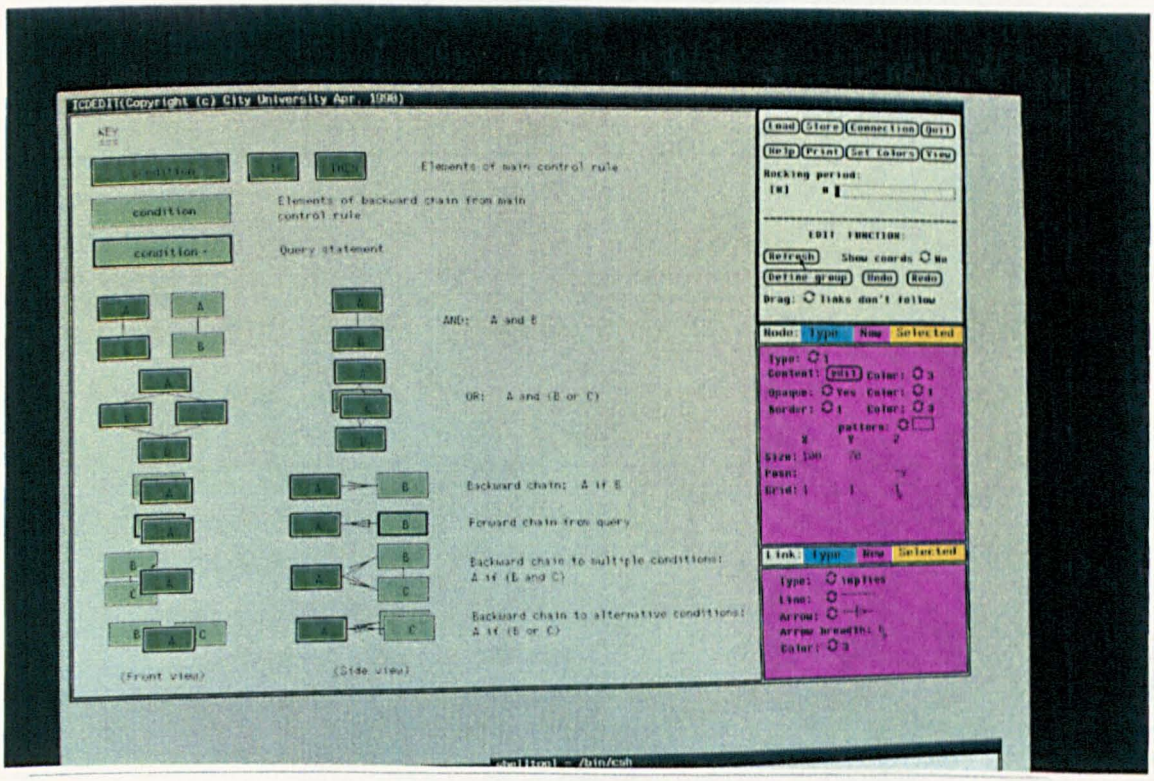


Figure D.8: Key Describing the Visual Language Used in Case Study 4

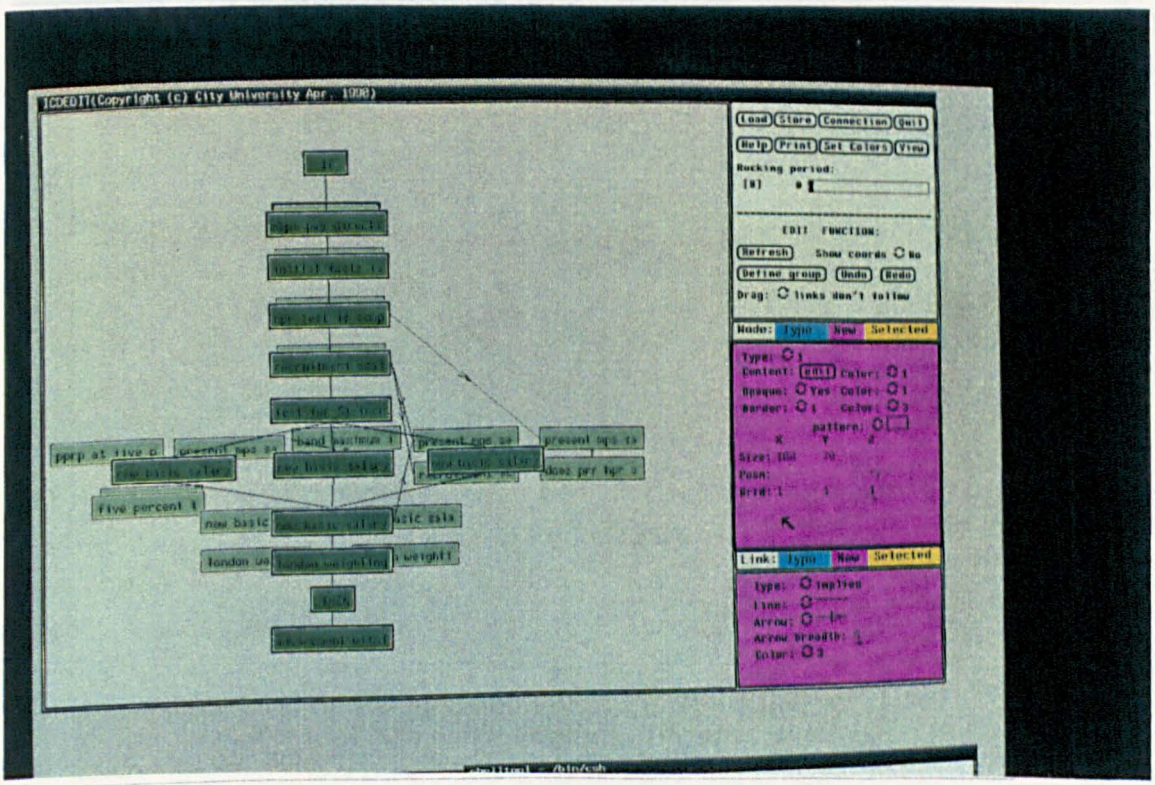


Figure D.9: Actual 3-d ICD Representation Used in Case Study 4

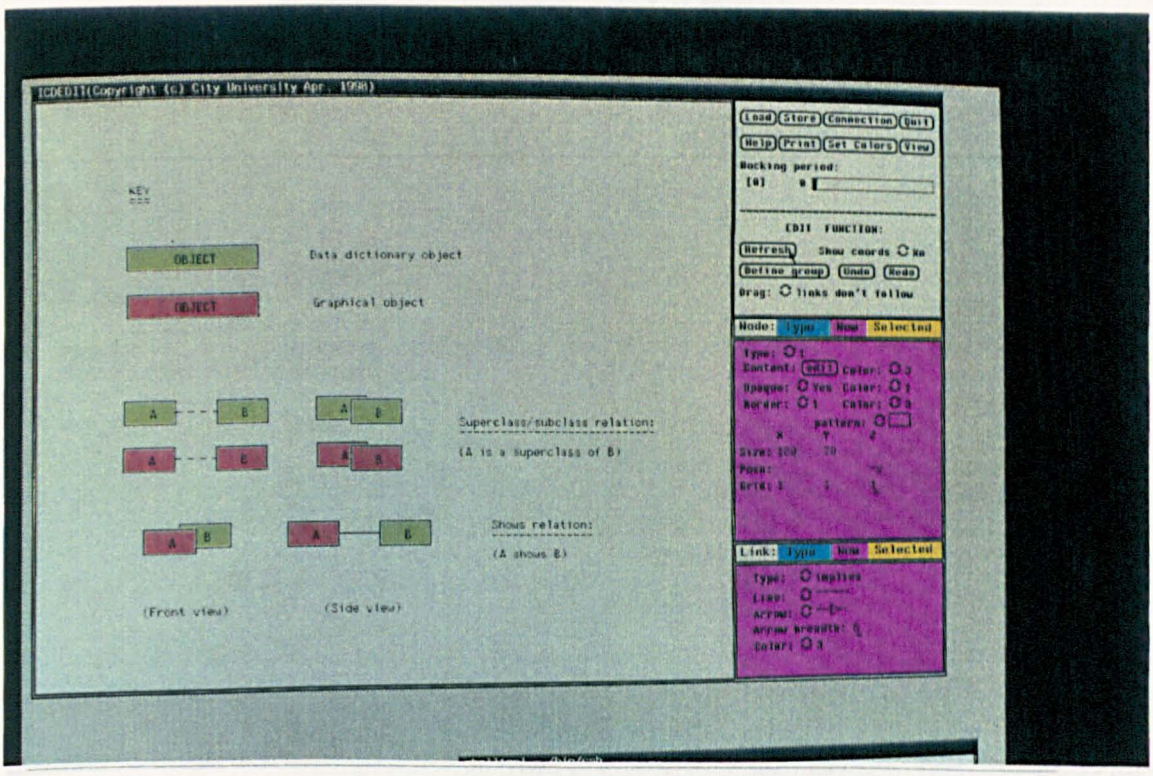


Figure D.10: Key Describing the Visual Language Used in Case Study 5

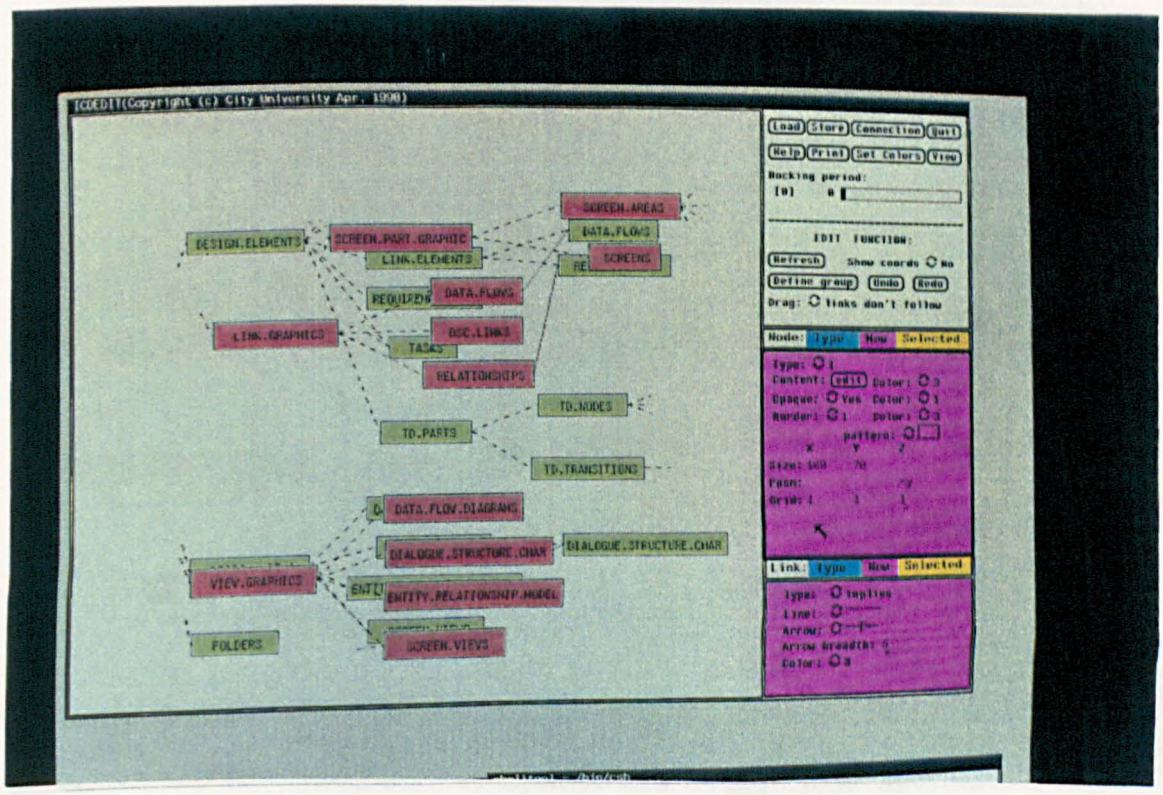


Figure D.11: Actual 3-d ICD Representation Used in Case Study 5

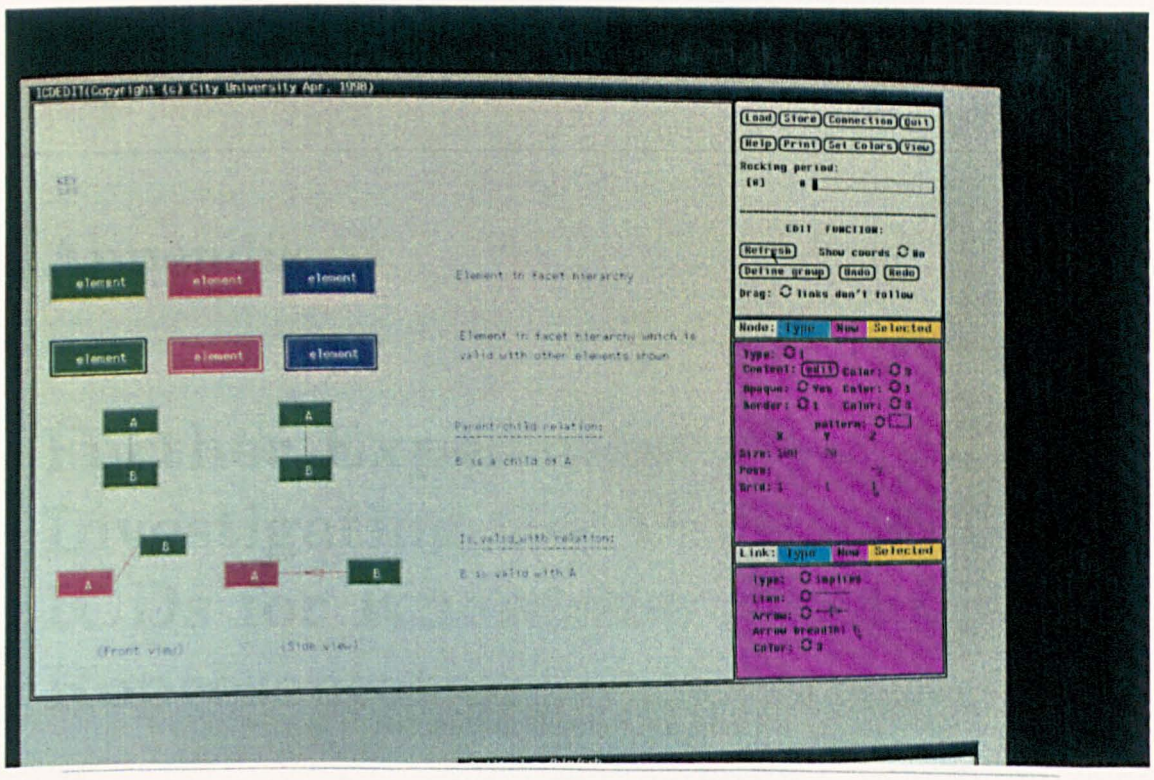


Figure D.12: Key Describing the Visual Language Used in Case Study 6

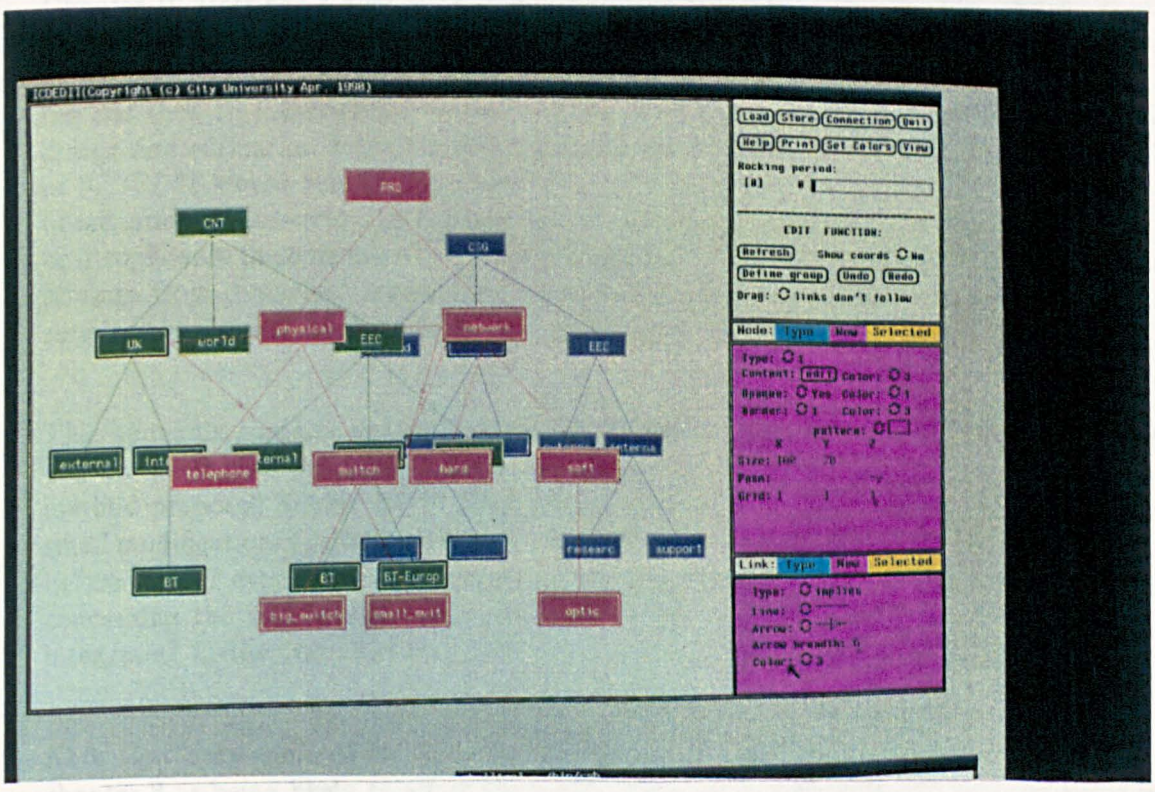


Figure D.13: Actual 3-d ICD Representation Used in Case Study 6

Appendix E

Further Experiments for Investigating the Utility of 3-d ICDs for Knowledge Structure Representation

Chapter 10 described a pilot experiment which could be used to investigate claims about the utility of 3-d ICDs for knowledge structure representation initially made as a result of the case studies. The difficulties of using controlled laboratory experiments in carrying out such an investigation were described in chapter 10. It was concluded that the design and evaluation activities appropriate to the relatively early stage of development of ICDEDIT would best be supported by data from empirical studies based on the use of scenarios. If, however, the findings of scenario-based studies were unclear in any way, or if confidence in some aspect of these findings was judged to be particularly important, findings from controlled experiments using larger numbers of subjects could be used to supplement those from the studies.

This appendix contains descriptions of several experiments which could be used in this way to investigate particular aspects of the question of utility. It is assumed that the method proposed for use in the pilot study would be broadly successful and that only small modifications (perhaps to the nature of the stimuli or tasks used) would be necessary before further experiments of a similar kind could be conducted. Suggestions are made concerning the way in which the effects of particular factors could be investigated by integrating further variables into the framework used in the pilot study.

Note that only some of the aspects of users, environments and systems identified in chapter 8 as being likely to affect the utility of 3-d ICD interfaces will be considered in this way. Formal experimental investigation of other factors identified in that chapter is felt to be less appropriate at present.

E.1 Effects of User Characteristics

E.1.1 Visualisation Skills

The effect of user visualisation skills on performance of problem-solving tasks using 3-d, 2-d and textual representations of the relevant information could be investigated using the design proposed for use in the pilot study.

Before participating in the experiment, subjects would be asked to take a test of visualisation skill such as that proposed by Vandeberg and Kuse [147]. Subjects' scores in this test would then be used as a variable in the analysis of the results. This analysis would aim to determine whether a subject's test score was a good predictor of performance in tasks using any of the representation formats of interest.

E.1.2 Experience with Graphical Representations

The effect of previous experience with the use of two-dimensional graphical representations on ability to perform problem-solving tasks with various representations could be determined by studying two sets of subjects, one with extensive experience of using graphics, the other with little or none.

This could be achieved using a modified version of the above design in which half the subjects in each group were experienced and half were inexperienced. The total number of subjects used would ideally need to be increased in order that there should be sufficient data on each condition: it is suggested that 36 subjects per group would be a reasonable number with 18 in each group being experienced and 18 inexperienced. Note that order of presentation of stimuli for use in the experimental tasks would need to be balanced for each subgroup so that 6 out of 18 would see the stimuli in order ABC, 6 in order BAC and 6 in order CBA.

