Qualitative Reasoning about Dynamic Change in the Spatial Properties of a Physical System

by

Raman M. Rajagopalan, BSEE, MSEE, MSCS

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

December 1995

Qualitative Reasoning about Dynamic Change in the Spatial Properties of a Physical System

Approved by
Dissertation Committee:

I dedicate this dissertation to the memory of my uncle, Professor M. Venkataraman, member of the Indian Academy of Sciences, whose life represented an ideal balance of academic brilliance, relentless service, worldly achievement, and spirituality. If I can obtain just 1/4 of his knowledge and be 1/4 of the person that he was, my life would be fulfilled.

Acknowledgments

I must begin by acknowledging the contributions of my thesis advisor, Professor Benjamin Kuipers, and the members of my dissertation committee, Professors James Browne, Thomas Edgar, Donald Fussell, and Bruce Porter. Together, they provided many insightful comments and support, both financial and personal, to allow me to progress towards the completion of this dissertation. The suggestion of developing the Figure Understander, for example, came from Professor Kuipers, who also supplied the basic code upon which its ATN parser and text interpreter are built.

Next, I would like to acknowledge the contributions of Jimi Crawford, who built the Algernon system for knowledge representation, and Adam Farquhar, who built the QPC system for qualitative modeling. Both of these tools form the basic processing engines for the implementation of the theories presented in this dissertation. Similarly, I would like to acknowledge Professor Kuipers, who developed the QSIM qualitative simulation system, and Dan Clancy, Bert Kay, and David Throop, who have added many significant extensions. Without the availability of QSIM and its extensions, my examples could not have been executed.

During the course of one's doctoral work, peer reviews play an important role. At the University of Texas, many members of the Artificial Intelligence group provided helpful comments on both written and oral presentations of my work. In addition to all the persons already mentioned above, I would like to thank Professor Gordon Novak, Dan Berleant, Giorgio Brajnik, Dan Dvorak, David Franke, James Lester, Richard Mallory, Jeff Rickel, and Bob Schrag for their helpful comments and active encouragement of my work. Outside the University of Texas, I would like to thank Gautam Biswas, Sanjiv Bhatia, Tony Cohn, Christian Freksa, Klaus-Peter Gapp, Rasiah Loganantharaj, Wolfgang Maass, Paul McKevitt, Amitabh Mukerjee, Niel Rowe, and Rohini Srihari.

To complete one's work, the workplace has to run smoothly. I would like to thank Pat Horne, Fletcher Mattox, Kay Nettle, and Patti Spenser for helping to resolve any equipment problems, and Ariel Gamino, Lihn Pham, Phuong Pham, Bess Sullivan, and Gloria Ramirez for assistance in resolving administrative issues.

The greatest encourgement and support comes from one's friends and family. Although there are many friends who have provided immense support through my years at the University of Texas, I must especially thank Nandit Soparkar for pushing me towards achieving my goals. Closer to home, I would like to thank my parents (Meenakshi and Minakshisundaram Rajagopalan), sisters (Meera, Matangi, Shyamala, Poorna), and brothers-in-law (Krishnan, Ramesh, Vijay, Raja) for believing in me through both the good times and the bad times. My wife (Lalitha) and daughter (Jayashree) have waited patiently for me to finish this work. Jayashree, now it's your turn for attention from me.

Finally, I would like to thank our family's guru, Sri Gnanananda Giri Swamigal, who attained samadhi in 1974, but whose teachings live on. My experiences with him provided the most important type of support one needs to finish any complex task - faith and hope.

RAMAN M. RAJAGOPALAN

The University of Texas at Austin
December 1995

Qualitative Reasoning about Dynamic Change in the Spatial Properties of a Physical System

Publication No. ____________________

Raman M. Rajagopalan, Ph.D.

The University of Texas at Austin, 1995

Supervisor: Benjamin Kuipers

Spatial reasoning is an essential part of human interaction with the physical world. Of the many models that have been developed to support automated spatial reasoning, most rely on numerical descriptions of a spatial scene. This dissertation addresses problems where only qualitative descriptions of a spatial scene are available, such as natural language understanding, qualitative design, and physics problem-solving.

We provide the first set of solutions, given only a qualitative description of a spatial scene, for reasoning about dynamic change in both the spatial and non-spatial properties of a physical system. We use diagrams to *compactly* input the spatial scene for a problem, and text to describe any non-spatial properties. To match diagram and text objects so their descriptions can be integrated, we have developed a method for describing the conceptual class of objects directly in diagrams. Then, diagram and text objects can be matched based on their conceptual class.

The given problem is solved through qualitative simulation, and all spatial reasoning is done with respect to an extrinsic Cartesian coordinate system. We

model the relative positions of objects through inequality constraints on the coordinates of the points of interest. Changes due to translational motion are detected by noting changes in the truth values of inequality constraints. We model the orientation of an object through knowledge of its extremal points and its qualitative angle of rotation with respect to each coordinate axis. This model has been used to reason qualitatively about the effects of rotational motion, such as changes in the area projected by one object onto another.

We have implemented our spatial representation as production rules and as model fragments in the QPC qualitative modeling system. The former has been used for solving static-world problems such as understanding descriptions of an urban scene. The latter has been used to reason about situations where changes in spatial properties play a critical role, such as the operation of transformers, oscillators, generators, and motors. To support dynamic spatial reasoning, we have expanded the modeling capabilities of QPC to include methods for modeling piecewise-continuous variables, non-permanent objects, and variables with circular quantity spaces.

Contents

Acknowledgments **v**

Abstract **viii**

Chapter 1 Introduction **1**

1.1 The Need for Automated Spatial Reasoning 1

1.2 Properties of Spatial Models 2

1.2.1 Factors in Modeling Size and Shape 2

1.2.2 Factors in Modeling Relative Position and Orientation 4

1.3 What Problem is Solved in this Dissertation? 5

1.3.1 Limitations of Previously Developed Spatial Models 5

1.3.2 Problem Description 8

1.3.3 Statement of Thesis 9

1.3.4 An Illustrative Magnetic Fields Problem 10

1.3.5 Requirements on Our Qualitative Spatial Model 12

1.4 Summary of Contributions and Results 14

1.4.1 Modeling Spatial Extent, Position, and Orientation 14

1.4.2 Other Features of Our Spatial Model 18

1.4.3 Other Contributions - The Figure Understander 20

1.4.4 Implementation Architecture 23

1.5 Reader's Guide 25

Chapter 2 Modeling Spatial Extent **27**

2.1 Definition of a Spatial Object 27

2.2 Choices for Modeling Spatial Extent 28

2.3 Considerations for Choosing Shape Abstractions 30

2.3.1 Shape Abstractions Only Approximate Spatial Extent 30

2.3.2 Shape Abstractions Hide Information About Orientation . . . 33

2.4 Our Model of Spatial Extent 33

2.4.1 Shape Abstractions for Non-Rotating Objects 34

2.4.2 Shape Abstractions for Rotating Objects 34

2.4.3 Consequences of Using Two Different Shape Abstractions . . 36

2.4.4 How the Bounding Abstractions are Obtained 37

2.5 Previous Work 40

2.5.1 Shape Abstractions Used in Previous Work 40

2.6 Chapter Summary 44

Chapter 3 Modeling Position and Translational Motion **45**

3.1 Modeling the Absolute Position of Objects 45

3.2 Modeling the Relative Positions of Objects 47

3.2.1 Modeling Inequality Relations on Extremal Edges 49

3.3 Modeling Directional Spatial Relations 50

3.3.1 Modeling Directional Spatial Relations Between Two Objects 51

3.4 Modeling Topological Spatial Relations 52

3.4.1 Previous Work in Modeling Containment Relations 53

3.4.2 Our Model of Containment-Based Topological Relations . . . 55

3.4.3 Limitations of Our Model of Topological Relations 57

3.5 Modeling Translational Motion 57

3.5.1 Detecting Changes in the Relative Positions of Objects . . . 58

3.5.2 Modeling Changes in Area of Overlap 59

3.5.3 Race Conditions 63

3.6 Implementation of our Relative Position Model 63
3.6.1 Algernon Rules for Modeling Relative Positions 63
3.6.2 Support for Reasoning with Other Frames of Reference . . . 64
3.6.3 Modeling Spatial Relations Through QPC Views 69
3.7 Chapter Summary 75

Chapter 4 Modeling Orientation and Rotational Motion 77
4.1 Modeling the Orientation of Surfaces of Objects 78
4.1.1 Modeling Surface Tangent Directions 79
4.1.2 Definition of Clockwise and Counterclockwise 80
4.1.3 Modeling the Connectivity of an Object 81
4.2 Orientation and Rotation in 3-dimensional space 81
4.2.1 Modeling Rotation Around the X or Y axis 81
4.2.2 Rotation in the X-Y Plane 85
4.3 Modeling Vectors and Vector Operations 87
4.3.1 Modeling Vectors 88
4.3.2 The Qualitative Vector Cross Product Operation 89
4.4 Implementation of Our Model of Rotation 91
4.5 Chapter Summary 94

Chapter 5 The Figure Understander - Fundamentals 95
5.1 Why Use Diagrams and Text as Input? 96
5.2 The Coreference Problem 96
5.2.1 Overview of Our Goals and Solution 98
5.2.2 Related Research Issues 100
5.3 The Picture Semantics Description 101
5.3.1 Solving the Coreference Problem in Figure 5.2 103
5.3.2 Features of the Picture Semantics Description 104
5.4 The Figure Understander 106

5.4.1 Diagram Processing 107
5.4.2 Text Processing 107
5.5 A Scene Interpretation Example 110
5.5.1 Intrinsic/Deictic Frames of Reference 112
5.5.2 Annotated Demonstration 115
5.6 On Composite Objects 118
5.7 Evaluation of the Picture Semantics Approach 120
5.7.1 Applicability of Our Solution 121

Chapter 6 The Figure Understander - Implementation 123

6.1 The Input PostScript File 123
6.2 Selecting the Frame of Reference for the Problem 124
6.2.1 Restrictions on Automatic Selection of Frames of Reference . 124
6.2.2 Criteria for Automatic Selection of Frames of Reference . . . 126
6.2.3 Benefits of Automatic Selection of Frames of Reference . . . 126
6.2.4 Implementation of Frame of Reference Selection 127
6.3 Qualitative Description of the Initial Spatial Scene 128
6.3.1 Spatial KB Description of Object Boundaries and Orientation 129
6.3.2 Spatial KB Description of the Relative Positions of Objects . 133
6.4 Summary of Chapters 5 and 6 136

Chapter 7 Enhancements For Qualitative Modeling 138

7.1 Modeling Piecewise Continuous Variables 139
7.1.1 Modeling Piecewise-Continuous Variables in QPC 141
7.2 Modeling Piecewise-Continuous Entities 143
7.2.1 Modeling Piecewise-Continuous Entities in QPC 145
7.2.2 A Limitation of Our Solution 146
7.3 Modeling Circular Quantity Spaces 146
7.3.1 Definition of a Circular Quantity Space 147

7.3.2 Use of a Circular Quantity Space 148
7.3.3 QPC Implementation of Circular Quantity Spaces 150
7.4 Chapter Summary 151

Chapter 8 Demonstration and Evaluation 153
8.1 Static World Examples 154
8.2 The Implemented QPC Model Libraries 155
8.3 Overview of the Magnetic Fields Examples 156
8.3.1 Classes of Magnetic Fields Problems 157
8.4 Spatial Reasoning Isses Covered by the Examples 158
8.5 A Transformer Example 160
8.5.1 Problem Description 160
8.5.2 What Does This Example Illustrate? 162
8.5.3 Input Processing For This Example 162
8.5.4 The Models Produced by QPC 164
8.5.5 The Qualitative Behaviors For This Example 168
8.5.6 Final Knowledge Base Descriptions for the Objects 170
8.6 The Loop Moving Down the Track Example 170
8.6.1 Problem Description 171
8.6.2 What Does This Example Illustrate? 172
8.6.3 Input Processing For This Example 174
8.6.4 The Initial Model Produced by QPC 177
8.6.5 The Second Model Produced by QPC 178
8.6.6 The Remaining Models Created by QPC 181
8.6.7 The Qualitative Behaviors For This Example 183
8.6.8 Final Knowledge Base Descriptions for the Objects 187
8.7 An Oscillator Example 188
8.7.1 Problem Description 188
8.7.2 What Does This Example Illustrate? 190

8.7.3 Input Processing For This Example 191
8.7.4 The Models Produced by QPC 193
8.7.5 The Qualitative Behaviors For This Example 194
8.7.6 Final Knowledge Base Descriptions for the Objects 198
8.8 A Generator Example 198
8.8.1 Problem Description 198
8.8.2 What Does This Example Illustrate? 200
8.8.3 Input Processing For This Example 201
8.8.4 The Models Produced by QPC 202
8.8.5 The Qualitative Behaviors For This Example 205
8.8.6 Final Knowledge Base Descriptions for the Objects 207
8.9 Chapter Summary 207

Chapter 9 Conclusions **209**
9.1 Strengths and Limitations of Our Work 210
9.1.1 Our Model of the Relative Positions of Objects 210
9.1.2 Reasoning about Orientation 214
9.2 Directions for Future Work 215

Appendix A A Complete Magnetic Fields Example **219**
A.1 Description of the Example 219
A.2 Annotated Demonstration 221
A.2.1 Input Processing 222
A.2.2 Scenario Description Provided to QPC 225
A.2.3 The First Model Proposed by QPC 227
A.2.4 First Branch of the First Model - Constant Flux In Loop . . 230
A.2.5 Second Branch of the First Model - Increasing Flux In Loop . 232
A.2.6 The Model After the Transition 235
A.2.7 Third Branch of the First Model - Decreasing Flux In Loop . 238

A.2.8 The Behaviors Produced by Qualitative Simulation 240
A.2.9 The Final KB Descriptions of the Loop and Field 245
A.2.10 The QSIM QDE Produced by QPC for model M-2 246

Bibliography **253**

Vita **263**

Chapter 1

Introduction

1.1 The Need for Automated Spatial Reasoning

Spatial reasoning, reasoning about properties such as the shape, size, position, and orientation of objects, is an integral part of human interaction with the physical world. A number of tools have been developed by many research communities to assist human users with spatial reasoning tasks, and there is much interest in the continued development of such tools with even greater capabilities [55, 79, 2]. The following examples illustrate the broad interest in tools with spatial reasoning capabilities.

Computer aided design (CAD) tools [81] automatically keep track of numerical spatial constraints between objects, such as distances between objects. Geographic Information Systems (GIS) [51, 90, 54, 24] can track the movement of hurricanes and note the changes in the areas of forested and urban areas over time. Unmanned robots can make use of specialized mapping techniques to explore unknown environments [12, 45, 46]. Image databases are using retrieval techniques based on spatial models [8, 7, 52], and there are projects in natural language processing for producing natural language descriptions of a spatial scene [3, 39, 53, 77], and for visualizing a spatial scene based on natural language descriptions [29, 49, 56, 65].

Further, within the research area of Qualitative Reasoning [89, 43], the topic area of this dissertation, many spatial models have been proposed for performing a number of diverse tasks, including, but not limited to, the effects of kinematic constraints on objects in motion [14, 15, 23, 36, 40, 61], modeling qualitative vector algebra [88], reasoning about the relative positions and orientation of objects in an urban scene [27, 50, 57], and reasoning about changes in spatial relations due to the translation motion of objects [11, 28], or due to changes in the size and shape of objects [13].

1.2 Properties of Spatial Models

All of the tools mentioned above make use of spatial models to accomplish their goals. The basic properties captured by a spatial model include the size, shape, position, and orientation of objects. The capabilities and complexity of the previously developed spatial models vary greatly based on the needs of the application domain for which the model has been designed. Some application-dependent factors include: the availability and need for numerically precise descriptions, whether the problem space requires reasoning about one, two, or three dimensional objects, and whether the spatial configuration of the world is static or whether it can change dynamically.

To illustrate the breadth of choices involved in creating a spatial model, here are some examples of application-based constraints that have influenced previously developed spatial models. First, we will consider the spatial properties of size and shape, and then we will consider the properties of relative position and orientation.

1.2.1 Factors in Modeling Size and Shape

The simplest representation for modeling the properties of size and shape is to treat a spatial object as a point. This model is sufficient, for example, for the task of robot exploration [45, 46], since the relative positions and distances between landmark objects may be of more interest than the shapes, size, and orientation of the objects.

When an application requires a multi-dimensional model for the spatial extent of an object, there are two choices - using the exact shapes of objects or using a simplying abstraction which approximates the actual shape of an object. If the latter option is used in a model, then the criteria for selecting the abstract shape can become an important issue [72].

As examples, consider the application areas of constructing an automated tour guide and constructing a design aid which reasons about the interactions between gears. An automated tour guide requires a multi-dimensional model of objects so that their relative sizes can be taken into account in generating a description. As Retz-Schmidt points out in [77], while the description 'The bicycle is in front of the church' is generally an acceptable statement, the alternate description 'The church is behind the bicycle' is usually not acceptable.

In the tour guide application, knowledge of the relative sizes of objects is sufficient, and details about the exact shapes of objects are generally not required. In such applications, simplying shape abstractions, such as a rectangle, can be used to approximate the space occupied by an object.

If the problem domain is reasoning about the kinematic properties of objects, such as whether two gears will mesh together [23, 36, 62], then knowledge of the exact shapes and sizes of object is very important. To model the exact shapes and sizes of objects, a numerical model may be necessary. While terms such as triangle and rectangle can be used to describe some shapes, many other shapes don't have a qualitative description in natural language. Further, terms such as triangle only describe the basic shape of an object. Quantitative information may be required, such as the lengths of edges or the internal angles of a triangle, to uniquely describe the actual shape of an object.

1.2.2 Factors in Modeling Relative Position and Orientation

Once a choice has been made on the method for representing the spatial extent of an object, the next issue is how to model the spatial relationships between objects. The relative positions of two objects may be described in terms of either topological or directional relations. Topological spatial relations, such as overlapped and touching, are independent of orientation and do not require a frame of reference. However, the interpretation of many topological relations, such as near, far, and along, can be context dependent. What is 'far' is one context may be 'near' in another. Recently, there has been much work in the development of quantitative methods for evaluating the applicability of various topological relations within a given context [1, 29, 56, 65].

Directional spatial relations, such as left-of and right-of, require a frame of reference. Depending on the needs of the application, the frame of reference may be extrinsic (predesignated fixed frame of reference), intrinsic (based on the orientation of an object of interest, such as the intrinsic front of an object), or deictic (based on the orientation of the speaker or listener). Some applications, particularly those involving natural language processing [77], may require the capability to switch arbitrarily between different frames of reference.

In addition to the frame of reference, another important factor in modeling the relative position of objects is the basis of the comparison. Some spatial models used for retrieval from image databases [8, 52], for example, describe the spatial extent of an object through a minimal bounding rectangle drawn around the object. However, for the purposes of describing the relative positions between objects, only the centers of the bounding rectangles are compared. By effectively reducing objects to a single point, directional spatial relations can be easily defined through inequality relations on individual points. The tradeoff is that some spatial properties, such as the existence of overlap between objects, cannot be addressed.

When an application requires the positions of two multi-dimensional objects to be compared, if the model uses exact shapes, relations such as right-of and overlapped may need to be defined quantitatively, such as the Configuration Space-based definitions used in much of the work on qualitative reasoning about mechanical constraints [14, 15, 23, 36, 62].

If simplifying abstract shapes (i.e., 'idealizations' [35, 47]) are used to describe the bounds of objects, then it becomes easier to develop qualitative definitions of directional and topological spatial relations that take advantage of the simpler properties of the chosen abstract shapes. For example, to determine whether a bird is in a tree, Herskovits [35] suggests that people may draw a circular boundary around the branches of the tree, and answer 'yes' if most of the bird lies within the boundary.

However, when idealizations are used, some precision is sacrificed, and conclusions may be reached that are not true of the underlying physical world. For example, if the bird in not sitting on a branch, a human observer may not consider it to be in the tree. However, if the only definition for 'within' available to an automated reasoning system is 'enclosed in the bounding circle abstraction', then such a system could still conclude that the bird is in the tree.

1.3 What Problem is Solved in this Dissertation?

Thus far, we have suggested that many spatial models exist for use in as many applications. The reader might ask, what questions remain to be solved and what contributions are made in this dissertation?

1.3.1 Limitations of Previously Developed Spatial Models

To answer these questions, consider the following observations about the previously developed spatial models. The previous work can be roughly divided into two categories: work toward the development of spatial models to support specific, complex real-world applications (e.g., [23, 53]), and work toward the development of formal, intuitive theories for qualitative spatial reasoning (e.g., [25, 33, 11]).

The most powerful spatial models are associated with the former category, and allow reasoning about both static worlds and worlds where is translation and rotational motion. They allow reasoning about both the relative positions and orientation of objects, and can handle both 2 and 3-dimensional objects. The drawback is that these models require quantitative descriptions of the spatial scene, such as a video image [53] or a configuration space representation [23].

The models in the second class have the advantage of being both intuitive and elegant, but each of these qualitative models addresses only a small subset of the many spatial reasoning issues. Further, each of these models has been tested only on small examples, and their applicability for solving complex problems in larger reasoning systems is yet to be demonstrated. For example, Cui, Cohn, and Randell [11] provide an axiomatic description of topological spatial relations such as disconnected, touching, and enclosed, but cannot reason explicitly about motion. Their theory allows one to determine all possible spatial states that may follow a given state, but does not include a method for using more specific information, such as a motion vector, to narrow the possibilities. Similarly, Freksa [27] focuses only on the issue of reasoning about the orientation of objects.

Further, many of the formal models appear to be qualitative in nature, but an actual reasoning system may require a fair amount of precise metric information to use the definitions. The definitions often assume that some particular piece of knowledge is available, when in fact, the information may have to be computed using numerically precise data.

Accidental dependencies on very precise metric information can defeat one of the purposes of a qualitative formalism, which is to allow reasoning about a physical situation even when absolute knowledge of the values of variables is not available. However, the existing formalizations are still a good starting point for further work, since they succeed in mapping many physical situations into a much smaller set of qualitative situations. This facilitates reasoning about the most important issues and allows one to abstract away unimportant details.

As examples of accidental dependencies of quantitative knowledge, consider Cui, Cohn, and Randell's [11] assumption that a 'convex hull' function is available to provide the convex hulls of the objects in a scene. Since they allow objects to be of any shape, the convex hull function will require a quantitative description of the shape of each object. Hadzilacos and Tryfona [31] also assume that knowledge of the boundaries of objects is available, without providing a method for modeling these boundaries.

Another example is the collision parallelogram representation of Mukerjee and Joe [57]. Projections called lines of travel are drawn parallel to the surface normal direction of the intrinsic front of an object. When the lines of travel of two objects intersect, they form a parallelogram. Mukerjee and Joe provide a rich theory of how directional relations between two objects may be derived given knowledge of the positions of objects and the collision parallelograms. Unfortunately, to obtain the collision parallelograms, quantitative information about the initial position of the objects in the scene will be necessary. Further, Mukerjee and Joe can only reason about static scenes and scenes where motion is limited to translational motion along the intrinsic front direction of each object, since there is no means provided for qualitatively recomputing the location and shape of each collision parallelogram if objects are allowed to engage in any other types of motion.

In describing the CITYTOUR system, Retz-Schmidt [77] describes an algorithm for reasoning about directional spatial relations between a primary object and a reference object based on the intrinsic or deictic frames of reference provided by the reference object. CITYTOUR's inference method calls for drawing right, left, front, and back half-planes around the the intrinsic front or accidental front (in deictic use) of the reference object, and determining the half-plane in which the primary object lies. CITYTOUR's methods succeed because of the assumed availability of a video image, which provides sufficient metric information to quantitative compute the half-planes of interest, and to determine the half-plane in which the primary object lies.

1.3.2 Problem Description

For many important problems, only qualitative descriptions of the spatial state may be available. Example problems include understanding natural language discourse, where the speaker and listener operate with mental images; tutoring and training, where an instructor or a word problem may focus on abstract concepts instead of numerical details; and qualitative reasoning (e.g., qualitative design, diagnosis, or simulation) about the physical world.