Degree of experience could be included in the analysis of the results as a categorical variable.

Note that one problem with this kind of experiment would be the difficulty of categorising subjects according to degree of experience, since there is no obvious measure which could easily be applied to determine which category particular subjects would belong in.

E.1.3 Experience with Knowledge-Based Systems

The effect of experience with knowledge-based systems on subjects' ability to perform problem-solving tasks with various representations of relevant knowledge structures could be investigated using a design the same as that described in the previous section.

Once again, two sets of subjects would need to be studied, one with extensive experience of using knowledge-based systems, the other with little or none. These subjects would need to be evenly distributed across groups and an appropriate categorical variable would need to be used in the analysis of the results.

There is again likely to be a problem with attempting to classify subjects according to their degree of experience.

E.2 Effects of Environmental Factors

E.2.1 Knowledge Representation Formalism

The effect of the underlying knowledge representation formalism on the relative utility of two- and three-dimensional representations in simple problem-solving tasks could also be investigated using an experiment similar to those described above.

Further stimuli would need to be developed to give a total of, say, 3 sets of representations of parts of rule-based systems and 3 of frame-based systems. Each subject would see representations in an appropriate form (3-d, 2-d or textual) of each of the 6 structures. Stimuli could be presented to each subject in a different random order. Numbers of tasks performed using each stimulus could be reduced in order to minimise the number of tasks performed by each subject. For example, if each subject performed 6 tasks per stimulus rather than 10, the total number of tasks performed by each subject would in this case be 41.

Data from this experiment could again be analysed using an analysis of variance with subject group and knowledge structure type (*ie* rule-based or frame-based) as the main variables of interest. Knowledge structure or stimulus instance (*ie* whether variant 1, 2 or 3 of either rule- or frame-based stimuli) and task number (1 - 6) should again be included in the analysis to remove unwanted variance.

E.3 Effects of System Characteristics

E.3.1 System Size

The effect of system size on the relative utility of two- and three-dimensional representations could be investigated using a design exactly analogous to that described in section E.2.1.

This experiment could again use variants on the representations developed during the case studies and used in the pilot experiment described above. In this case, 2 versions of each representation would be needed: one of each type representing a large component of a knowledge structure (A1, B1 and C1) and one representing a small component (A2, B2 and C2). Stimuli would need to be presented in a pseudo-random order which ensured that representations of large and small versions of the same knowledge structure were not presented consecutively.

Analysis would be carried out as described in section E.2.1.

Note that the identification of a metric by which knowledge structures of different kinds could be reliably judged to be 'big' or 'small' would be highly problematic. For the purposes of this experiment, however, some function of the number of nodes and links used in a graphical representation of such a structure would serve as a reasonable measure.

Note also that the degree to which 'size' of a knowledge structure determines the relative utility of 2-d and 3-d ICD interfaces is likely to depend strongly on the efficacy of the interactive functionality provided in each case. An experiment such as that described above would not be able to take account of this dependency and the significance of its results would therefore need to be judged with care.

E.3.2 Number of Different Relations

An investigation of the effect of the number of different relations on the relative utility of 3-d and 2-d representations of a knowledge structure would require the development of a number of variants of representations using the same underlying scheme.

It is suggested that two of the representations developed during the case studies should be picked for development, those developed for studies 3 and 4 being the most obvious candidates as these initially included representations for 4 and 6 different relations respectively, at least one of these being represented in depth in each case.

Three versions of each type of stimulus could be developed, one of each (A1 and B1, say) representing only 2 of the relevant relations, one of each (A2 and B2) representing 3 and the others (A3 and B3) representing 4. Care would need to be taken to ensure that the stimuli were as similar as possible in other respects: for example, an attempt would need to be made to ensure that all stimuli contained approximately the same number of nodes and links.

The design of an experiment investigating the use of such stimuli would be as described in sections E.2.1 and E.3.1, with stimuli again being presented in a pseudo-random order such that variants on the same underlying representation scheme (*eg* A1, A2 and A3) would not be presented consecutively.

Analysis would also be carried out as in E.2.1.

E.3.3 Number of Relationships per Entity

Finally, it can be seen how a similar experiment could be used to investigate the effect of the number of relationships per entity on the relative effectiveness of two- and three-dimensional representations of knowledge structures in supporting simple problem-solving tasks.

The design for such an experiment would be exactly analogous to that described in the previous section with variants on a particular stimulus type (*eg* A1, A2 and A3) in this case constructed so that each entity was shown related to other entities in, say, 3, 5 and 7 ways. In this case, care should again be taken to ensure that the total number of nodes used in each stimulus is the same and that the number of different kinds of relation displayed is constant.

The same form of analysis would again be appropriate.

Appendix F

Introduction to the Z Notation

The Z notation is described in [134] and is in common usage in software engineering. A formal definition of the notation will therefore not be given here. The following is intended as a brief informal introduction to allow readers unfamiliar with the notation to understand the specifications presented elsewhere in this thesis. A fuller introduction to Z can be found in [107].

A specification in the Z notation is begun by defining any *basic types* which will be needed over and above those provided by mathematical theory (such as $\mathbb{Z}$, the set of integers, and $\mathbb{N}$, the set of natural numbers). Basic types may simply be named:

$$[BASIC\ TYPE]$$

or may be defined by describing a set of values within which the value of a variable of that type must fall:

$$Basic\ Type ::= value1 \mid \ldots \mid valuen$$

Groups of properties relating to a particular concept may be drawn together in a *schema*:

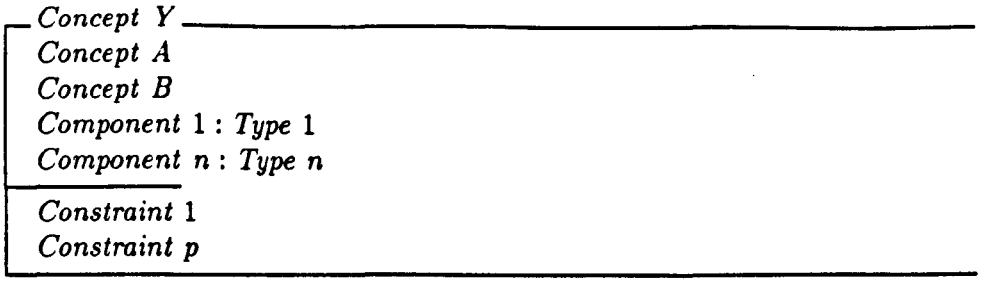
Concept X

Component 1 : *Type* 1
Component n : *Type* n

Constraint 1
Constraint p

where expressions in the top part of the schema describe the *components* of that concept and the *types* of those components (drawing where necessary on the basic types previously defined) and expressions in the bottom part describe *constraints* on those components.

In specifying a new concept, we may make use of other concepts already specified by *importing* existing schemas into the definition of a new one, as for example:



The top part of the new schema is thought of as containing the contents of the top parts of any imported schemas as well as what is explicitly specified. In the same way, the bottom part is seen as containing the contents of the bottom parts of any imported schemas as well as any new constraints explicitly specified. This combination of schemas must not lead to any type conflicts or contradictions in the constraints.

Dynamic aspects of a system such as operations and changes of state may also be specified. If schemas are taken to represent components of some universal state in which we are interested, then a change in that component of state is signified using the Δ*schema* notation. The fact that there is no change in a particular component of state can be denoted using the Ξ*schema* notation.

In order to define the way in which particular aspects of a component of state differ before and after a change in that component, we use the $'$ notation. An identifier *identifier* used to refer to a particular aspect of the component of state before change can be used in a modified form (*identifier*$'$) to refer to the same aspect after the relevant change.

To define operations with inputs and outputs, we may use identifiers annotated in a different way. The form *identifier*? is used to denote an input, and the form *identifier*! denotes an output.

F.1 Glossary of Symbols

Apart from the notation of formal logic and set theory, the specifications described in this thesis use the following symbols:

$\bullet$	such that
$\mid$	where
$\mathbb{P}$	power set of
$\mathbb{F}$	finite set of
$\mathbb{Z}$	the set of integers
$\mathbb{N}$	the set of natural numbers
$\mathbb{N}_1$	the set of natural numbers excluding 0
$\pfun$	a mapping which is a partial function
$\mapsto$	a mapping between individual elements
$\ndres$	domain restriction: element on the left is omitted from the domain of the function on the right
$\Delta schema$	signifies a change in the state denoted by *schema*
$\Xi schema$	signifies that there is no change in the state denoted by *schema*
variable'	used to define the nature of a variable after a specified change in state
variable?	used in descriptions of operations to signify an input variable
variable!	used in descriptions of operations to signify an output variable

Appendix G

Formal Specifications of 3-d ICD Representations

G.1 Introduction

This appendix presents specifications of 5 of the 6 languages for visual knowledge representation developed during the case studies described in chapters 7, 8 and 9. One of the languages has already been described in chapter 11.

These specifications are presented as worked examples analogous to those in chapter 11. Each of the examples begins with a brief description of the knowledge based system represented. This is followed by a description of requirements for graphical representation elicited from subjects during the course of the study. A formal specification of the design developed is then given.

Examples shown here are from studies 1,3,4,5 and 6.

G.2 Case Study 1

G.2.1 Knowledge-Based System Profile

Hardware platform: mainframe

Software: Prolog

Architecture: rule-based

Size: less than 200 rules

General Description:

The system was a prototype medical diagnostic system aimed at general practitioners. Important entities to be represented included particulars, or fragments of knowledge, and rules which related them in various ways. Provisos on some of the particulars and logical functions which related particulars within rules were also represented. The investigative importance of individual particulars and rules at a given (imaginary) point in time is also represented.

G.2.2 Representation Requirements

G.2.2.1 Vocabulary

Types of entity to be represented:

- particulars
- rules
- provisos on rules or particulars
- logical operators acting on particulars

The relationships to be represented were mainly those embodied by the rules in the system. Other relations represented were those linking provisos with particulars and conditions with rules. Existential dependency relations were also represented.

Other requirements:

- nodes representing rules should be characteristically different from those representing particulars
- investigative importance of particulars or rules should be represented by node border thickness
- links entering and leaving rule nodes should be in line or parallel

G.2.2.2 Layout

- layout can be fairly ad hoc
- in general, evidence should flow from left to right in a diagram

- nodes should be placed in depth in such a way as to minimise crossings of links

G.2.3 Representation Specification

G.2.3.1 Vocabulary

Three different colours of the same hue were chosen for use in representing the various types of system entity. A dark colour which would show up well against either of these was chosen for use in labelling nodes and drawing links. The colour used for the background was the lightest of the three colours chosen for representation of nodes.

$$L1DarkNodeCol, L1MedNodeCol, L1LightNodeCol, L1OutlineCol : COLOUR$$

Nodes were mainly of one of two heights and widths.

$$L1NodeHeightA, L1NodeHeightB, L1NodeWidth : \mathbb{Z}_1$$

$$L1NodeWidth > L1NodeHeightA$$
$$L1NodeHeightA > L1NodeHeightB$$

All nodes had certain common properties which we may define as follows.

L1CommonNodeType

$$VisualNodeType$$

$$BorderColour = L1OutlineCol$$
$$LabelColour = L1OutlineCol$$
$$BorderStyle = solid$$

The use of border thickness to represent investigative importance gives us:

L1Important

$$L1CommonNodeType$$

$$BorderWidth = thick$$

L1SemiImportant

$$L1CommonNodeType$$

$$BorderWidth = medium$$

L1Unimportant

$$L1CommonNodeType$$

$$BorderWidth = thin$$

Nodes representing particulars can be defined as follows:

L1Particular

L1CommonNodeType

Height = *L1NodeHeightA*
Width = *L1NodeWidth*
BodyColour = *L1MedNodeCol*
LabelStyle = *misc*

Note that nodes representing particulars were labelled with the number of the particular eg p16, p23.

Combining this definition with that of importance, we have:

L1PartA

L1Important
L1Particular

L1PartB

L1SemiImportant
L1Particular

L1PartC

L1Unimportant
L1Particular

Similarly, nodes representing rules can be described as follows:

L1Rule

L1CommonNodeType

Height = *NodeHeightB*
Width = *Height*
BodyColour = *DarkNodeCol*
LabelStyle = *numeric*

This yields:

L1RuleA

L1Rule
L1Important

L1RuleB

L1Rule
L1SemiImportant

L1RuleC

L1Rule
L1Unimportant

The remaining entities are represented as follows:

L1Proviso

L1CommonNodeType

BodyColour = *L1LightNodeCol*
BorderWidth = *thin*
LabelStyle = *lower*

L1LogicOp

L1CommonNodeType

Height = *L1NodeHeightB*
Width = *Height*
BodyColour = *L1LightNodeCol*
BorderWidth = *thin*
LabelStyle = *misc*

Note that nodes representing logical operators are labelled with the appropriate logical symbol where possible.

The only property shared by all links is that of colour:

L1CommonLinkType

VisualLinkType

Colour = *L1OutlineCol*

Links representing the relationships embodied by rules can be defined as follows. (Note that in several cases, the type of join, that is whether a link runs from top to bottom or side to side, is not constrained.)

L1IsTreatedBy

L1CommonLinkType

Line = *solid*
Arrow = *closed*

L1Suggests

L1CommonLinkType

Line = *short dashed*
Arrow = *open*

L1IsMeaningOf
L1CommonLinkType

Line = pattern2
Arrow = open

Links representing a causal relation are shown as long dashed lines:

L1IsCausedBy
L1CommonLinkType

Line = long dashed
Arrow = open

L1Causes
L1CommonLinkType

Line = long dashed
Arrow = closed

Links representing subsumption are shown as dotted lines:

L1IsSubsumedBy
L1CommonLinkType

Line = dotted
Join = top bottom
Arrow = open

L1Subsumes
L1CommonLinkType

Line = dotted
Join = top bottom
Arrow = closed

Links representing association or implication are shown as lines of pattern type 1:

L1IsAssociatedWith
L1CommonLinkType

Line = pattern1
Arrow = open

L1Implies
L1CommonLinkType

Line = pattern1
Arrow = closed

Finally, links representing other kinds of relationships are defined as follows:

L1DependsOn

L1CommonLinkType

Line = dotted
Join = top bottom
Arrow = noarrow

L1Conditional

L1CommonLinkType

Line = dotted
Join = top bottom
Arrow = noarrow

(Note that these last two link types are distinguishable only by the types of node they connect - see below.)