For other problems, the goal may be prove a general result about the behavior of a class of objects. For example, many textbooks problems [76], such as the one described in Figure 1.1, test the student's understanding of a concept based on idealized cases. In Figure 1.1, for example, none of the objects in the example have a complex shape. However, it is well understood that the results apply to the many, more complex, real world examples that have the same qualitative description as the idealized problems in the textbooks.

The modeling and simulation techniques developed to date for Qualitative simulation [43] guarantee that all the real behaviors that can follow from a qualitative description of an initial state will be produced. Since a single qualitative description can be used to model an entire class of real-world initial states, the set of qualitative behaviors produced from a qualitative simulation provides a more general result than a numerical simulation, whose results may apply only to the case at hand.

This dissertation considers the problem of developing an intuitive qualitative spatial model which does not require precise quantitative descriptions of a spatial scene, and yet, can be applied to the task of reasoning about the relative positions and orientation of objects in both static worlds and worlds where objects are allowed to translate and rotate. Further, as this work has been performed within a larger effort to develop techniques for qualitative modeling and simulation of real world phenomena [17, 26, 38, 43, 44, 78], two other needs arise: to develop a spatial model that supports integrated reasoning about simultaneous dynamic change in both the spatial and non-spatial properties of a given system, and to develop a model which guarantees that all real behaviors achievable from an initial state will be produced. To date, none of the previously developed qualitative spatial models, except [22, 82, 87], which deal only with steady-state mechanisms, address these needs.

1.3.3 Statement of Thesis

This dissertation provides the first set of solutions, given only a qualitative description of the spatial state, for complex integrated reasoning about dynamic change in both the spatial and non-spatial properties of a system. We have developed a new representation for qualitative spatial reasoning which incorporates many of the ideas in previous work for reasoning about the spatial extent, relative position, and orientation of objects, and yet, is compatible with existing methods [9, 17, 20, 42] for qualitative modeling and simulation of physical systems. Our spatial representation allows qualitative reasoning about both static worlds and worlds with spatial change due to translational or rotational motion.

In the chapters that follow, we shall demonstrate the simplicity, generality, and power of our approach by presenting two independent implementations of our spatial representation, and illustrating their use to solve problems from two application areas, natural language understanding and the qualitative simulation of physical systems. More specifically, our methods have been applied to solve qualitative textbook problems [76] from the magnetic fields domain [68, 75, 71], and to solve problems in understanding natural language descriptions of an urban scene [70, 73].

The magnetic fields problems are solved through qualitative simulation, and

- **Diagram:**

754 / Faraday's Law of Induction

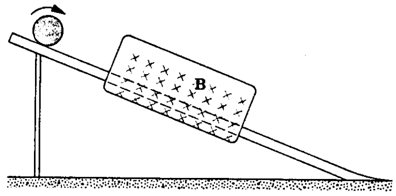


Figure 13 Question 7.

- **Text:** Figure 13 shows an inclined wooden track that passes, for part of its length, through a strong magnetic field. You roll a copper disk down the track. Describe the motion of the disk as it rolls from the top of the track to the bottom.

Figure 1.1: A problem from Resnick and Halliday's Physics textbook, *Fundamentals of Physics* [76].

the implementation makes use of the QPC [9, 17] and QSIM [42, 43] qualitative reasoning systems, and the Algernon [10] knowledge representation system. This implementation architecture will allow us to inherit additional capabilities as they are developed for the underlying systems, such as methods for incorporating incomplete quantitative knowledge into the reasoning process [44, 38].

1.3.4 An Illustrative Magnetic Fields Problem

This dissertation research was inspired by a simple observation: Although there has been much work in spatial reasoning and in qualitative simulation, there has been little progress in solving problems involving qualitative reasoning about simultaneous changes in both the spatial and non-spatial properties of a physical system.

For example, Figure 1.1 describes a magnetic fields problem where the reader is asked to describe what happens as a conducting disk rolls down a track into a magnetic field which is pointed into the page. To solve the problem, the reader must first keep track of the changes to the spatial configuration as the disk rolls down the track. Then, the reader must determine what magnetic fields processes may become active due to the changes in the spatial configuration, and determine the effects of these processes.

In this problem, the disk is initially to the left of the magnetic field, and is rolling down the track. The reader needs to recognize that the next qualitatively interesting state occurs when the disk begins to enter the magnetic field. This causes the magnitude of the magnetic flux passing through the disk to increase, resulting in an induced electromotive force (emf) in the disk which in turn causes a counterclockwise current flow. A magnetic force, whose magnitude depends on the strength of the field and the strength of the generated current flow, then acts on the right side of the disk, resulting in a net force that is opposite to the direction of motion. The force slows the acceleration of the disk through the field.

Then, the next qualitatively interesting state occurs when the disk is fully enclosed in the field. Because the flux through the disk no longer changes, no dynamic processes, other than motion, are active until the disk reaches the right boundary of the field, and begins to exit the field. At this point, the area of overlap between the disk and the field begins to decrease. In turn, the magnitude of the flux through the loop also decreases, and an induced emf is established in the loop which results in a clockwise current flow. A resultant magnetic force, opposite to the direction of motion, now acts on the left side of the disk, and slows the acceleration of the disk as it moves out of the field.

An additional effect is that a new magnetic field is created whenever there is current flow in the disk. This magnetic field serves to oppose the change in flux through the disk. Thus, while the disk is entering the permanent field given in Figure 1.1, the new magnetic field generated by the current flow in the disk will have a direction pointed out of the page within the disk, and pointed into the page outside the disk. This new magnetic field is not permanent, and ceases to exist once the disk is fully enclosed in the permanent magnetic field and the current flow in the disk stops.

When the disk is exiting the permanent magnetic field, and a clockwise current flow is established in the disk, a new magnetic field is again created, with its direction pointed into the page inside the disk, and out of the page outside the disk. In this example, the newly created magnetic fields simply oppose the change in the magnetic flux through the disk and play no other active role. However, in other devices, such as a transformer, dynamically created magnetic fields play an important role and have to be considered explicitly.

1.3.5 Requirements on Our Qualitative Spatial Model

Let us now consider the requirements on a spatial model to solve problems such as the one given in Figure 1.1. First, the model should provide a **multi-dimensional representation of the shape/spatial extent** of objects to allow reasoning about whether there is overlap between objects. Then, the model must allow reasoning about both **topological relations**, such as the area of overlap between objects, and **directional spatial relations**, such as the fact that the disk is initially above and left of the field and is rolling towards the field.

The model must allow reasoning about changes in both topological and directional relations as objects engage in **translational motion**. It must also be possible to describe the **orientation** of objects to track such things as the direction of motion of the disk, the direction of the magnetic field, the direction of current flow in the disk, and to determine where the net magnetic force acts on the disk.

Further, the spatial model must support **vectors and vector operations** such as the cross product. The latter is required, for example, to determine the direction of the magnetic force acting at a given point on the disk. This is determined by the equation $\mathbf{F} = i\mathbf{l} \times \mathbf{B}$, where $\mathbf{l}$ is a vector that gives the direction of current flow at the point of interest, and $\mathbf{B}$ is a vector that describes the magnitude and direction of the magnetic field at the point of interest.

Finally, the model must not assume that spatial objects in the scene are permanent. It needs to support the **dynamic creation and destruction of spatial objects** to allow the fact to be captured that the magnetic fields generated by current flow in an object could appear and disappear dynamically.

The requirements given above are based on one example, but, as we shall show later, serve to cover the spatial modeling needs for reasoning about many problems involving static scenes or those where there is only translational motion, including problems in natural language understanding involving reasoning with intrinsic and deictic frames of reference. However, in our work, we also wish to provide the capability to reason about such devices as generators and motors, where there is a need to reason about rotating objects.

To support reasoning about rotation, the following additional requirements are placed on our spatial model. Since rotation can change both the position and orientation of an object, the choices for modeling these properties will have to be made more carefully. For example, consider terms such as clockwise and counterclockwise. These terms require a perspective. We take the perspective of always viewing objects from above (positive infinity on the Z axis).

With this definition, let's assume that there is clockwise current flow in a closed conducting object, and that the object is rotating around the X-axis of the chosen Cartesian coordinate system for the problem. Then, at some point, the orientation of the object will become *inverted*, and what appeared to be clockwise current flow before will now appear counterclockwise when viewed from the same perspective.

Another property of interest is that in devices such as a motor, the cause of rotation is a magnetic torque on a conducting object placed in a magnetic field. As the object rotates, the magnitude of the torque can change, as well as the magnetic flux passing through the object. The latter change occurs because the projected area of the object onto the plane of the magnetic field changes, thereby affecting the lines of flux passing through the object. This change in flux through the object can in turn result in an induced emf in the object, and subsequently, a current flow. To support these type of reasoning, the spatial model must provide the capability to describe **torque** and its relationship to **rotational motion**. It must also be capable of tracking **changes in the projected area of overlap** between two planes as objects rotate.

1.4 Summary of Contributions and Results

When considered as a whole, the many spatial models that have been developed to date cover most of the requirements on our spatial model as described in the previous section. However, no single model, particularly any existing qualitative spatial model, can cover all of these modeling issues.

1.4.1 Modeling Spatial Extent, Position, and Orientation

We have succeeded in developing a single qualitative spatial model that can be used for reasoning about all the requirements described in Section 1.3.5. Our success is based on the following observations:

1. First, we recognize that the spatial property of orientation is a local property of an object which is independent of other objects. The concept of orientation is based on the direction faced by an actual surface of an object, such as the surface which forms the intrinsic front. This observation means that if a spatial model uses an abstract shape to approximate the actual shape of an object, then there must be some other means for keeping track of the orientation of an actual surface, since the orientation of the abstract shape need not correspond to the orientation of the underlying object.

2. On the other hand, the spatial property of relative position is a global property that depends on the positions of other objects. In the problems of interest to us, interesting events occur only when there is overlap between two objects, such as a conducting object and a magnetic field. Therefore, to cover all the interesting behaviors, we only have to detect when it is *at all possible* for there to be overlap between two objects. This allows us to use simplying shape abstractions to approximate the position and spatial extent of each object, but requires that the chosen abstract shape must fully cover the actual spatial extent of the object at all times.

3. Our problems require reasoning about both translational and rotational motion, with the restriction that the latter is limited to the rotation of an object about an axis passing through its center of gravity. Translational motion can change the position of an object, but not its orientation. Unless it is circular in shape, rotation can change both the position and orientation of an object.

4. Without quantitative information, particularly regarding the distances between objects, the problem of determining whether the rotation of a non-circular object about its center of gravity can cause new intersections with other objects, or remove existing intersections, is an inherently ambiguous problem. For example, in Figure 1.2, if object 'A' rotates around the Z axis, no new intersections are possible. However, if object 'B' rotates around the Z axis, then an intersection with object 'A' is possible.

   To avoid this problem, unlike any previous work, we allow a choice of two possible abstract shapes for each object, as illustrated in Figure 1.2. If an object is fixed in space, such as a permanent magnetic field, or if it can only engage in translational motion, we use a minimum bounding rectangle drawn around the extremal points (i.e., its rightmost, leftmost, topmost, and bottommost points) of the object, such as that given for object B in Figure 1.2.

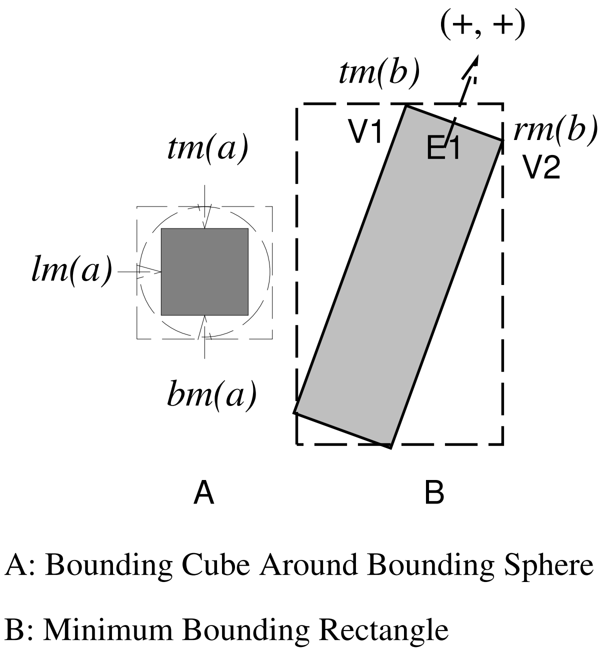


Figure 1.2: Examples of minimum bounding rectangles and bounding spheres as abstractions of shape for two polygons.