L1IsProvisoOn

L1CommonLinkType

Line = solid
Arrow = noarrow

G.2.3.2 Language

The complete visual language may now be defined as follows:

Language1

$VisualLanguage$

$RuleRelations : \mathbb{F}\ VisualLinkType$

$RuleRelations = \{L1IsTreatedBy, L1Suggests, L1IsMeaningOf, L1IsCausedBy, L1Causes, L1IsSubsumedBy, L1Subsumes, L1IsAssociatedWith, L1Implies\}$

$\forall d \in Diagrams \bullet$

$d.Vocabulary.NodeTypes = \{L1PartA, L1PartB, L1PartC, L1RuleA, L1RuleB, L1RuleC, L1Proviso, L1LogOp\} \land$

$d.Vocabulary.LinkTypes = RuleRelations \cup \{L1DependsOn, L1Conditional, L1IsProvisoOn\} \land$

$d.Vocabulary.Colours = \{L1DarkNodeCol, L1MedNodeCol, L1LightNodeCol, L1OutlineCol\}$

$\forall d \in Diagrams \bullet$

$(\forall l : \text{ran } LinkRep \mid l.VisType \in L1RuleRelations \bullet$

$\exists n1, n2 : VisualNODE \mid$

$n1.VisType \in \{L1RuleA, L1RuleB, L1RuleC\} \land$

$n2.VisType \in \{L1PartA, L1PartB, L1PartC, L1Proviso, L1LogOp\} \bullet$

$(l.Origin = n1 \land l.Dest = n2) \lor$

$(l.Origin = n2 \land l.Dest = n1)) \land$

$(\forall l : \text{ran } LinkRep \mid l.VisType = L1DependsOn \bullet$

$(l.Origin).VisType \in \{L1PartA, L1PartB, L1PartC\} \land$

$(l.Dest).VisType \in \{L1PartA, L1PartB, L1PartC\}) \land$

$(\forall l : \text{ran } LinkRep \mid l.VisType = L1Conditional \bullet$

$\exists n1, n2 : VisualNODE \mid$

$n1.VisType \in \{L1RuleA, L1RuleB, L1RuleC\} \land$

$n2.VisType \in \{L1PartA, L1PartB, L1PartC\} \bullet$

$(l.Origin = n2 \lor (l.Origin).VisType = L1Proviso) \land$

$(l.Dest = n1) \lor$

$(l.Origin = n2 \land (l.Dest).VisType = L1Proviso)) \land$

$(\forall l : \text{ran } LinkRep \mid l.VisType = L1IsProvisoOn \bullet$

$\exists n1, n2 : VisualNODE \mid$

$n1.VisType \in \{L1RuleA, L1RuleB, L1RuleC\} \land$

$n2.VisType \in \{L1PartA, L1PartB, L1PartC\} \bullet$

$(l.Origin = n1 \land (l.Dest).VisType = L1Proviso) \lor$

$((l.Origin).VisType = L1Proviso \land$

$(l.Dest = n2 \lor (l.Dest).VisType = L1LogOp)))$

$\forall d \in Diagrams \bullet$

$(\forall l : Link \mid d.LinkRep(l).VisType = L1IsSubsumedBy \bullet$

$NodePos(l.Origin).X < NodePos(l.Dest).X \land$

$NodePos(l.Origin).Y < NodePos(l.Dest).Y) \land$

$(\forall l : Link \mid d.LinkRep(l).VisType = L1Subsumes \bullet$

$NodePos(l.Origin).X < NodePos(l.Dest).X \land$

$NodePos(l.Origin).Y > NodePos(l.Dest).Y) \land$

$(\forall l : Link \mid d.LinkRep(l).VisType = L1DependsOn \bullet$

$NodePos(l.Origin).X < NodePos(l.Dest).X \land$

$NodePos(l.Origin).Y < NodePos(l.Dest).Y) \land$

$(\forall l : Link \mid LinkRep(l).VisType = L1IsSubsumedBy \bullet$

$NodePos(l.Origin).Y > NodePos(l.Dest).Y)$

G.3 Case Study 3

G.3.1 Knowledge-Based System Profile

Hardware platform: general purpose workstation

Software: C and Prolog-based toolset developed in house, including textual and specialised 2-d graphical browsers

Architecture: frame-based knowledge representation with rule-based transaction manager

Size: 60,000 Prolog clauses

General Description:

The system as a whole was intended to provide intelligent assistance in the analysis and prediction of protein structures. The aim of the graphical representation was to illustrate how objects in a small fragment of the knowledge base were related by three different types of relationship. Each of the three relationships had the property of transitive closure. They were also mutually orthogonal in the sense that the position of an element with respect to one relationship was logically independent from its position with respect to either of the others.

Important entities to be represented could be divided into the two main categories of conceptual and physical entities. Physical entities corresponded to individual protein structures and their components. Protein structures can be described in terms of their constituent domains. Domains are made up of sheets, which can be further decomposed into strands, helices and loops.

G.3.2 Representation Requirements

G.3.2.1 Vocabulary

Types of entity to be represented:

- conceptual entities
- physical entities including:
  - protein structures
  - protein structure domains
  - sheets
  - strands
  - helices
  - loops

Types of relation to be represented:

- super/subclass

- has a known structure
- has part
- follows

Every entity is part of a class hierarchy and inherits properties from its superclasses. The relationship 'has known structure' indicates that a class of protein structures is exemplified by a particular known protein structure. 'Has part' and 'follows' are relationships between physical entities.

Other requirements:

- border width and colour should be used to discriminate between nodes representing conceptual and physical entities
- colour or border type should be used to distinguish between nodes representing the different forms of physical entity

G.3.2.2 Layout

- the spatial arrangement of the nodes should reflect the transitive nature of the relationships between them
- the super/subclass relation should be shown vertically
- the 'has part' relation should be shown in depth
- the 'follows' relation should be shown from left to right

G.3.3 Representation Specification

G.3.3.1 Vocabulary

Two shades of brown were chosen for use in representing the two main categories of conceptual and physical entities, one shade darker and more saturated than the other. A medium shade of brown which would show up well against either of these, and against the background was chosen for use in drawing node borders. A very dark brown colour was chosen for the background.

$$L3DarkNodeCol, L3LightNodeCol, L3OutlineCol, L3BackgroundCol : COLOUR$$

Nodes could be one of a number of different heights and widths:

$$\begin{array}{|l} L3NodeHeightA, L3NodeHeightB, L3NodeHeightC, \\ L3NodeWidth : \mathbb{Z}_1 \\ \hline L3NodeWidth > L3NodeHeightA \\ L3NodeHeightA > L3NodeHeightB \\ L3NodeHeightB > L3NodeHeightC \end{array}$$

For all types of node, the colour of the border and label was the same:

L3CommonNodeType

VisualNodeType

BorderColour = *L3OutlineCol*
LabelColour = *L3BackgroundCol*

Nodes representing conceptual properties can therefore be defined as follows:

L3Concepts

L3CommonNodeType

Height = *L3NodeHeightA*
Width = *L3NodeWidth*
BodyColour = *L3LightNodeCol*
BorderStyle = *solid*
BorderWidth = *thin*
LabelStyle = *upper*

Nodes representing physical entities shared two further properties:

L3PhysicalNodeType

L3CommonNodeType

BodyColour = *L3DarkNodeCol*
BorderWidth = *thick*

Nodes representing particular kinds of physical entities can therefore be described as follows:

L3KnownStruct

L3PhysicalNodeType

Height = *L3NodeHeightA*
Width = *L3NodeWidth*
BorderStyle = *solid*
LabelStyle = *lower*

L3Domain

L3PhysicalNodeType

Height = *L3NodeHeightA*
Width = *L3NodeWidth*
BorderStyle = *dotted*
LabelStyle = *lower*

L3Sheet

L3PhysicalNodeType

Height = *L3NodeHeightA*
Width = *L3NodeWidth*
BorderStyle = *short dashed*
LabelStyle = *upper*

L3Strand

L3PhysicalNodeType

Height = *L3NodeHeightB*
BorderStyle = *pattern1*
LabelStyle = *numeric*

L3Helix

L3PhysicalNodeType

Height = *L3NodeHeightB*
BorderStyle = *pattern2*
LabelStyle = *numeric*

L3Loop

L3PhysicalNodeType

Height = *L3NodeHeightC*
Width = *Height*
BorderStyle = *pattern2*
LabelStyle = *nolabel*

All links were drawn as solid lines:

L3CommonLinkType

VisualLinkType

Line = *solid*

Links representing relationships between conceptual entities were drawn in the same colour as the nodes representing those entities:

L3ConceptualLinkType

L3CommonLinkType

Colour = *L3LightNodeCol*
Join = *top bottom*

Links representing relationships between physical entities were also drawn in the same colour as the nodes representing the corresponding entities:

L3PhysicalLinkType

L3CommonLinkType

Colour = *L3DarkNodeCol*

We may now define the four link types which make up the link vocabulary for this language as follows:

L3Superclass

L3ConceptualLinkType

Arrow = noarrow

L3HasKnownStruct

L3ConceptualLinkType

Arrow = closed

L3HasPart

L3PhysicalLinkType

Join = top bottom
Arrow = open

L3Follows

L3PhysicalLinkType

Join = side side
Arrow = noarrow

G.3.3.2 Language

The complete visual language may now be defined as follows:

Language3

$$
\begin{array}{l}
VisualLanguage \\
FrontPlane, DepthDiff, TopPlane, HeightDiff : \mathbb{Z} \\
\hline
\forall\, d \in Diagrams \bullet \\
\quad d.Vocabulary.NodeTypes = \{L3Concepts, L3KnownStruct, L3Domain, \\
\qquad L3Sheet, L3Strand, L3Helix, L3Loop\} \land \\
\quad d.Vocabulary.LinkTypes = \{L3Superclass, L3HasKnownStruct, L3HasPart, \\
\qquad L3Follows\} \land \\
\quad d.Vocabulary.Colours = \{L3DarkNodeCol, L3LightNodeCol, L3OutlineCol, \\
\qquad L3BackgroundCol\} \\
\forall\, d \in Diagrams \bullet \\
\quad (\forall\, l : \mathrm{ran}\ LinkRep \mid l.VisType = L3Superclass \bullet \\
\qquad (l.Origin).VisType = L3Concepts) \land \\
\quad (\forall\, l : \mathrm{ran}\ LinkRep \mid l.VisType = L3HasKnownStruct \bullet \\
\qquad (l.Origin).VisType = L3Concepts \land \\
\qquad (l.Dest).VisType = L3KnownStruct) \land \\
\quad (\forall\, l : \mathrm{ran}\ LinkRep \mid l.VisType = L3HasPart \bullet \\
\qquad (l.Origin).VisType \in \{L3KnownStruct, L3Domain, L3Sheet\} \land \\
\qquad (l.Dest).VisType \in \{L3Domain, L3Sheet, L3Strand, L3Helix, L3Loop\}) \land \\
\quad (\forall\, l : \mathrm{ran}\ LinkRep \mid l.VisType = L3Follows \bullet \\
\qquad (l.Origin).VisType \in \{L3Strand, L3Helix, L3Loop\} \land \\
\qquad (l.Dest).VisType \in \{L3Strand, L3Helix, L3Loop\} \land \\
\qquad ((l.Origin).VisType = L3Loop \lor (l.Dest).VisType = L3Loop)) \\
\forall\, d \in Diagrams \bullet \\
\quad (\forall\, n : NODE \mid NodeRep(n).VisType = L3Concepts \bullet \\
\qquad NodePos(n).Y > TopPlane \land NodePos(n).Z = FrontPlane) \land \\
\quad (\forall\, n : NODE \mid NodeRep(n).VisType = L3KnownStruct \bullet \\
\qquad NodePos(n).Y = TopPlane \land NodePos(n).Z = FrontPlane) \land \\
\quad (\forall\, n : NODE \mid NodeRep(n).VisType = L3Domain \bullet \\
\qquad NodePos(n).Y = TopPlane - HeightDiff \land \\
\qquad NodePos(n).Z = FrontPlaneDepth + DepthDiff) \land \\
\quad (\forall\, n : NODE \mid NodeRep(n).VisType = L3Sheet \bullet \\
\qquad NodePos(n).Y = TopPlane - 2(HeightDiff) \land \\
\qquad NodePos(n).Z = FrontPlane + 2(DepthDiff)) \land \\
\quad (\forall\, n : NODE \mid NodeRep(n).VisType = L3Strand \lor \\
\qquad NodeRep(n).VisType = L3Helix \lor NodeRep(n).VisType = L3Loop \bullet \\
\qquad NodePos(n).Y = TopPlane - 3(HeightDiff) \land \\
\qquad NodePos(n).Z = FrontPlane + 3(DepthDiff)) \land \\
\quad (\forall\, l : Link \mid LinkRep(l).VisType = L3Superclass \bullet \\
\qquad NodePos(l.Origin).Y > NodePos(l.Dest).Y) \land \\
\quad (\forall\, l : Link \mid LinkRep(l).VisType = L3HasPart \bullet \\
\qquad NodePos(l.Origin).Z < NodePos(l.Dest).Z)
\end{array}
$$

G.4 Case Study 4

G.4.1 Knowledge-Based System Profile

Hardware platform: PC AT compatible

Software: expert system shell with Prolog-like knowledge representation and text-based development environment

Architecture: rule-based

Size: 13 knowledge bases, 25 - 50 rules per knowledge base

General Description:

The system as a whole was intended for use by company accountants in determining employee salaries. The aim of the case study was to explore the possibility of graphically representing the main control rule from a typical knowledge base, together with the chains of conditions traversed during inferencing initiated by that rule. Important entities to be represented included elements of the main control rule itself, elements of the backward chains from components of the control rule, and an initial query statement.

G.4.2 Representation Requirements

G.4.2.1 Vocabulary

Types of entity to be represented:

- components of the control rule
- elements in a backward chain
- query statement

Note that components of the control rule include keywords (IF and THEN), conditions in the antecedent and actions in the consequent. Elements in the backward chain are conditions which must hold in order for the conditions in the antecedent to evaluate to true. The query statement forward chains onto the first condition in the antecedent of the main control rule.

Types of relation to be represented:

- control rule component connectors
- backward chain
- forward chain

The control rule component connectors simply connect together the components of the main control rule. Backward chain links connect a condition in the main control rule with the conditions on which it backward chains. A forward chain link connects the query statement with the first condition in the antecedent of the main control rule.

Other requirements:

- nodes representing components of the control rule should be shown in a different colour to those representing elements in a backward chain
- the query statement should not be too prominent
- components of the control rule and elements of a backward chain should all be shown in green

G.4.2.2 Layout

- components of the control rule should all be shown in the same x-y plane
- elements in a backward chain should be shown behind the condition in the control rule from which the chain originates

G.4.3 Representation Specification

G.4.3.1 Vocabulary

Two shades of green were chosen for use in representing the entities of importance, one shade being darker and more saturated than the other. A dark colour which would show up well against either of these was chosen for use in labelling nodes and drawing links, and a light grey colour was chosen for the background.

$$L4DarkNodeCol, L4LightNodeCol, L4OutlineCol, L4BackgroundCol : COLOUR$$

All nodes were of the same height and had one of two different widths.

$$L4NodeHeight, L4NodeWidthA, L4NodeWidthB : \mathbb{Z}_1$$

$$L4NodeWidthA > L4NodeWidthB$$
$$L4NodeWidthB > L4NodeHeight$$

All nodes had certain common properties which we may define as follows.

L4CommonNodeType

$$VisualNodeType$$

$$Height = L4NodeHeight$$
$$BorderColour = L4OutlineCol$$
$$LabelColour = L4OutlineCol$$
$$BorderStyle = solid$$

Nodes representing components of the main control rule were dark green and had borders of medium thinkness:

L4Control

$$L4CommonNodeType$$

$$BodyColour = L4DarkNodeCol$$
$$BorderWidth = medium$$

Nodes representing keywords in the main control rule were labelled in upper case with the keyword they represented and were less wide than other nodes. Nodes representing conditions or actions in the control rule were labelled with the text of the appropriate condition or action written in lower case.

L4ControlKey

L4Control

Width = *L4NodeWidthB*
Labelstyle = *upper*

L4ControlCond

L4Control

Width = *L4NodeWidthA*
Labelstyle = *lower*

Nodes representing elements in a backward chain were light coloured with thin borders and were of the same width as those representing conditions and actions in the control rule. They were labelled with the text of the appropriate condition in lower case.

L4BackCond

L4CommonNodeType

Width = *L4NodeWidthA*
BodyColour = *L4LightNodeCol*
BorderWidth = *thin*
LabelStyle = *lower*

Finally, the node representing the query statement was light coloured and the same width as nodes representing conditions or actions, but had a thick border. It was labelled with the text of the query statement in lower case.

L4Query

L4CommonNodeType

Width = *L4NodeWidthA*
BodyColour = *L4LightNodeCol*
BorderWidth = *thick*
LabelStyle = *lower*

All links were the same colour, and were drawn as solid lines:

L4CommonLinkType

L4VisualLinkType

Line = *solid*
Colour = *L4OutlineCol*

Links representing the different relationships were as follows:

L4Control

$$L4CommonLinkType$$

$$Join = top\ bottom$$
$$Arrow = noarrow$$

L4BackChain

$$L4CommonLinkType$$

$$Join = side\ side$$
$$Arrow = open$$

L4ForChain

$$L4CommonLinkType$$

$$Join = side\ side$$
$$Arrow = closed$$

G.4.3.2 Language

The complete visual language may now be defined as follows:

Language4

$$VisualLanguage$$
$$FrontPlaneDepth, DepthDiff : \mathbb{Z}$$

$$\forall d \in Diagrams \bullet$$
$$d.Vocabulary.NodeTypes = \{L4ControlKey, L4ControlCond, L4BackCond, L4Query\} \land$$
$$d.Vocabulary.LinkTypes = \{L4Control, L4BackChain, L4ForChain\} \land$$
$$d.Vocabulary.Colours = \{L4DarkNodeCol, L4LightNodeCol, L4OutlineCol, L4BackgroundCol\}$$
$$\forall d \in Diagrams \bullet$$
$$(\forall l : \operatorname{ran} d.LinkRep \mid l.VisType = L4BackChain \bullet$$
$$((l.Origin).VisType = L4ControlCond \lor$$
$$(l.Origin).VisType = L4BackCond) \land$$
$$(l.Dest).VisType = L4BackCond) \land$$
$$(\forall l : \operatorname{ran} d.LinkRep \mid l.VisType = L4ForChain \bullet$$
$$(l.Origin).VisType = L4Query \land$$
$$(l.Dest).VisType = L4ControlCond)$$
$$\forall d \in Diagrams \bullet$$
$$(\forall n : NODE \mid$$
$$NodeRep(n).VisType = L4ControlKey \lor$$
$$NodeRep(n).VisType = L4ControlCond \bullet$$
$$NodePos(n).Z = FrontPlaneDepth) \land$$
$$(\forall n : NODE \mid NodeRep(n).VisType = L4Query \bullet$$
$$NodePos(n).Z = FrontPlaneDepth + DepthDiff) \land$$
$$(\forall l : Link \mid LinkRep(l).VisType = L4BackChain \bullet$$
$$NodePos(l.Dest).Z = NodePos(l.Origin).Z + DepthDiff)$$

G.5 Case Study 5

G.5.1 Knowledge-Based System Profile

Hardware platform: general purpose workstation

Software: Lisp environment including 2-d graphical browser

Architecture: mainly frame-based, some rules

Size: 13 - 14 separate knowledge bases, total 60 MBytes

General Description:

The system as a whole aimed to capture expert knowledge about the design of human-computer interfaces. Important entities in the part of the system considered were frames representing either graphical or data dictionary objects. The aim of the graphical representation was to depict a relationship of interest between objects in the two class hierarchies.

G.5.2 Representation Requirements

G.5.2.1 Vocabulary

Types of entity to be represented:

- data dictionary objects
- graphical objects

Types of relation to be represented:

- super/subclass
- shows

Other requirements:

- entities of each type are related to other entities of the same type by super/subclass relations
- the shows relation links graphical objects with the data dictionary objects they are used to depict
- all entities and relations within a class hierarchy should be represented in the same way
- different types of entities and relations should be visually discriminable
- links representing the shows relation should be more visually prominent than those representing the super/subclass relation
- arrow heads should not be used

G.5.2.2 Layout

- all entities of the same type should be shown within a single lamina, (*ie* in the same $x - y$ plane)
- entities of different types should be shown in different laminas
- the super/subclass relation between entities of the same type should run from left to right within a lamina
- one lamina should be displayed directly in front of the other

G.5.3 Representation Specification

G.5.3.1 Vocabulary

Two saturated and visually opposing colours were chosen for use in representing the two different types of entity. A dark colour which would show up well against either of these was chosen for use in labelling nodes and drawing links, and a light grey colour was chosen for the background.

$$L5NodeColA, L5NodeColB, L5OutlineCol, L5BackgroundCol : COLOUR$$

All nodes were of the same height.

$$L5NodeHeight : \mathbf{Z}_1$$

All nodes had certain common properties which we may define as follows.

L5CommonNodeType

VisualNodeType

$Height = L5NodeHeight$
$BorderColour = L5OutlineCol$
$LabelColour = L5OutlineCol$
$BorderStyle = solid$
$BorderWidth = thin$
$LabelStyle = upper$

Nodes representing entities of different types were of different colours:

L5DataObject

L5CommonNodeType

$BodyColour = L5NodeColA$

L5GraphicalObject

L5CommonNodeType

$BodyColour = L5NodeColB$

Links also had a number of properties in common which may be defined as follows.

L5CommonLinkType

VisualLinkType

Colour = L5OutlineCol
Join = side side
Arrow = noarrow

Links representing different relationship types were as follows:

L5Superclass

L5CommonLinkType

Line = short dashed

L5Shows

L5CommonLinkType

Line = solid

G.5.3.2 Language

Before a complete design could be specified, the following extra constraints had to be adopted by the designer:

- the lamina of graphical object entities should be displayed in front of the lamina of data dictionary objects

The complete visual language may now be defined as follows:

Language5

VisualLanguage

$FrontPlaneDepth, BackPlaneDepth : \mathbb{Z}$

$FrontPlaneDepth < BackPlaneDepth$

$\forall d \in Diagrams \bullet$

$d.Vocabulary.NodeTypes = \{L5DataObject, L5GraphicalObject\} \land$

$d.Vocabulary.LinkTypes = \{L5Superclass, L5Shows\} \land$

$d.Vocabulary.Colours = \{L5NodeColA, L5NodeColB, L5OutlineCol, L5BackgroundCol\}$

$\forall d \in Diagrams \bullet$

$(\forall l : \text{ran } LinkRep \mid l.VisType = L5Shows \bullet$

$(l.Origin).VisType = L5GraphicalObject \land$

$(l.Dest).VisType = L5DataObject) \land$

$(\forall l : \text{ran } LinkRep \mid l.VisType = L5Superclass \bullet$

$\exists n : VisualNodeType \bullet$

$(l.Origin).Vistype = n \land (l.Dest).Vistype = n)$

$\forall d \in Diagrams \bullet$

$(\forall n : NODE \mid NodeRep(n).VisType = L5GraphicalObject \bullet$

$NodePos(n).Z = FrontPlaneDepth) \land$

$(\forall n : NODE \mid NodeRep(n).VisType = L5DataObject \bullet$

$NodeRep(n).Z = BackPlaneDepth) \land$

$(\forall l : Link \mid LinkRep(l).VisType = L5Superclass \bullet$

$NodePos(l.Origin).X < NodePos(l.Dest).X)$

G.6 Case Study 6

G.6.1 Knowledge-Based System Profile

Hardware platform: PC AT compatible

Software: Prolog and word processor

Architecture: purpose-built FOPC representation describes frame- and rule-like structures

Size: 90 KBytes

General Description:

The system was intended for use by company accountants in managing the company accounts. Important entities in the part of the system considered were the various account codes, some of which were valid with other codes, some not. The aim of the graphical representation was to illustrate which codes were valid with which others by explicitly depicting an 'is valid with' relation between certain pairs of codes.