If an object can rotate, since we only allow rotation about an axis passing through its center of gravity, we draw a bounding sphere around the object which covers all the points that could be touched by the object due to any rotation. We then draw a bounding cube around the extremal points of this spherical shape, as given for object A in Figure 1.2. For the purposes of reasoning about the relative positions between objects, the rectangle projected by the bounding cube onto the X-Y plane is used.

This assures that all objects have the same simple bounding shape, a rectangle, for reasoning about position. However, by following the methods given above, we also ensure that only translational motion can change the position of an object, and that only rotation can change its orientation.

5. As we shall show in subsequent chapters, it is straightforward to write qualitative constraints for determining the relative positions of two 'boxed' objects,

and to detect possible changes due to any translational motion.

6. For reasoning about orientation, such as the direction of current flow along a given surface of an object, the orientation of the chosen abstract shape cannot be substituted. There needs to be a means of keeping track of the actual orientation of the underlying object.

   It can be observed that for convex objects, knowledge of the extremal points of an object is sufficient for determining the qualitative orientation of any surface of the object. We define the qualitative orientation of a surface through its surface normal vector. In Figure 1.2 the qualitative orientation of edge 'E1' for object 'B' is (+, +), since its surface normal vector, like any other surface which lies between the topmost and rightmost extremal points of an object, has positive components along each of the X and Y directions.

7. If an object cannot rotate, the identities of its extremal points will be fixed. If the object can rotate, to maintain knowledge of its qualitative orientation, we only need to keep track of changes in the identities of the edges and/or vertices which form its extremal points as the object rotates.

   Currently, our model of 3-dimensional rotation assumes that objects are 2-dimensional and that all objects are initially in the X-Y plane. If an object rotates around the X-axis or Y-axis, it can become inverted. That is, the identities of its topmost and bottommost points, or its rightmost and leftmost points, respectively, could be switched. By detecting when these events take place, based on the angle of rotation around each axis, we can easily maintain knowledge of the orientation of the object.

   Rotation around the Z-axis is more complex because the identities of any of the four extremal points, topmost, bottommost, rightmost, and leftmost, can change. For a general polygon, with only qualitative information, determining which extremal point changes first is an inherently ambiguous problem.

However, if the underlying objects has a special shape, such as a rectangular shape, then rules can be provided to uniquely determine the order in which the identities of the extremal points change during rotation.

1.4.2 Other Features of Our Spatial Model

The previous section outlined our basic strategy for modeling the spatial extent, position, and orientation of objects. In addition to these basic features, our spatial model provides the following additional capabilities.

First, we have implemented methods for reasoning about vectors and vector operations. Previously, Weinberg, et. al. [88] and Nielsen [61, 62] have both described techniques for reasoning about vectors. Our work extends their contributions by providing the ability to qualitatively perform the vector cross product operation.

Second, to complement our basic methods for reasoning about rotation, we provide the ability to reason about the projection of one plane onto another based on the angle of rotation between the two planes. To reason about the existence and magnitude of torque due to a magnetic forces, we also have the ability to track the qualitative distance between two equal and opposite forces acting on an object. Third, we have implemented the ability to directly model temporary spatial entities, such as a magnetic field created by current flow in an object. This complements the capabilities of existing qualitative modeling systems such as QPC [9, 17], which allow only continuous objects to be modeled. Our techniques will now allow 'piecewise-continuous' entities to be modeled directly in the framework of these systems.

Fourth, we recognized that when reasoning about dynamic changes in spatial relations, some parameters of interest could be piecewise-continuous. For example, assume that a rectangular object, 'A' is moving with a steady velocity a rectangular object 'B', As object A enters object B, the area of overlap between the two increases steadily. That is, the derivative of the area of overlap between the two objects has a positive value. However, once object A becomes fully enclosed in object B, the change in the area of overlap drops *discontinuously* to zero.

Similarly, consider a closed conducting loop of wire that is initially in the X-Y plane and is rotating about an axis parallel to the X-axis and passing through its center of gravity. Assume that there is a clockwise current flow in the loop, as viewed looking down at the loop from the positive Z axis. When the loop has rotated ninety degrees, its orientation will be inverted, and the current flow will appear counterclockwise. If clockwise current flow is considered positive, there will be a discontinuous change in the value of the current after a ninety degree rotation, since its value will go from positive to undefined to negative without passing through zero.

The existing qualitative modeling and simulation systems, such QPC [9, 17], QPE [21], and QSIM [42, 43], do not provide the capability for directly modeling and reasoning about piecewise-continuous parameters. We have also developed techniques to allow 'piecewise-continuous' variables to be modeled directly in the framework of these systems.

A fifth contribution arises from the observation that qualitative vector algebra can be inherently ambiguous when vectors are in free space. However, for many cases, unambiguous results can be obtained, even for the cross product operation, when vectors are aligned with the coordinate axes. Thus, we have developed heuristics to automatically select a frame of reference (Cartesian coordinate system) for an initial scene which maximizes the number of edges of objects and vectors that are aligned with the coordinate axes. For example, for the problem in Figure 1.1, we first rotate the diagram so that the track is aligned with the X-axis of the coordinate system. This simplifies the problem by aligning all vectors of interest with the coordinate axes. Our heuristics give priority to immobile objects and permanent vectors since their orientation will remain fixed for the problem to be solved. This complements the work of Nielsen [62] who suggests similar heuristics, but does not appear to have tested the ideas through an implementation.

Finally, for problems requiring reasoning about intrinsic and deictic frames of reference, we have implemented a limited capability to automatically translate a query to prove a spatial relation given in terms of an alternate (e.g., intrinsic, deictic) frame of reference to an equivalent query in terms of the extrinsic Cartesian coordinate system chosen for the initial scene. This capability is demonstrated in our examples involving the processing of natural language descriptions of a spatial scene.

1.4.3 Other Contributions - The Figure Understander

One important question for an implemented spatial model is, 'What is the form of the input?'. We start with a diagram that describes the spatial scene, and text that describes additional problem-specific properties of the objects in the scene, and extract a unified qualitative description of the information in each form of input. This is accomplished through an independent tool which we call the Figure Understander [74, 70, 69, 73]. Once the input has been processed, if the problem involves dynamic change, we use a QPC-based implementation of our spatial model to solve the problem through qualitative simulation [71]. If the problem involves a static scene, we can instead use an Algernon [10] rule-based implementation of our spatial model to solve the problem [70, 73].

The use of diagrams to describe the spatial scene serves two purposes. One is convenience. We extract a lot of spatial information from a diagram, and as the popular saying goes, 'a picture can be worth a thousand words' [48]. The second is to meet a necessity. As described in the previous section, it is sometimes necessary to rotate the frame of reference of the input scene to maximize the number of vectors and edges of objects that are aligned with the coordinate axes. Whereas rotating a diagram numerically is a well understood procedure, rotating a diagram starting with a qualitative description is non-trivial and is likely to yield many ambiguities [67]. We found that although some general qualitative solutions exist, these solutions require specialized information about the scene that is unlikely to be available unless a diagram is also available. Therefore, we have opted to numerically rotate the diagram when necessary.

The result of the input processing done by the Figure Understander is a qualitative description of the spatial scene. This qualitative description is all that is used for all further reasoning.

Integrating Text and Diagram Input

Although we use the Figure Understander only as a tool for processing an input problem description, it solves a significant research issue in its own right. The issue of integrating the differing information in diagram and text input is a non-trivial problem that has been attacked by many researchers [55, 59, 18, 32, 63, 64, 80, 83, 85, 84].

The basic issue is that of establishing coreference [63] between object references in text with objects in a diagram. Solutions to the coreference problem are important because the unique information in each form of input, diagrams and text, can be integrated only after the mappings between diagram and textual objects are identified.

To solve the coreference problem, human readers can make use of a large "commonsense" knowledge base to quickly visualize the spatial properties of an object mentioned in a text description and identify the appropriate object(s) in a picture. Similar strategies have been adopted in previous work to automate the ability to solve the coreference problem. One approach is to use model libraries which map domain-specific terms to a spatial model for the object [64], such as the model that a 'car' is a rectangle with four circles attached to it. Another approach is to use an image database to directly associate a picture with a textual term [80, 84].

Both approaches face problems with the completeness of the model library or image database and ambiguities in assessing the quality of a match against a given input. Unfortunately, a given object or collection of objects in a diagram may map to several models in a model library, or objects in the input may have a different shape or orientation, and thus appear to be different, than objects in an image database.

In previous work, researchers have addressed the problem of automatically interpreting text and pictorial input *intended for human use*, such as Srihari's interests in identifying persons in newspaper photographs based on clues in an associated text caption [84], or Novak and Bulko's [64] interest in solving textbook Physics problems using the exact text problem statement and associated diagrams that are available to the human reader. These problem assumptions do not allow any automation-friendly modifications in either the diagram or text input, thus preserving the full complexity of the coreference problem, and making the solutions very difficult to implement and use.

Our goal is to provide a simpler methodology which allows the combined use of text and diagrams to compactly provide the spatial and contextual properties of a scene as input to a given system [74, 69, 73]. We have developed a unique solution to the coreference problem that is far simpler than any existing solution. Whereas the previous work in the area of integrating diagram and text input specifically addresses interpreting descriptions intended for *human use*, our solution has been designed to allow diagrams and text to be used as *computer input.*

Our solution is based on the following observation. In the textbook diagram given in Figure 1.1, note the object with a '+' pattern. This is a standard convention used in textbooks to denote a magnetic field with a direction pointed into the page. This pattern allows the reader to immediate associate the diagram object with the term magnetic field in the text.

Our solution makes uses of a special description language we have developed, the **Picture Semantics**, to associate problem and domain-specific contextual information with the use of specific patterns, colors, and brush types in diagrams. Thus, when extracting information from a diagram, we can not only obtain spatial information, we can also obtain non-spatial, domain-specific, contextual information directly from the diagram. We show in subsequent chapters that this greatly simplifies the solution of the coreference problem.

1.4.4 Implementation Architecture

An outline of our implementation architecture is shown in Figure 1.3. We start with a diagram and a text description which specifies a problem, and use the Figure Understander to obtain a unified knowledge base description of the information in each form of input. Then, if the problem involves reasoning about motion, an input description compatible with the QPC qualitative modeling and simulation system [9, 17, 16] is generated, and a solution is obtained through qualitative simulation.

All information is maintained within an Algernon [10] object-oriented knowledge base. All objects in an Algernon knowledge base must be instances of object classes. The objects classes in Algernon are defined through a *taxonomy*. Relations (*slots*) and forward and backward chaining inference rules can be defined on these classes.

The taxonomy and slot definitions used in our work come from three sources. The first source is the background knowledge base defined by QPC (henceforth, the *QPC KB*). The second source is an Algernon knowledge base we have developed to support our spatial model (henceforth, the *spatial KB*). The third source is an application-specific domain model which the user of our spatial reasoning system is expected to provide (henceforth, the *domain KB*).