G.6.2 Representation Requirements

G.6.2.1 Vocabulary

Types of entity to be represented:

- product codes
- customer codes
- account codes

also:

- valid codes
- invalid codes

Note that these classifications are orthogonal. A code may be either valid or invalid whether it is a product, customer or account code.

Types of relation to be represented:

- has child
- is valid with

Other requirements:

- codes of each type are related to other codes of the same type in a hierarchy by 'has child' relations
- all codes within a given hierarchy should be represented by nodes of the same kind
- colour should be used to discriminate between different hierarchies of codes

- 'has child' relations should be represented using links which are the same colour as the nodes in the hierarchy
- there should be no arrow heads on links representing the 'has child' relationship
- the 'is valid with' relation links the top-most code of a sub-hierarchy which is valid with a segment of another hierarchy with the top-most code in that other segment
- links representing the'is valid with' relation should be a different colour to any of the other links
- links representing the 'is valid with' relation should have arrow heads to indicate the directionality of the relationship
- border width should be used to discriminate between codes which are valid with other codes and those which are not

G.6.2.2 Layout

- the 'has child' relation should run from top to bottom of the screen
- nodes representing product, customer and account codes should each be shown in a different x-y plane
- it should be possible to use the difference in apparent size due to perspective to discriminate between nodes representing codes in different hierarchies

G.6.3 Representation Specification

G.6.3.1 Vocabulary

Three saturated and highly discriminable colours were chosen for use in representing the three different hierarchies of codes. A light grey colour was chosen for the background.

$$L6NodeColA, L6NodeColB, L6NodeColC, L6BackgroundCol : COLOUR$$

All nodes were of the same height and width.

$$L6NodeHeight, L6NodeWidth : \mathbb{Z}_1$$
$$L6NodeWidth > L6NodeHeight$$

All nodes had certain common properties which we may define as follows.

L6CommonNodeType

$$VisualNodeType$$
$$Height = L6NodeHeight$$
$$Width = L6NodeWidth$$
$$BodyColour = L6BorderCol$$
$$LabelColour = L6BackgroundCol$$
$$BorderStyle = solid$$
$$BorderWidth = thin$$
$$LabelStyle = lower$$

Colour was used to discriminate between nodes representing codes in different hierarchies:

L6 CustomerCode
L6 CommonNode Type

BodyColour = *L6 NodeColA*

L6 ProductCode
L6 CommonNode Type

BodyColour = *L6 NodeColB*

L6 AccountCode
L6 CommonNode Type

BodyColour = *L6 NodeColC*

Border thickness was used to discriminate between nodes representing codes which were valid with other codes and those which were not:

L6 ValidCode
L6 CommonNode Type

BorderWidth = *thick*

L6 InvalidCode
L6 CommonNode Type

BorderWidth = *thin*

This yields a total of six types of node:

L6 ValidCustCode
L6 CustomerCode
L6 ValidCode

L6 InvalidCustCode
L6 CustomerCode
L6 InvalidCode

L6 ValidProdCode
L6 ProductCode
L6 ValidCode

L6InvalidProdCode

L6ProductCode
L6InvalidCode

L6ValidAccCode

L6AccountCode
L6ValidCode

L6InvalidAccCode

L6AccountCode
L6InvalidCode

Links representing the 'has child' relationship within each of the code hierarchies have the following properties in common:

L6HasChild

VisualLinkType

Line = solid
Join = top bottom
Arrow = noarrow

Links representing the three distinct 'has child' relationships can therefore be defined as follows:

L6CustHasChild

L6HasChild

Colour = L6NodeColA

L6ProdHasChild

L6HasChild

Colour = L6NodeColB

L6AccHasChild

L6HasChild

Colour = L6NodeColC

Links representing the 'is valid with' relation are defined as follows:

L6IsValidWith

VisualLinkType

Line = solid
Colour = L6NodeColC
Join = side side
Arrow = closed

G.6.3.2 Language

Before a complete design could be specified, the following extra constraint had to be adopted by the designer:

- the three planes of code hierarchies should be displayed in the order (from front to back) product, account, customer

The complete visual language may now be defined as follows:

Language6

$$
\begin{array}{l}
VisualLanguage \\
FrontPlaneDepth, MidPlaneDepth, BackPlaneDepth, \\
DepthDiff : \mathbb{Z} \\
\hline
MidPlaneDepth = FrontPlaneDepth + DepthDiff \\
BackPlaneDepth = MidPlaneDepth + DepthDiff \\
\forall\, d \in Diagrams \bullet \\
\quad d.Vocabulary.NodeTypes = \{L6\,ValidCustCode, L6InvalidCustCode, \\
\qquad L6\,ValidProdCode, L6InvalidProdCode, L6\,ValidAccCode, \\
\qquad L6InvalidAccCode\} \land \\
\quad d.Vocabulary.LinkTypes = \{L6\,CustHasChild, L6\,ProdHasChild, \\
\qquad L6\,AccHasChild, L6\,IsValidWith\} \land \\
\quad d.Vocabulary.Colours = \{L6NodeColA, L6NodeColB, L6NodeColC, \\
\qquad L6\,BackgroundCol\} \\
\forall\, d \in Diagrams \bullet \\
\quad (\forall\, l : \mathrm{ran}\, LinkRep \bullet \\
\qquad (l.VisType = L6\,CustHasChild \Rightarrow \\
\qquad\quad (l.Origin).VisType \in \{L6\,ValidCustCode, L6InvalidCustCode\} \land \\
\qquad\quad (l.Dest).VisType \in \{L6\,ValidCustCode, L6InvalidCustCode\}) \land \\
\qquad (l.VisType = L6\,ProdHasChild \Rightarrow \\
\qquad\quad (l.Origin).VisType \in \{L6\,ValidProdCode, L6InvalidProdCode\} \land \\
\qquad\quad (l.Dest).VisType \in \{L6\,ValidProdCode, L6InvalidProdCode\}) \land \\
\qquad (l.VisType = L6\,AccHasChild \Rightarrow \\
\qquad\quad (l.Origin).VisType \in \{L6\,ValidAccCode, L6InvalidAccCode\} \land \\
\qquad\quad (l.Dest).VisType \in \{L6\,ValidAccCode, L6InvalidAccCode\})) \land \\
\quad (\forall\, l : \mathrm{ran}\, LinkRep \mid l.VisType = L6\,IsValidWith \bullet \\
\qquad l.Origin \in \{L6\,ValidCustCode, L6\,ValidProdCode, L6\,ValidAccCode\} \land \\
\qquad l.Dest \in \{L6\,ValidCustCode, L6\,ValidProdCode, L6\,ValidAccCode\}) \\
\forall\, d \in Diagrams \bullet \\
\quad (\forall\, n : NODE \bullet \\
\qquad (NodeRep(n).VisType = L6\,ValidProCode \lor \\
\qquad NodeRep(n).VisType = L6InvalidProCode \Rightarrow \\
\qquad\quad NodePos(n).Z = FrontPlaneDepth) \land \\
\qquad (NodeRep(n).VisType = L6\,ValidAccCode \lor \\
\qquad NodeRep(n).VisType = L6InvalidAccCode \Rightarrow \\
\qquad\quad NodePos(n).Z = MidPlaneDepth) \land \\
\qquad (NodeRep(n).VisType = L6\,ValidCustCode \lor \\
\qquad NodeRep(n).VisType = L6InvalidCustCode \Rightarrow \\
\qquad\quad NodePos(n).Z = BackPlaneDepth)) \land \\
\quad (\forall\, l : Link \mid LinkRep(l).VisType = L6\,CustHasChild \lor \\
\qquad LinkRep(l).VisType = L6\,AccHasChild \lor \\
\qquad LinkRep(l).VisType = L6\,ProdHasChild \bullet \\
\qquad\quad NodePos(l.Origin).Y > NodePos(l.Dest).Y) \land \\
\quad (\forall\, l : Link \mid LinkRep(l).VisType = L6\,IsValidWith \bullet \\
\qquad NodePos(l.Origin).Z < NodePos(l.Dest).Z)
\end{array}
$$

Bibliography

[1] L. Adams, W. Krybus, D. Meyer-Ebrecht, R. Rueger, J. Gilsbach, R. Moesges, and G. Schloendorff. Computer-assisted surgery. *IEEE Computer Graphics and Applications*, 10(3), 1990.

[2] A. Ambler and M. Burnett. Influence of visual technology on the evolution of language environments. *IEEE Computer*, 22, October 1989.

[3] J. R. Anderson. Arguments concerning representations for mental imagery. *Psychological Review*, 85(4), 1978.

[4] W. Barfield and R. Robless. The effects of two and three-dimensional graphics on the problem-solving performance of experienced and novice decision makers. *Behaviour and Information Technology*, 8(5), 1989.

[5] J. Barnett, K. Knight, I. Mani, and E. Rich. Knowledge and natural language processing. *Communications of the ACM*, 33(8), 1990.

[6] V. Bellotti. Implications of current design practice for the use of HCI techniques. In D. M. Jones and R. Winder, editors, *People and Computers IV*. CUP, 1988.

[7] J. Bennet. Managing to meet usability requirements: Establishing and meeting software development goals. In J. Bennet et al, editor, *Visual Display Terminals*. 1984.

[8] J. Bertin. *Semiologie Graphique*. Wisconsin, 1967.

[9] K. Boff and J. Lincoln. *Engineering Data Compendium: Human Perception and Performance*. AAMRL, Wright-Patterson AFB, 1988.

[10] J. Boose and J. Bradshaw. Expertise transfer and complex problems: Using aquinas as a knowledge acquisition workbench for knowledge-based systems. *International Journal of Man-Machine Systems*, 26, 1987.

[11] A. Borning. The programming language aspects of thinglab; a constraint-oriented simulation laboratory. *Transactions on Programming Language and Systems*, 3(4), 1981.

[12] A. Borning. Defining constraints graphically. In *Proceedings CHI86*, 1986.

[13] M. Braunstein and G. Andersen. Velocity gradients and relative depth perception. *Perception and Psychophysics*, 29(2), 1981.

[14] M. Braunstein, G. Andersen, and D. Reifer. The use of occlusion to resolve ambiguity in parallel projections. *Perception and Psychophysics*, 31(3), 1982.

[15] M. Braunstein, G. Andersen, M. Rouse, and J. Tittle. Recovering viewer-centred depth from disparity, occlusion and velocity gradients. *Perception and Psychophysics*, 40(4), 1986.

[16] M. Brayshaw and M. Eisenstadt. A practical tracer for Prolog. Technical Report 42, Human Cognition Research Laboratory, Open University, 1989.

[17] J. Brooke, N. Bevan, F. Brigham, S. Harker, and D. Youmans. Usability statements and standardisation - work in progress at ISO. In D. Diaper, D. Gilmore, G. Cockton, and B. Shackel, editors, *Human-Computer Interaction: INTERACT90.* Elsevier Science Publishers BV (North-Holland), 1990.