The Figure Understander makes use of several specialized components to process the diagram and text input. These components include an application-specific *Picture Semantics* description for the diagram, an ATN grammar for parsing the text, an application-specific text interpreter, and the QPC, spatial, and domain

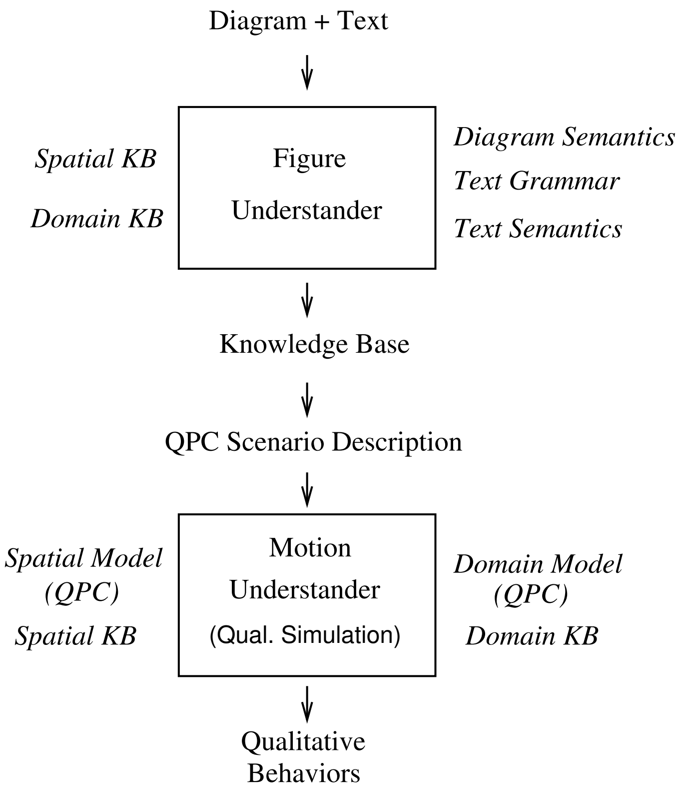


Figure 1.3: The architecture of our spatial reasoning system.

KBs. The output of the Figure Understander is an Algernon knowledge base description of the initial state expressed in terms of the object classes and relations defined through the QPC KB, spatial KB, and domain KB.

If the problem to be solved involves spatial reasoning about a static initial scene, the Algernon rule-based implementation of our spatial model provided in the spatial KB is used to prove queries on spatial relations about the initial state. If the problem to be solved involves reasoning about dynamic change, then qualitative simulation is performed using QPC to determine the qualitative behaviors that are achievable from the initial spatial state. The spatial KB and a QPC-based implementation of our spatial model are used in conjunction with the domain KB and the QPC domain model to initialize and perform the qualitative simulation. We will henceforth refer to the conjunction of the spatial KB and the QPC implementation of our spatial model as the *Motion Understander*.

1.5 Reader's Guide

The remainder of the dissertation is organized as follows:

- Chapters 2-4 describe the theory behind our spatial representation.

  Chapter 2 describes our methods for choosing bounding abstractions for objects. These are used to approximate the position and spatial extent of objects for the purpose of reasoning about their relative positions.

  Chapter 3 describes our methods for modeling the relative positions of objects and for reasoning about the effects of translational motion.

  Chapter 4 describes our methods for modeling the orientation of objects and for reasoning about the effects of rotational motion.

- Chapters 5 and 6 describe the Figure Understander, our tool for integrating information in text and diagram input into a single unified knowledge base description.

Chapter 5 describes the theory behind the Figure Understander, and presents an example of its use for understanding diagram and text descriptions of an urban scene. In particular, we introduce the concept of Picture Semantics descriptions as a method for providing non-spatial contextual information directly in diagrams. We will show that this method greatly simplifies the solutions to the coreference problem in integrating text and diagram input.

Chapter 6 describes the implementation of the Figure Understander, with emphasis on its diagram processing capabilities.

- Chapter 7 describes our extensions to the modeling capabilities of the QPC qualitative simulation system. These include methods for modeling piecewise-continuous variables, for modeling objects whose existence may be piecewise-continuous, and for modeling variables with circular quantity spaces.
- Chapter 8 presents an empirical evaluation of the Figure Understander and our spatial representation. A number of examples are presented from the magnetic fields domain which illustrate the capabilities of our implemented spatial reasoning methods.
- Finally, Chapter 9 summarizes the contributions of this dissertation.

Chapter 2

Modeling Spatial Extent

Chapters 2 through 4 describe our spatial representation. Our goal is to develop a single qualitative model that can be used to reason about the spatial properties of spatial extent, position, and orientation, and how these properties can change due to translational and rotational motion.

This chapter defines the kinds of spatial objects we consider, and presents our methods for modeling the spatial extent (size and shape) of an object.

2.1 Definition of a Spatial Object

Two initial questions in defining a spatial representation are, 'What is space?' and 'What is an object?'. We consider two-dimensional, closed, convex objects that can have any orientation in three dimensional Euclidean space. One-dimensional objects, such as vectors, are allowed, but are not considered to be active objects - they can only provide additional information about the multi-dimensional objects in the scene. These restrictions are due to our model of orientation, which requires that objects are convex, and our model of rotation, which has been developed for two-dimensional objects. The latter's extensibility to cover three-dimensional objects has not been investigated. Our model of position and translational motion can cover three-dimensional and non-convex objects.

Our spatial representation assumes that a domain model is available which provides knowledge of certain properties of an object, such as whether the object is solid, or is permeable (liquid or gas). This property is important in our model of overlap, since two solid objects cannot overlap in space. Additionally, we use knowledge of whether the object is mobile or immobile. For example, in our domain model for magnetic fields problems, all permanent magnetic fields given in the initial scene are assumed to be permeable and immobile.

Finally, although we provide heuristics for choosing the frame of reference for a problem, and for reasoning with intrinsic and deictic frames of reference, ultimately, every frame of reference is grounded in an extrinsic Cartesian coordinate system. For convenience, all objects in the initial scene are assumed to be in the X-Y plane. This can change later due to the rotation of objects.

2.2 Choices for Modeling Spatial Extent

There are two choices for modeling the *spatial extent* (i.e, size and shape) of a multi-dimensional object: through its exact shape and size, or through a simplifying abstraction. In the general case, it is not possible to describe exact shapes and sizes qualitatively. While terms such as triangle and rectangle can be used to describe some shapes, many other shapes don't have a qualitative description in natural language. Further, terms such as triangle only describe the basic shape of an object. Quantitative information may be required, such as the lengths of edges or the internal angles of a triangle, to uniquely describe the actual shape and size of an object.

If a problem requires reasoning about the exact shapes of objects, such as whether two gears will mesh together, a solution approach which combines qualitative and quantitative reasoning, like the methods developed for reasoning about the qualitative features of a configuration space representation [23, 36], will be required.

Fortunately, there are many spatial reasoning problems which can be solved with only qualitative information about the spatial scene. Both Herskovits [35] and Landau and Jackendoff [47] have conducted extensive studies of human spatial reasoning that confirm that human models of spatial relations appear to depend mainly on the general boundedness, surface, or volumetric properties of an object. Herskovits has observed that we often use 'idealizations', such as points, lines, or simple bounded shapes, to approximate the sizes and shapes of objects. This is appropriate for many problems in natural language understanding, for example, where the goal may simply be to form a rough mental image that corresponds to a qualitative description of a spatial scene.

We observe that such approximations are also appropriate for many problems involving qualitative reasoning about the physical world. For example, to focus as closely as possible on the basic principles to be taught, Physics texts such as Resnick and Halliday's *Fundamentals of Physics* [76] often use examples where the objects in the scene have idealized, simple shapes, such as the problem described in Figure 1.1. Once learned, a student can use knowledge of these basic principles to reason about more complex problems.

Problems such as the one described in Figure 1.1 are ideal for automated qualitative reasoning systems which employ qualitative simulation, since one of the strengths of qualitative simulation is the ability to produce all the qualitative behaviors that can be achieved from a given initial qualitative state. The qualitative solution can be remembered and applied to more complex problems that have the same qualitative features since it has been shown [43] that qualitative solutions obtained from a given qualitative model and initial state generalize to cover the behaviors achievable from any physical initial state whose qualitative description matches the qualitative initial state.

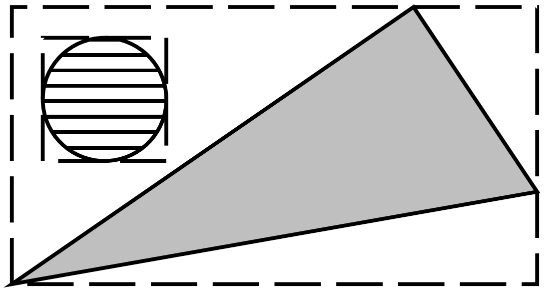


Figure 2.1: Example of two objects which do not intersect, but whose shape abstractions do intersect.

2.3 Considerations for Choosing Shape Abstractions

A spatial representation can benefit from the use of shape abstractions to hide unimportant details and to simplify the reasoning process. If a spatial model makes use of shape abstractions, choosing an abstract shape then becomes a fundamental issue. Clearly, a shape abstraction is useful only if it is easier to maintain than knowledge of the actual position and orientation of an object. Two other factors to consider are: (1) Shape abstractions provide only an approximation of the actual spatial extent of an object. (2) Shape abstractions hide details about the actual shape and orientation of an object.

2.3.1 Shape Abstractions Only Approximate Spatial Extent

Shape abstractions provide only an approximate, incomplete description of the actual spatial extent of an object. Therefore, it is possible for the shape abstractions of two objects to satisfy a spatial property, such as overlap, even though there may be physical situations when the actual positions of the objects do not satisfy the property.

For example, in Figure 2.1, the circular object is enclosed inside the rectangular shape abstraction chosen for the triangular object, but does not actually touch the triangular object. If spatial relations are proven only with respect to the shape abstraction chosen for an object, it is possible to conclude the circular object

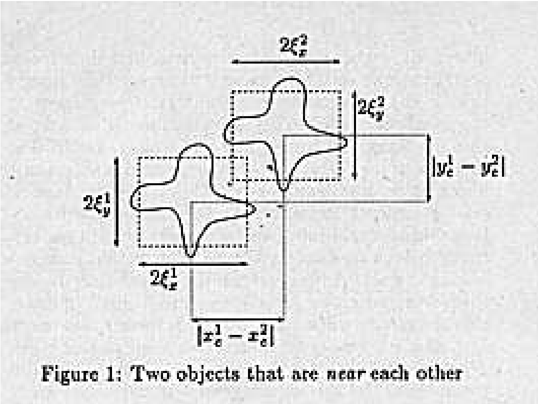


Figure 2.2: Figure from [1] used by Abella and Kender to illustrate their model of 'near'.

is overlapped with or enclosed within the triangular object even though it is not true of the actual spatial scene.

What Guarantees Can Shape Abstractions Provide?

Any spatial representation which relies on the use of shape abstractions should make it clear what conclusions are guaranteed to be true of the actual spatial scene, and which conclusions need not be true. Without an explicit statement of what each model guarantees to be true of the actual spatial scene, some of the conclusions based on a qualitative spatial model can be counterintuitive, or even false.

For example, Abella and Kender [1] define 'near' to be true when the rectangular shape abstractions of two objects intersect, as shown in Figure 2.2. However, visual inspection of their figures reveals that their method of drawing the rectangular shape abstractions does not always cover the object, and that it is possible for two objects to actually intersect, and not be considered 'near'.

Abella and Kender's model has been developed as a means of quantitatively constructing a 'fuzzy' representation for the boundaries of an object. Their model suffers from the fact that they can provide very few guarantees about whether spatial relations proven with respect to their 'fuzzy' objects are necessarily true of the actual objects in the scene.

The following observations may be made about the guarantees that can be provided by shape abstractions:

- If a shape abstraction completely covers an object at all times, it provides the guarantee that non-intersection conclusions are always true of the actual scene. That is, if two completely covering shape abstraction do not intersect, then it is impossible for the underlying objects to intersect.

  If the definition of directional relations such as strictly-right-of and strictly-left-of are based on the possibility of intersection due to movement along a given axis (e.g., such as the X-axis), then the guarantee extends to cover conclusions about these relations: if two completely covering shape abstractions do not satisfy a given relation, then the underlying objects will not either.

  When completely covering shape abstractions are used, it is possible that two bounding shapes intersect when the underlying objects do not. To reduce the possibility of false intersection conclusions, the bounding abstraction should be chosen to fit the underlying object as tightly as possible.

- If a shape abstraction is completely inside the actual extent of an object at all times, it provides the guarantee that conclusions of intersections between objects will always be true of the actual scene.

- If a shape abstraction neither covers the underlying object at all times, nor is covered by the actual object at all times, as is the case in Abella and Kender's model, then it may not be possible to guarantee any conclusions about the relative positions of objects to be true of the actual scene.

- The only way for a shape abstraction to guarantee both intersection and non-intersection conclusions to always be true of an actual scene is if the shape abstractions of objects exactly match their actual shape.

In the magnetic fields problems that are the focus of our work, interesting events occur only when a conducting object intersects or is enclosed in a magnetic field. Thus, the ability to guarantee that a conducting object and magnetic field are not overlapped will be useful since one could conclude with absolute certainty that no magnetic fields-related processes could occur.

This observation leads to the following **adequacy criterion** for evaluating potential shape abstractions for our purposes:

- Any abstraction for the spatial extent of an object should fully cover the actual extent of the object at all times.

2.3.2 Shape Abstractions Hide Information About Orientation

While shape abstractions can be useful in reasoning about the relative positions of objects, they may not provide sufficient detail about the shape of the actual object to reason about its orientation, and an additional representation may be required. For example, shape abstraction cannot be used to reason about either mechanical constraint problems, as considered in [61], or to reason about the direction of current flow at a given point on the surface of a conductor. This type of reasoning requires knowledge of the actual orientation of a given surface of an object, which is exactly the detail that is ignored when a shape abstraction is used.

In our work, we use a shape abstraction to simplify reasoning about relative positions, but provide an additional representation to maintain the actual orientation of each surface of an object.

2.4 Our Model of Spatial Extent

Our work involves reasoning about possible interactions and overlaps among extended objects in 2-dimensional and 3-dimensional space. We therefore require the use of shape abstractions that are at least two dimensional in shape, and whose positions can be maintained with qualitative information alone as objects translate and rotate. This observation, along with the adequacy criterion presented in the previous section, has led to the use of two shape abstractions in our spatial representation, minimum bounding rectangles for objects that are confined to the X-Y plane, and a bounding cube drawn around a bounding sphere for objects that are free to rotate about their center of gravity in 3-dimensional space.

2.4.1 Shape Abstractions for Non-Rotating Objects

For immobile objects, and for mobile objects in problems where there is no motion or only translational motion, we use a minimum bounding rectangle drawn around the **extremal points**, i.e., the `rightmost`, `leftmost`, `topmost`, and `bottommost` points, of an object. The extremal points of an object are defined with respect to the extrinsic Cartesian coordinate system that has been chosen for the problem. If an extremal surface of an object is formed by a edge, then any point on that edge can be used as the extremal point. This representation is illustrated by the bounding rectangle shown for object 'B' in Figure 1.2, which for the reader's convenience has been redrawn as Figure 2.3. Currently, we only consider 2-dimensional objects. To extend this representation to cover 3-dimensional objects, a minimum bounding cube could be drawn using two additional extremal points, the `frontmost` and `rearmost` points.

2.4.2 Shape Abstractions for Rotating Objects

The only type of rotation we consider is the rotation of an object about an axis passing through its center of gravity.

For mobile objects that may be rotating, or which are free to rotate, we use a bounding cube drawn around the extremal points of the minimum bounding sphere that completely covers all points that may be touched by the object due to any rotation about its center of gravity. This is illustrated by the bounding abstraction

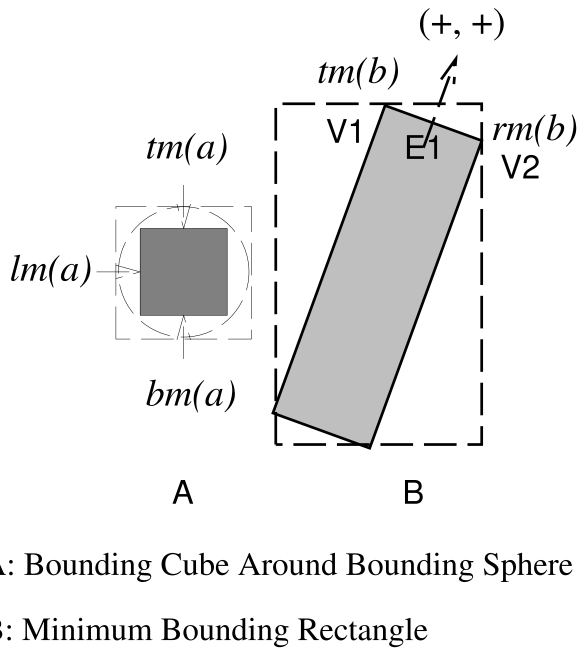


Figure 2.3: Examples of minimum bounding rectangles and bounding spheres as abstractions of shape for two polygons.

shown for object 'A' in Figure 2.3.

For reasons discussed later in Chapter 3, our models of directional and topological spatial relations only consider two-dimensional objects in the X-Y plane. Therefore, for the purposes of reasoning about the relative positions between objects, we use the rectangle projected by its bounding cube onto the X-Y plane to represent the spatial extent of the object. Unless the underlying object is circular in shape, this projected rectangle is always larger, and therefore weaker, than a minimum bounding rectangle drawn around the actual extremal points of the object. Thus, it is more likely that false intersection conclusions will be reached.

However, this weaker abstraction is required because for rotating objects, the minimum bounding rectangle violates the adequacy criterion. Consider object 'B' in Figure 2.3, and the minimum bounding rectangle drawn about its extremal points. Assume that it is rotating clockwise about its center of gravity in the X-Y plane. It is easy to see that the rightmost point of object 'B' will immediately fall outside the minimum bounding rectangle shown for object 'B', thus violating the adequacy criterion that an object will always remain within the boundaries of the shape abstraction chosen for it.

The alternative is to adjust the shape of the bounding rectangle as the object rotates, to account for changes in the positions of its extremal points. The problem here is that if the shape of the bounding rectangle is allowed to change, we have to consider the possibility of changes in spatial relations with other objects, such as the possibility of a new intersection with another object.

Unfortunately, without more precise metric information about the distances between objects and the sizes of objects, in general, it is not possible to determine whether a new intersection can result if two rectangles rotate. In Figure 2.3, for example, if object 'A' rotates around its center of gravity, then no new intersections are possible. If instead, object 'B' rotates arounds its center of gravity, then it would eventually intersect with object 'A'.

2.4.3 Consequences of Using Two Different Shape Abstractions

The main consequence of our use of the minimum bounding rectangle abstraction for non-rotating objects and the bounding sphere abstraction for objects that can rotate is that once chosen, the shapes and sizes of the bounding abstraction will remain fixed. Further, since the use of the bounding sphere abstraction ensures that rotation cannot change the relative positions between two objects, we avoid a source of qualitative ambiguity, and get the additional guarantee that only translational motion can change the position of the bounding shapes.

The penalty for using the bounding sphere approximation is that it is more *incomplete*. Notice that if a bounding sphere were used for object 'B' in Figure 1.2, it would intersect with the bounding shape for object 'A', and the two objects would be considered to be intersecting in the initial scene. The use of the minimum bounding rectangle abstraction for object 'B', as shown, allows us to more accurately conclude that objects 'A' and 'B' are disconnected in the initial scene.

2.4.4 How the Bounding Abstractions are Obtained

Our spatial representation requires knowledge of the bounding abstractions chosen for each object only to define the relative positions of objects in the initial scene. All subsequent reasoning is performed based on a qualitative description of the relative positions between objects, as described in Chapter 3, and orientation, as described in Chapter 4.

Therefore, our criteria for choosing shape abstractions can be applied manually, and a qualitative description of the initial scene can be provided directly through the forms described in Chapters 3 and 4. Alternatively, a qualitative description can be computed from a quantitative description, such as a camera image or a diagram.

Computing Bounding Shapes for Objects in Diagrams

In our work, we input a initial scene through a diagram, and automatically compute a qualitative description of the initial spatial state. If a problem is solved through numerical simulation, the results will apply only to the input initial state. By converting the quantitative description of the initial spatial state given in the diagram to a qualitative description, we obtain the additional benefit that the results of a qualitative simulation based on this qualitative description will apply to any physical initial state whose qualitative description is the same.

Based on the numerical descriptions available for each object, we determine whether each extremal surface is formed by a point (vertex or point on a curve) or a line segment (edge). We then record the unique X-coordinate values, `rm-xval` and `lm-xval`, of the rightmost and leftmost extremal points/edges, and the unique Y-coordinate values, `tm-yval` and `bm-yval`, of the topmost and bottommost extremal

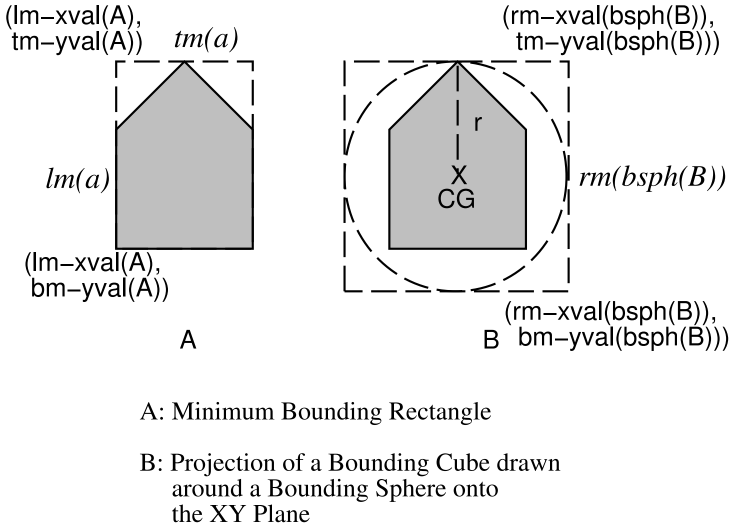


Figure 2.4: Expressing the spatial extent of bounding abstractions.

points/edges. If a minimum bounding rectangle is to be used as the shape abstraction for the object, we use these coordinates to define the boundaries of the bounding rectangle. The rectangle will be bounded by the points *((lm-xval, tm-yval), (rm-xval, tm-yval), (lm-xval, bm-yval), and (rm-xval, bm-yval))*, as illustrated for case A in Figure 2.4.

If a bounding sphere abstraction is to be used, we first numerically compute the coordinates of the center of gravity of the object, then determine the point on the surface of the object that is furthest from the center of gravity, and use the distance between the two points as the radius (labeled `r` in Figure 2.4) of the bounding sphere. Note that with this approach, the center of gravity need not be the center of the object, as illustrated through case B of Figure 2.4.

The coordinates of the extremal points of the bounding sphere are obtained by adding or subtracting the length of the radius, as appropriate, to the values of the X, Y, and Z coordinates of the center of gravity of the object. The X-coordinate values of the rightmost and leftmost points of the bounding sphere, and the Y-coordinate values of its topmost and bottommost points, are used to define the boundaries of the projection of a bounding cube drawn around the sphere onto the X-Y plane, which yields a rectangle. As illustrated for case B in Figure 2.4, the boundaries of this rectangle are given by:

```
((lm-xval(bsph(obj)), tm-yval(bsph(obj))),
 (rm-xval(bsph(obj)), tm-yval(bsph(obj))),
 (lm-xval(bsph(obj)), bm-yval(bsph(obj))),
 (rm-xval(bsph(obj)), bm-yval(bsph(obj)))),
```

where `bsph(obj)` refers to be bounding sphere of the object under consideration.

Internal Representation for the Spatial Extent of Bounding Boxes

Regardless of whether a minimum bounding rectangle or bounding sphere abstraction is used for an object, the physical boundaries of the spatial extent of the object are given by the rectangle projected by the bounding abstraction onto the XY plane. Henceforth, we shall refer to this projected rectangle as the **bounding box** for the object. A physical description of the bounding box is stored internally in terms of the following Algernon predicates defined in the spatial KB.

- Rightmost(obj, xval, min-yval, max-yval, scope)
- Leftmost(obj, xval, min-yval, max-yval, scope)
- Topmost(obj, min-xval, max-xval, yval, scope)
- Bottommost(obj, min-xval, max-xval, yval, scope)

The first two predicates capture the unique X coordinate values and the minimum and maximum Y coordinate values of the rightmost and leftmost edges of the bounding box. The latter two predicates capture the unique Y coordinate values and the minimum and maximum X coordinate values of the topmost and bottommost edges of the bounding box. Wherever it is used, the term `scope` or the variable ?*sco* represent the spatial state or states over which the information in a given predicate asserted in the knowledge base is true. In the predicates given above, the scope argument allows us to model the fact that the physical coordinates of the bounding box could change over time if objects move.