[18] J. B. Brooke and K. D. Duncan. Effects of system display format on performance in a fault location task. *Ergonomics*, 24(3), 1981.

[19] F. Brooks, M. Ouh-Young, J. Batter, and P. Kilpatrick. Project GROPE - haptic displays for scientific visualisation. *Computer Graphics*, 24(4), 1990. Proceedings SIGGRAPH90.

[20] J. Seely Brown. From cognitive to social ergonomics and beyond. In D. Norman and S. Draper, editors, *User Centred System Design*, chapter 22. Lawrence Erlbaum Associates, 1986.

[21] D. Bryce and R. Hull. SNAP: A graphics-based schema manager. In *Proceedings of the IEEE International Conference on Data Engineering*, 1986.

[22] S. K. Card, T. P. Moran, and A. Newell. *The Psychology of Human-Computer Interaction.* Erlbaum, 1983.

[23] J. Carroll and W. Kellogg. Artifact as theory-nexus: Hermeneutics meets theory-based design. In *Proceedings CHI89*, 1989.

[24] C. M. Carswell, S. Frankenberger, and D. Bernhard. Graphing in depth: Perspectives on the use of three-dimensional graphs to represent lower-dimensional data. *Behaviour and Information Technology*, 10(6), 1991.

[25] S. Chang. Visual languages: A tutorial and survey. *IEEE Software*, January 1987.

[26] W. S. Cleveland and R. McGill. Graphical perception: Theory, experimentation and application to the development of graphical methods. *Journal of the American Statistical Association*, 79(387), 1984.

[27] W. S. Cleveland and R. McGill. An experiment in graphical perception. *International Journal of Man-Machine Studies*, 25(5), 1986.

[28] L. Console, M. Fasso, and P. Torasso. Knowledge acquisition via a graphical interface. In I. Plander, editor, *Artificial Intelligence and Information-Control Systems of Robots 87.* Elsevier Science Publishers BV (North-Holland), 1987.

[29] E. Cordingley. Knowledge elicitation techniques for knowledge-based systems. In D. Diaper, editor, *Knowledge Elicitation: Principles, Techniques and Applications*, chapter 3, pages 104 – 116. Ellis Horwood, 1989. (Section on interview techniques).

[30] J. Cugini. Graphical conceptual navigation as a presentation technique for a graphics standard. In *NCGA89*, 1989.

[31] J. Cui. Interactive three dimensional connection graphics, 1990. MPhil Thesis: Department of Computer Science, City University.

[32] B. Curtis, H. Krasner, and N. Iscoe. A field study of the software design process for large systems. *Communications of the ACM*, 31(11), 1988.

[33] B. Czejdo, R. Elmasri, and M. Rusinkiewicz. A graphical data manipulation language for an extended entity-relationship model. *IEEE Computer*, March 1990.

[34] A. Dewar and J. Cleary. Graphical display of complex information within a Prolog debugger. *International Journal of Man-Machine Studies*, 25, 1986.

[35] C. Ding and P. Mateti. A framework for the automated drawing of data structure diagrams. *IEEE Transactions on Software Engineering*, 16(5), 1990.

[36] D. Dodson. 2-3/4-D: Interactive 3-d diagrams without 3-d graphics hardware. Technical Report TCU/CS/1991/2, City University, 1991.

[37] D. Dodson. ICDEDIT: Interactive 3-d diagrams without 3-d graphics hardware. Technical Report TCU/CS/1991/1, City University, 1991.

[38] D. C. Dodson. Interaction with knowledge systems through connection diagrams. In D. Berry and A. Hart, editors, *Expert Systems: Human Issues*. Kogan Page, 1989. To appear.

[39] D. C. Dodson. JIN: A prototype system for 3-dimensional interactive connection diagram (ICD) graphics. In *Proceedings of the 4th Alvey Explanation Workshop*, 1989.

[40] J. Domingue and M. Eisenstadt. A new metaphor for the graphical explanation of forward-chaining rule execution. Technical Report 52, Human Cognition Research Laboratory, Open University, 1989.

[41] T. Dudley. Multi-dimensional interfaces for software design. In D. Diaper, D. Gilmore, G. Cockton, and B. Shackel, editors, *Human-Computer Interaction: INTERACT90*. Elsevier Science Publishers BV (North-Holland), 1990. (Panel session).

[42] K. Eason. Toward the experimental study of usability. *Behaviour and Information Technology*, 3(2), 1984.

[43] A. Ryan (ed). *Utilitarianism and Other Essays: J. S. Mill and Jeremy Bentham*. Penguin, 1987.

[44] M. Good (ed). Seven experiences with contextual field research. *ACM SIGCHI Bulletin*, 20(4), 1989.

[45] M. Helander (ed). *Handbook of Human-Computer Interaction*. Elsevier Science Publishers BV (North-Holland), 1988.

[46] P. Szolovits (ed). *Artificial Intelligence in Medicine*. Westview Press, 1982. (AAAS Selected Symposium 51).

[47] M. Eisenstadt and M. Brayshaw. AORTA diagrams as an aid to visualising the execution of Prolog programs. In A. C. Kilgour and R. A. Earnshaw, editors, *Graphics Tools for Software Engineering*. BCS Documentation and Displays Group, 1988.

[48] M. Eisenstadt, J. Domingue, T. Rajan, and E. Motta. Visual knowledge engineering. *IEEE Transactions on Software Engineering*, 16(10), 1990.

[49] P. Eklund and P. Kuczora. A conceptual graph implementation for knowledge base systems development. In *Human and Organisational Issues of Expert Systems*, 1988.

[50] K. A. Ericsson and H. Simon. Verbal reports as data. *Psychological Review*, 87(3), 1980.

[51] K. Fairchild, G. Meredith, and A. Wexelblat. The Tourist artificial reality. In *Proceedings CHI89*, 1989.

[52] K. M. Fairchild, S. E. Poltrock, and G. W. Furnas. SemNet: Three-dimensional graphic representations of large knowledge bases. In R. Guindon, editor, *Cognitive Science and its Applications for Human-Computer Interaction.* Lawrence Erlbaum Associates, 1988.

[53] T. E. Ferrin, C. C. Huang, L. E. Jarvis, and R. Langridge. The MIDAS display system. *Journal of Molecular Graphics*, 6(3), 1988.

[54] S. Fickas. Supporting the programmer of a rule-based language. *Expert Systems*, 4(2), 1987.

[55] R. A. Finke. Theories relating mental imagery to perception. *Psychological Bulletin*, 98(2), 1985.

[56] R. A. Finke. Mental imagery and the visual system. *Scientific American*, 254(3), 1986.

[57] M. J. Fitter and T. R. G. Green. When do diagrams make good computer languages? In J. L. Alty and M. J. Coombs, editors, *Computing Skills and the User Interface.* Academic Press, 1981.

[58] J. D. Foley and V. L. Wallace. The art of natural graphic man-machine conversation. *Proceedings of the IEEE*, 62(4), 1974.

[59] J. D. Foley, V. L. Wallace, and P. Chan. The human factors of computer graphics interaction techniques. *IEEE Computer Graphics and Applications*, November 1984.

[60] M. J. Freiling and J. H. Alexander. Diagrams and grammars: Tools for mass-producing expert systems. In *Proceedings of the 1st Conference on AI Applications.* IEEE, 1984.

[61] H. Fuchs, M. Levoy, and S. Pizer. Interactive visualisation of 3-d medical data. *IEEE Computer*, August 1989.

[62] J. Gibson. *The Ecological Approach to Visual Perception.* Houghton Mifflin, 1979.

[63] M. Gittins. LOOPS: a multi-paradigm environment. In R. Hawley, editor, *Artificial Intelligence Programming Environments.* Ellis Horwood, 1987.

[64] E. Glinert. Towards software metrics for visual programming. *International Journal of Man-Machine Studies*, 30, 1989.

[65] K. J. Goldman, S. A. Goldman, P. C. Kannelakis, and S. B. Zdonik. ISIS: Interface for a semantic information system. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 1985.

[66] E. Golin. *A Method for Specification and Parsing of Visual Languages.* PhD thesis, Brown University, 1990.

[67] E. Golin and S. Reiss. The specification of visual language syntax. *Journal of Visual Languages and Computing*, 1, 1990.

[68] S. Grotch. Three-dimensional and stereoscopic graphics for scientific data display and analysis. *IEEE Computer Graphics and Applications*, November 1983.

[69] J. Grudin. Utility and usability: Research issues and development contexts. *Interacting with Computers*, 4(2), 1992.

[70] J. Hannan and P. Politakis. ESSA: An approach to acquiring decision rules for diagnostic expert systems, 1985. IEEE publication.

[71] D. Hearn and M. Baker. *Computer Graphics.* Prentice Hall, 1986.

[72] L. Hodges and D. McAllister. True three-dimensional CRT-based displays. *Information Display*, 5, 1987.

[73] H. Iwata. Artificial reality with force-feedback: Development of desktop virtual space with compact master manipulator. *Computer Graphics*, 24(4), 1990. Proceedings SIGGRAPH90.

[74] D. Jordan, J. McDermid, and I. Toyn. CADiZ - computer aided design in Z. In J. E. Nicholls, editor, *Workshops in Computing: Z User Workshop.* Springer Verlag, 1991.

[75] B. Julesz. Binocular depth perception of computer generated patterns. *Bell System Technology Journal*, 39, 1960.

[76] T. Kamada and S. Kawai. A general framework for visualising abstract objects and relations. *ACM Transactions on Graphics*, 10(1), 1991.

[77] J. Kellett and L. Esfahani. Human computer issues in knowledge engineering tools: VEGAN and KET. In *Human and Organisational Issues of Expert Systems*, May 1988.

[78] W. Kellogg. Qualitative artifact analysis. In D. Diaper, D. Gilmore, G. Cockton, and B. Shackel, editors, *Human-Computer Interaction: INTERACT90.* Elsevier Science Publishers BV (North-Holland), 1990.

[79] A. L. Kidd and M. B. Cooper. Man-machine interface for an expert system. In *Proceedings of Expert Systems '83.* BCS Expert Systems Special Interest Group, 1983.

[80] A. Korte. Kinematoskopische Untersuchungen. *Zeitschrift fur Psychologie*, 72, 1915.

[81] J. C. Kunz, T. P. Kehler, and M. D. Williams. Applications development using a hybrid AI development system. In R. Hawley, editor, *Artificial Intelligence Programming Environments.* Ellis Horwood, 1987.

[82] M. Kyng. Scenario? Guilty! *ACM SIGCHI Bulletin*, 24(4), 1992.

[83] T. Landauer. Research methods in human-computer interaction. In M. Helander, editor, *Handbook of Human-Computer Interaction*, chapter 42. Elsevier Science Publishers (North-Holland), 1988.

[84] J. Lappin, J. Doner, and B. Kottas. Minimal conditions for the visual detection of structure and motion in three dimensions. *Science*, 209, August 1980.

[85] J. H. Larkin and H. A. Simon. Why a diagram is (sometimes) worth a thousand words. *Cognitive Science*, 11, 1987.

[86] J. Lewis. An effective graphics user interface for rules and inference mechanisms. In *Proceedings CHI83*, 1983.

[87] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Computer Graphics*, 5(2), 1986.

[88] J. Mackinlay, S. Card, and G. Robertson. Rapid controlled movement through a virtual 3-d workspace. *Computer Graphics*, 24(4), 1990. Proceedings SIGGRAPH90.

[89] A. MacLean, P. Barnard, and N. Hammond. Recall as an indicant of performance in interactive systems. In B. Shackel, editor, *Human-Computer Interaction — INTERACT84*. Elsevier Science Publishers BV, 1984.