Currently, we only make use of the physical coordinate values of the bounding boxes when processing the initial state. As discussed later in Chapter 3, production rules are run against these numerical coordinate values to generate an equivalent qualitative description of the relative positions of objects in the initial spatial state. This qualitative description is then used for all subsequent reasoning about the relative positions of objects.

2.5 Previous Work

As described below, many shape abstraction methods have been proposed for qualitative spatial reasoning. However, to date, no previous work explicitly presents the guarantees given by the chosen abstraction method. In most previous work, only a subset of spatial reasoning issues is considered, and a spatial abstraction is chosen with the sole criterion being that it should have the qualitative properties to meet the present spatial reasoning needs. The limitation of the abstraction method for other spatial reasoning problems is generally not discussed. For example, no other work that make use of shape abstractions explicitly notes they may be inappropriate for problems requiring knowledge of the orientation of surfaces, and provides an additional representation to maintain the orientation of individual surfaces.

2.5.1 Shape Abstractions Used in Previous Work

One of the easiest shape abstraction methods is simply to treat each object as a point. For example, Kuipers [46] represents distinctive places (the only objects of interest) as points and connections between them as lines. While this approach is quite appropriate for navigation and route finding, queries about the areas of overlap between objects, or the intrinsic fronts of objects, are inexpressible.

Rectangular bounding boxes and bounding cubes are a commonly used shape abstraction method for multi-dimensional objects [1, 8, 37, 53, 57, 77]. Other shape abstractions which have been used include convex hulls [11], and symbolic arrays [30]. For our purposes, convex hulls are inappropriate since they require quantitative knowledge of the shapes of objects to describe and maintain. Glasgow's [30] symbolic arrays are useful for capturing the relative sizes and relative positions of objects, but there is no means provided to describe properties such as areas of overlap or the orientation of individual surfaces of objects.

In the models that use them, the criteria for drawing rectangular shape abstractions vary greatly. Chang [8], Jungert [37], and Maass [53] draw a minimum bounding rectangle or cube around each object based on an extrinsic Cartesian coordinate system. Chang and Jungert exploit the same properties of minimum bounding rectangles that we do. However, our spatial representation is broader as we consider problems involving rotation and problems requiring knowledge of the orientation of individual surfaces of objects.

Abella and Kender [1] draw a bounding box based on a mathematical abstraction, inertia tensors, as illustrated in Figure 2.2. This method is inappropriate for our use since the definitions used are quantitative, the abstractions don't meet our adequacy criterion, and only static scenes are considered.

CITYTOUR, as described in [77], uses rectangular bounding boxes aligned with the intrinsic fronts of objects, as illustrated in Figure 2.5. The goal of CITYTOUR, along with other research that is part of the VITRA (VIsual TRAnslator) project, [3, 29, 53], is to develop methods for *qualitatively reasoning about visual descriptions* of a spatial scene. Retz-Schmidt does not provide any qualitative model for the bounding abstractions that are used. Instead, with the availability of precise, metric information, any bounding shapes and any spatial relationships with other objects are simply computed numerically on an as-needed basis.

Mukerjee and Joe [57, 58] also draw bounding rectangles around the intrinsic

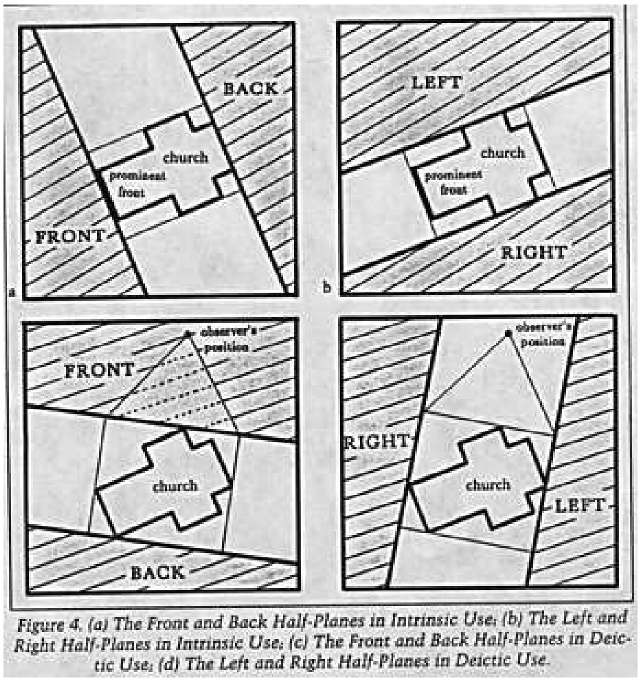


Figure 2.5: The shape abstractions used by CITYTOUR [77].

fronts of objects. They cite that for problems involving reasoning with the intrinsic frames of reference of objects, minimum bounding rectangles drawn around the intrinsic front of an object may provide a 'tighter' abstract shape than a bounding rectangle based on an arbitrary extrinsic coordinate system.

Mukerjee and Joe's [58] spatial representation is based on the concept of 'collision parallelograms', as illustrated in Figure 2.6. In their representation, projections are drawn based on the front direction of each bounding rectangle. Whenever two sets of projections meet, a collision parallelogram is formed. Mukerjee and Joe discuss how reasoning about the relative positions and orientations of objects can be done with respect to the collision parallelograms, and how the qualitative properties of the parallelograms could change if objects translate or rotate.

Mukerjee and Joe's representation assigns a unique frame of reference to each object, and calls for maintaining the relative positions of all objects in terms of all of these frames of reference. This appears to be complex and difficult to maintain,

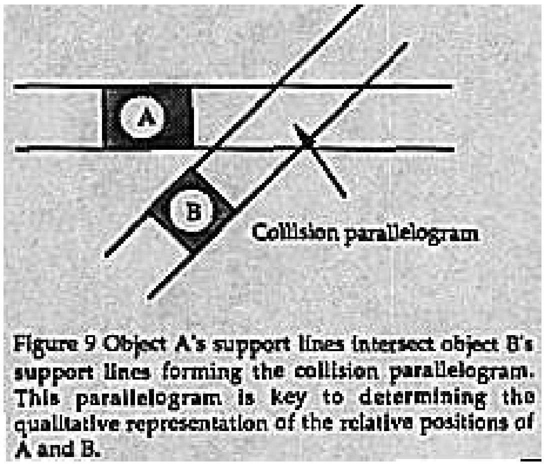


Figure 2.6: Collision Parallelograms, as used by Mukerjee and Joe [57].

particularly if objects are allowed to freely translate and rotate. Unfortunately, Mukerjee and Joe [57, 58] do not provide an implementation of their model. While we agree that it should be possible to extract a qualitative description of the form used by Mukerjee and Joe from a quantitative description such as a diagram, and also agree that this description could be used successfully to reason about a static scene, it is not at all clear how the positions and orientations of the collision parallelograms can be tracked *with only qualitative information* if objects are free to rotate, or even translate, along any axis.

We shall show in Chapter 3, that, instead, when the rectangular bounding boxes of all objects are drawn aligned with an extrinsic coordinate system as is the case in our representation, it is straightforward to *qualitatively* describe the relative positions of objects with respect to this extrinsic coordinate system, and more importantly, to *qualitatively* track changes in the relative positions of objects as they translate and rotate.

2.6 Chapter Summary

This chapter has presented our methods for modeling the spatial extent of objects. Since exact shapes are often difficult to describe qualitatively, we use simplifying abstractions to approximate the shapes of objects. We employ the adequacy criterion that a shape abstraction must fully cover an object at all times. This provides the guarantee that if the bounding abstractions of two objects do not intersect, then the enclosed objects will not be intersecting. This is useful in our work with the magnetic fields domain since we need to reason about all spatial states where it is at all possible for a conducting object to be partially or fully enclosed in a magnetic field.

To meet the adequacy criterion, for objects that cannot rotate, we use a minimum bounding rectangle, drawn around the extremal points of the object, as the abstract shape. For objects that can rotate (about an axis passing through its center of gravity), we use a minimum bounding sphere drawn to cover all points that can be touched by the object as it rotates. These shape abstractions not only satisfy the adequacy criterion, they provide the additional property that only translational motion can change the position an object (as approximated by its bounding shape), and only rotation can change the orientation of an object.

Chapter 3

Modeling Position and Translational Motion

In Chapter 2, we presented our methods for choosing shape abstractions for objects. In this chapter, we present techniques for modeling the absolute positions of objects, and topological and directional spatial relations between objects. After presenting these models, we will illustrate their use in reasoning about the effects of translational motion, including detecting changes in the relative positions of objects and detecting changes in the area of overlap between intersecting objects.

3.1 Modeling the Absolute Position of Objects

In our spatial representation, knowledge of the absolute position of objects serves only one purpose: to detect, for objects in translational motion, when an object has reached a world boundary. Otherwise, all spatial reasoning is done in terms of the relative positions between objects and the orientation of individual objects.

The absolute position of an object is represented abstractly through the position of the rectangular bounding box projected onto the X-Y plane by the minimum bounding rectangle or bounding sphere shape abstraction chosen for the object. In our spatial representation, we choose a Cartesian coordinate system for a problem such that all objects initially lie inside a finite portion of the positive quadrant of the X-Y plane. The bounds of the world are defined by $X \geq 0 \wedge X \leq Maxval$ on the X axis, and $Y \geq 0 \wedge Y \leq Maxval$ on the Y-axis.

The position of any point with coordinates (X,Y) in the X-Y plane is given by its qualitative value ([X], [Y]), where $[X] \in \{+, 0, -\}$ and $[Y] \in \{+, 0, -\}$. Here, a qualitative value of $+$ denotes a positive coordinate value, 0 denotes a zero coordinate value, and $-$ denotes a negative coordinate value. With this representation, for problems involving reasoning about a static world, a model of the absolute position of objects is not required, since all points on all objects will have the same qualitative representation, $(+, +)$, in the initial spatial state.

If a problem involves reasoning about translational motion, then we have to consider the possibility that an object may cross a world boundary. As a rectangular bounding box that is aligned with the coordinate system moves about in the X-Y plane, the points that can first reach a world boundary are those on the rightmost edge of the bounding box if motion is taking place in the positive X direction, the leftmost edge if motion is in the negative X direction, the topmost edge if motion is in the positive Y direction, and the bottommost edge if motion is in the negative Y direction. If there is motion in both the X and Y directions, there can be a *race condition* regarding which edge of the bounding box will reach a world boundary first.

Race conditions are handled automatically by the existing qualitative simulation systems, and will be discussed in a later section. For the purpose of modeling the absolute position of an object, we note that only the following coordinate values are needed for the bounding box of an object, O: the X-coordinate values, $rm(O)$ and $lm(O)$, of the rightmost and leftmost edges, and the Y-coordinate values, $tm(O)$ and $bm(O)$, of the topmost and bottommost edges of the bounding box.

The qualitative value of a variable such as $rm(O)$ is given by the pair

$\langle qmag, qdir \rangle$ its qualitative magnitude and its qualitative direction of change (i.e., increasing ($\uparrow$), decreasing ($\downarrow$), or steady ($\ominus$)) [43]. The qualitative magnitude of a variable is given by its mapping to a *quantity space*, which is a finite, ordered list of symbols representing the qualitative *landmark values* of interest for that variable. Since we use our model of the absolute position of objects only to detect when an object reaches a world boundary, the only landmarks values of interest in modeling the position of an object are those that represent world boundaries.

Since the boundaries of the world are formed by the lines $X = 0$, $X = Maxval$, $Y = 0$, and $Y = Maxval$, the quantity space for each of $rm(O)$, $lm(O)$, $tm(O)$, and $bm(O)$ becomes $(0, Maxval)$. Then, during qualitative simulation, the fact that an object has reached a world boundary can be determined by detecting that any of its position variables has a qualitative magnitude of 0 or $Maxval$. Further, the fact that the world boundary is about to be crossed can be detected, and the simulation of that behavioral path can be halted, if any position variable has a qualitative value of $(0, \downarrow)$ or $(Maxval, \uparrow)$.