[90] Macromind Inc. *Macromind Director, version 3.0*, 1991.

[91] M. Maguire and M. Sweeney. Laboratory based methods and computer support tools for evaluating human computer interaction. In *Usability Metrics and Methodologies*, 1990. Proceedings of a one day meeting of the BCS HCI SG, 21 May.

[92] D. Marr. *Vision.* Freeman, 1982.

[93] C. Marshall, B. Christie, and M. M. Gardiner. Assessment of trends in the technology and techniques of human-computer interaction. In M. M. Gardiner and B. Christie, editors, *Applying Cognitive Psychology to User Interface Design.* John Wiley, 1987.

[94] P. Mercurio and T. Erickson. Interactive scientific visualisation: An assessment of a virtual reality system. In D. Diaper, D. Gilmore, G. Cockton, and B. Shackel, editors, *Human-Computer Interaction: INTERACT90.* Elsevier Science Publishers BV (North-Holland), 1990.

[95] P. Mott and S. Brooke. A graphical inference mechanism. *Expert Systems*, 4(2), 1987.

[96] M. A. Musen, L. M. Fagan, and E. H. Shortliffe. Graphical specification of procedural knowledge for an expert system. Technical report, IEEE Computer Society, 1986. Reprinted from the 1986 Computer Society Workshop on Visual Languages.

[97] B. Myers. Taxonomies of visual programming and program visualisation. *Journal of Visual Languages and Computing*, 1(1), 1990.

[98] B. Nardi. The use of scenarios in design. *ACM SIGCHI Bulletin*, 24(4), 1992.

[99] N. Negroponte. An iconoclastic view beyond the desktop metaphor. *International Journal of Human-Computer Interaction*, 1(1), 1989.

[100] Neuron Data. Publicity material, 1990.

[101] J. Nielsen and R. Molich. Heuristic evaluation of user interfaces. In *Proceedings CHI90*, 1990.

[102] J. Nosek and I. Roth. A comparison of formal knowledge representation schemes as communication tools: Predicate logic vs semantic network. *International Journal of Man-Machine Studies*, 33, 1990.

[103] C. Wolf (organiser). The role of laboratory experiments in HCI: Help, hindrance or ho-hum? In *Proceedings CHI89*, 1989. (Panel session).

[104] R. Parslow. 3d visualisation. In D. Greenaway and E. Warman, editors, *Eurographics '82*. North-Holland Publishing Company, 1982.

[105] Pauker, Gorry, Kassirer, and Schwartz. Towards the simulation of clinical cognition. In W. Clancey and E. Shortliffe, editors, *Readings in Medical Artificial Intelligence*. Addison Wesley, 1984.

[106] M. Petre and T. R. G. Green. Where to draw the line with text: Some claims by logic designers about graphics in notation. In D. Diaper, D. Gilmore, G. Cockton, and B. Shackel, editors, *Human-Computer Interaction: INTERACT90*. Elsevier Science Publishers BV (North-Holland), 1990.

[107] B. Potter, J. Sinclair, and D. Till. *An Introduction to Formal Specification and Z*. Prentice Hall, 1991. International Series in Computer Science.

[108] K. Prazdny. Three-dimensional structure from long-range apparent motion. *Perception*, 15, 1986.

[109] Z. W. Pylyshyn. The imagery debate: Analogue media versus tacit knowledge. *Psychological Review*, 88(1), 1981.

[110] J. Ramanathan and R. Hartung. A generic iconic tool for viewing databases. *IEEE Software*, September 1989.

[111] VPL Research. *Swivel 3D Professional, version 2.0*, 1991.

[112] S. Richardson. Operationalising usability and acceptability: A methodological review. In J. Wilson and E. Corlett, editors, *New Methods in Applied Ergonomics*. 1987.

[113] M. H. Richer and W. J. Clancey. Guidon-watch: A graphic interface for viewing a knowledge based system. *IEEE Computer Graphics and Applications*, 5(1), 1985.

[114] W. Ripka. Computers picture the perfect drug. *New Scientist*, June 1988.

[115] M. Roberts and P. Samwell. A visual programming system for the development of parallel software. Technical report, City University, 1990.

[116] G. Robertson, J. Mackinlay, and S. Card. Cone trees: Animated 3-d visualisations of hierarchical information. In *Reaching Through Technology: Proceedings CHI91*, 1991.

[117] B. Rogers and M. Graham. Motion parallax as an independent cue for depth perception. *Perception*, 8, 1979.

[118] G. Roman and K. Cox. A declarative approach to visualising concurrent computations. *IEEE Computer*, 22, October 1989.

[119] L. Rosenblum. Scientific visualisation at research laboratories. *IEEE Computer*, 22, August 1989.

[120] D. E. Rumelhart, P. Smolensky, J. L. McClelland, and G. E. Hinton. Schemata and sequential thought processes in PDP models. In J. L. McClelland, D. E. Rumelhart, and the PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol 2: Psychological and Biological Models.* MIT Press, 1986.

[121] SAS Institute Inc, Cary, NC, USA. *SAS/STAT User's Guide*, fourth edition, 1990. Version 6, volume 2.

[122] P. Schoeler and A. Fournier. Profiling graphic display systems. In *Proceedings Graphics and Vision Interface '86*, 1986.

[123] B. Senach. Computer-aided problem solving with graphical display of information. In *Psychology of Computer Use.* Academic Press, 1983.

[124] C. Shearer. KEE and Poplog: Alternative approaches to developing major knowledge based systems. In *Proceedings of KBS '87.* Online Publications, 1987.

[125] B. Shneiderman. The future of interactive systems and the emergence of direct manipulation. *Behaviour and Information Technology*, 1, 1982.

[126] B. Shneiderman. Direct manipulation: A step beyond programming languages. *IEEE Computer*, 16, 1983.

[127] R. Shook. SemNet: A conceptual and interface evaluation. Technical Report HI-320-86-P, MCC, Austin, Texas, 1986.

[128] S. Shum. Real and virtual spaces: Mapping from spatial cognition to hypertext. *Hypermedia*, 2(2), 1990.

[129] Silicon Graphics. Realistic visuals at a realistic cost - Marconi's new periscope simulator. Publicity material, 1990.

[130] H. Snyder. Image quality. In M. Helander, editor, *Handbook of Human-Computer Interaction*, chapter 20. Elsevier Science Publishers BV (North-Holland), 1988.

[131] I. Somerville. *Software Engineering.* Addison-Wesley, fourth edition, 1992.

[132] J. Sowa. *Conceptual Structures: Information Processing in Mind and Machine.* Addison Wesley, 1984.

[133] I. Spence. Visual psychophysics of simple graphical elements. *Journal of Experimental Psychology: Human Perception and Performance*, 16(4), 1990.

[134] J. M. Spivey. *The Z Notation.* Prentice Hall, 1989.

[135] L. Standing. Learning 10,000 pictures. *Quarterly Journal of Experimental Psychology*, 25, 1973.

[136] M. Stelzner and M. D. Williams. The evolution of interface requirements for expert systems. In J. A. Hendler, editor, *Expert Systems: The User Interface.* Ablex, 1988.

[137] A. Stevens, B. Roberts, and L. Stead. The use of a sophisticated graphics user interface in computer-assisted instruction. *IEEE Computer Graphics and Applications*, 1983.

[138] A. Sutcliffe and U. Patel. The three-dimensional graphical user interface: Evaluation for design evolution, 1993. Paper submitted to HCI93, the annual conference of the BCS HCI special interest group.

[139] R. Tamassia, C. Batini, and G. Di Battista. Automatic graph drawing and readability of diagrams. *IEEE Transactions on System, Man and Cybernetics*, 1988. To appear.

[140] R. Tamassia, C. Batini, and M. Talamo. An algorithm for automatic layout of entity relationship diagrams. In C. G. Davis, S. Jahodia, P. A. Ng, and R. T. Yeh, editors, *Entity Relationship Approach to Software Engineering*. Elsevier, 1983.

[141] P. Nong Tarlton and M. Tarlton. Pogo: A declarative representation system for graphics. Technical Report ACA-HI-049-88, MCC Human Interface Laboratory, 1988.

[142] E. Tello. Between man and machine. *Byte*, September 1988.

[143] K. Toennies, J. Udupa, G. Herman, I. Wornom III, and S. Buchman. Registration of 3-d objects and surfaces. *IEEE Computer Graphics and Applications*, 10(3), 1990.

[144] S. Tsuji and E. H. Shortliffe. Graphical access to the knowledge base of a medical consultation system. Technical Report HPP-83-6, Stanford Heuristic Programming Project, 1983.

[145] T. Tullis. Screen design. In M. Helander, editor, *Handbook of Human-Computer Interaction*, chapter 18. Elsevier Science Publishers BV (North-Holland), 1988.

[146] A. Vainio-Larsson and R. Orring. Evaluating the usability of user interfaces: Research in practice. In D. Diaper, D. Gilmore, G. Cockton, and B. Shackel, editors, *Human-Computer Interaction: INTERACT90*. Elsevier Science Publishers BV (North-Holland), 1990.

[147] S. G. Vandeberg and A. R. Kuse. Mental rotations, a group test of three-dimensional spatial visualization. *Perceptual and Motor Skills*, 47, 1978.

[148] G. M. Vose and G. Williams. Labview: Laboratory virtual instrument engineering workbench. *Byte*, September 1986.

[149] J. D. Waldern. Studying depth cues in a three-dimensional computer graphics workstation. *International Journal of Man-Machine Studies*, 24, 1986.

[150] H. Wallach and D. O'Connell. The kinetic depth effect. *Journal of Experimental Psychology*, 45, 1953.

[151] P. Walsh and J. Laws. Methods and tools for usability testing: The stl usability evaluation workbench. In *Usability Metrics and Methodologies*, 1990. Proceedings of a one day meeting of the BCS HCI SG, 21 May.

[152] J. Whiteside, J. Bennett, and K. Holtzblatt. Usability engineering: Our experience and evolution. In M. Helander, editor, *Handbook of Human-Computer Interaction*, chapter 36. Elsevier Science Publishers (North-Holland), 1988.

[153] F. Wilson and J. Long. Evaluation methods: Supporting the development and usage of interactive systems, 1988. Tutorial presented at HCI88, the annual conference of the BCS HCI SG.

[154] D. Wixon, K. Holzblatt, and S. Knox. Contextual design: An emergent view of system design. In *CHI90 Proceedings*, 1990.

[155] P. Wright. Why is it so difficult to measure usability? In *Usability Metrics and Methodologies*, 1990. Proceedings of a one day meeting of the BCS HCI SG, 21 May.

[156] P. Wright and A. Monk. Evaluation for design. In A. Sutcliffe and L. Macaulay, editors, *People and Computers V: Proceedings HCI89*. CUP, 1989.

[157] P. Wright, A. Monk, and T. Carey. *Co-Operative Evaluation: The York Manual*, July 1989. Version 0.4.

[158] P. Wright and F. Reid. Written information: Some alternatives to prose for expressing the outcomes of complex contingencies. *Journal of Applied Psychology*, 57(2), 1973.

[159] R. Yasdi. A conceptual design aid environment for expert-database systems. *Data and Knowledge Engineering*, 1985.

[160] R. Young and P. Barnard. The use of scenarios in human-computer interaction research: Turbocharging the tortoise of cumulative science. In J. Carroll and P. Tanner, editors, *Human Factors in Computing Systems and Graphics Interface*, 1987. Proceedings CHI + GI 87.

[161] R. Young and P. Barnard. Multiple uses of scenarios: A reply to campbell. *ACM SIGCHI Bulletin*, 24(4), 1992.