3.2 Modeling the Relative Positions of Objects

In the previous section, we noted that the absolute position of an object could be maintained through knowledge of just the X coordinate values of the rightmost and leftmost edges and the Y coordinate values of the topmost and bottommost edges of the rectangular bounding box which approximates its position in the X-Y plane. The relative positions of two rectangular bounding boxes in the X-Y plane can also be easily modeled and maintained *qualitatively* through inequality relations on these coordinate values.

For the remainder of this chapter, the following conventions will be used:

- Any references to a Cartesian coordinate system, the axes of a Cartesian coordinate system, or any planes formed by the axes of a coordinate system will all refer to the extrinsic Cartesian coordinate system chosen for a problem either manually, or automatically by the Figure Understander.

Specifically, if the initial spatial state is input through a diagram, the reader should remember that the Figure Understander sometimes chooses an alternate Cartesian coordinate system with a different orientation of the X, Y, and Z axes than the axes of the diagram.

- The term `bounding box`, as defined in Section 2.4.4, shall refer to the rectangle projected onto the X-Y plane by the minimum bounding rectangle or bounding sphere shape abstraction chosen for an object.

- Comparisons of the relative position of two objects along the X-axis are based on the X-coordinate values of the rightmost and leftmost edges of their bounding boxes. In all formulae, the terms $rm(O)$ and $lm(O)$ shall respectively refer to the *X-coordinate value* of the rightmost and leftmost edges of the bounding box of an object, O. The implemented code uses the names `rm-xval` and `lm-xval` for these terms.

- Likewise, comparisons of the relative position of two objects along the Y-axis are based on the Y-coordinate values of the topmost and bottommost edges of their bounding boxes. In all formulae, the terms $tm(O)$ and $bm(O)$ shall respectively refer to the *Y-coordinate value* of the topmost and bottommost edges of the bounding box of an object, O. The implemented code uses the names `tm-yval` and `bm-yval` for these terms.

- These terms are all *symbolic* references to the extent of a bounding box. Numerical coordinate values are not used in our spatial representation.

- The spatial representation described in this chapter can be extended to model the relative positions of objects in 3-dimensional space by first approximating the spatial extent and position of each object through a cube, and then including knowledge of the Z-coordinate values of the frontmost and rearmost edges of this cube in the model.

3.2.1 Modeling Inequality Relations on Extremal Edges

The inequality relations between the rightmost, leftmost, topmost, and bottommost edges of the bounding boxes of objects are maintained qualitatively in the form $(\langle qrel\rangle\langle BB\text{-}extremal\text{-}edge1\rangle\langle BB\text{-}extremal\text{-}edge2\rangle)$, where $\langle qrel\rangle$ can be any of the qualitative inequality relations recognized by QPC (`q<, q=, q>, q<=, q>=`). For example, the assertion $(q{<}\ rm(A)lm(B))$ captures the relationship that the X-coordinate value of the rightmost edge of the bounding box of object A is less than that of the leftmost edge of the bounding box of object B.

Thus, the spatial state of world is described through three components: (1) the absolute position of each object, (2) the relative position of each object with respect to every other object, and (3) the orientation of each object. The topic of orientation will be addressed in Chapter 4.

The absolute position of each object, O, is described through the qualitative value of $rm(O)$, $lm(O)$, $tm(O)$, and $bm(O)$. The relative position of an object with respect to every other object is maintained through 8 inequality constraints for each pair of objects, A and B, as shown below.

- $(\langle qrel\rangle\ lm(A)\ rm(B))$
- $(\langle qrel\rangle\ lm(A)\ lm(B))$
- $(\langle qrel\rangle\ rm(A)\ rm(B))$
- $(\langle qrel\rangle\ rm(A)\ lm(B))$
- $(\langle qrel\rangle\ tm(A)\ tm(B))$
- $(\langle qrel\rangle\ tm(A)\ bm(B))$

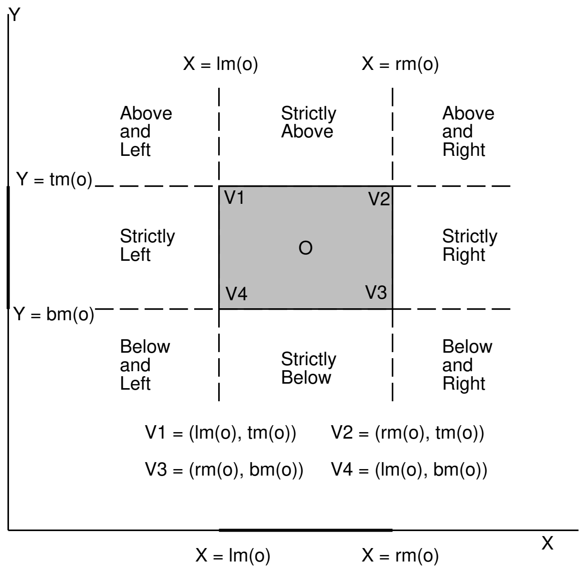


Figure 3.1: The regions that are to the left, right, above, and below an object.

- $(\langle qrel \rangle\ bm(A)\ tm(B))$
- $(\langle qrel \rangle\ bm(A)\ bm(B))$

3.3 Modeling Directional Spatial Relations

Directional spatial relations, such as `right-of` and `left-of`, require a frame of reference. In our work, all directional relations are defined with respect to a Cartesian coordinate system. Then, as pointed out by Chang [8] and Jungert [37], directional relations can be modeled in terms of the projected length of an object onto each of the X and Y axes of the Cartesian coordinate system.

The projected line of an object, O, onto the X-axis has endpoints given by *((lm(O), 0), (rm(O), 0))*, where $lm(O)$ and $rm(O)$ are the X-coordinates of the leftmost and rightmost points or edges of the object. Similarly, the projected line of an object, O, onto the Y-axis has endpoints given by *((0, bm(O)), (0, tm(O)))*, where $bm(O)$ and $tm(O)$ are the Y-coordinates of the topmost and bottommost points or edges of the object. Then, as shown in Figure 3.1, if the world boundaries are defined by $(X \geq 0 \wedge X \leq Maxval)$ and $(Y \geq 0 \wedge Y \leq Maxval)$:

- The region $(X \geq 0 \wedge X < lm(O))$ contains all points to the *left* of O.
- The region $(X > rm(O) \wedge X \leq Maxval)$ contains all points to the *right* of O.
- The region $(Y \geq 0 \wedge Y < bm(O))$ contains all points *below* O.
- The region $(Y > tm(O) \wedge Y \leq Maxval)$ contains all points *above* O.
- The region $(X > lm(O) \wedge X < rm(O))$ contains all points *Aligned-Y* with O. That is, for any object whose spatial extent intersects this region, there exists some motion parallel to the Y-axis that will cause the object to intersect O.
- The region $(Y > bm(O) \wedge Y < tm(O))$ contains all points *Aligned-X* with O. That is, for any object whose spatial extent intersects this region, there exists some motion parallel to the X-axis that will cause the object to intersect O.
- The region to the left of O and aligned-X with O is *strictly-left* of O.
- The region to the right of O and aligned-X with O is *strictly-left* of O.
- The region that is above O and aligned-Y with O is *strictly-above* O.
- The region that is below O and aligned-Y with O is *strictly-below* O.

3.3.1 Modeling Directional Spatial Relations Between Two Objects

In our spatial representation, we use a rectangular bounding box to approximate the spatial extents of all objects. The definitions of the directional spatial relations given above can be used to compare the relative positions of two bounding boxes, A and B, as follows. Similar to Figure 3.1, assume the vertices of bounding box A are given by *((lm(A), tm(A)), (rm(A), tm(A)), (rm(A), bm(A)), (lm(A), bm(A)))* and that the vertices of the bounding box B are given by *((lm(B), tm(B)), (rm(B), tm(B)), (rm(B), bm(B)), (lm(B), bm(B)))*. Then:

- Left-of(A,B) iff $rm(A) < lm(B)$.
- Right-of(A,B) iff $lm(A) > rm(B)$.
- Above(A,B) iff $bm(A) > tm(B)$.
- Below(A,B) iff $tm(A) < bm(B)$.
- Aligned-X(A,B) iff $tm(A) > bm(B) \wedge bm(A) < tm(B)$.
- Aligned-Y(A,B) iff $lm(A) < rm(B) \wedge rm(A) > lm(B)$.
- Strictly-left-of(A,B) iff $\textit{Left-of}(A, B) \wedge \textit{Aligned-X}(A, B)$.
- Strictly-right-of(A,B) iff $\textit{Right-of}(A, B) \wedge \textit{Aligned-X}(A, B)$.
- Strictly-above(A,B) iff $\textit{Above}(A, B) \wedge \textit{Aligned-Y}(A, B)$.
- Strictly-below(A,B) iff $\textit{Below}(A, B) \wedge \textit{Aligned-Y}(A, B)$.

3.4 Modeling Topological Spatial Relations

Topological relations can compare the positions of objects based on several different factors. For example, the relations `near` and `far` compare the *distance* between objects, relations such as `along` compare the *contours* of two paths, and relations such as `overlapped` and `disconnected` describe the property of *containment.*

Of these, the distance and contour based topological relations are the most difficult for qualitative modeling since the applicability of these relations to a pair of objects in a given scene can be highly context dependent. For example, what is 'near' in one context may be considered as 'far' in another. Much of the existing work in modeling these topological relations has focused on the development of *quantitative* techniques for measuring the applicability of these relations in a given context [1, 29, 49, 65].

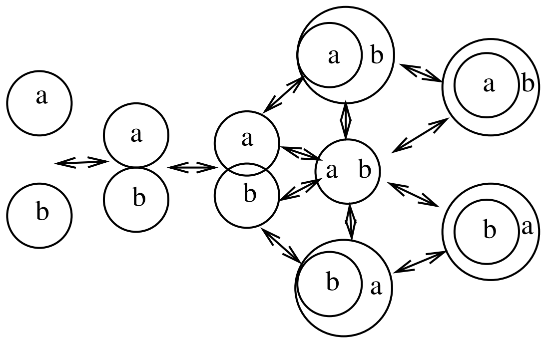


Figure 3.2: The topological relations considered by Cui, Cohn, and Randell [11].

The topological relations of primary interest in our work are those based on *containment*, since a magnetic field can only affect objects that are *in* the field. Particularly when the spatial extents of objects are described through rectangular bounding boxes, it is straightforward to qualitatively model containment-based topological relations.

3.4.1 Previous Work in Modeling Containment Relations

Cui, Cohn, and Randell [11] and Egenhofer and Al-Taha [13] both present models of topological spatial relations related to the property of containment. In their work, each topological relation is modeled as a state, and much of the discussion focuses on what transitions are possible between states. The illustrations in Figures 3.2 and 3.3 show the topological states considered by each of the researchers, and the possible transitions between these states. For example, from the state of being disconnected, the next topological state that two object can enter is one where the objects are touching each other.

The binary topological relations considered in the work of Cui, et. al. [11] and Egenhofer, et. al. [13] are the states *disconnected/disjoint, touching/meet, partial overlap, covered-by, equals, covers, inside, and contains.* Here, `covered-by` and `covers` refer to states where one object is inside the other, but where there is still contact between the boundaries of the objects. `Inside` and `contains` refer to

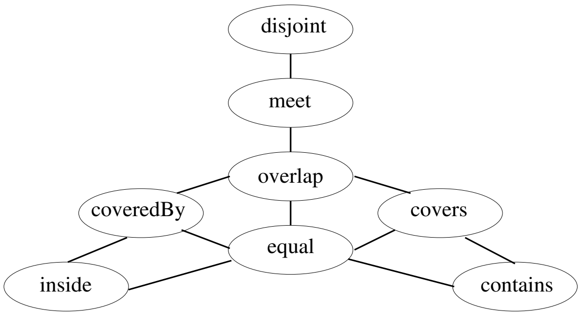


Figure 3.3: The topological relations considered by Egenhofer and Al-Taha [13].

states where there is containment of one object inside the other, but where there is no contact between their boundaries.

Egenhofer and Al-Taha [13] describe the possible changes in topological relations between objects due to changes in the sizes and shapes of objects. However, they do not provide any qualitative model for the positions of objects or for qualitatively detecting when state changes occur.

Similar to our interest, Cui, Cohn, and Randell [11] are interested in describing the possible changes in containment-based topological relations between objects due to the translational motion of objects. However, they too do not provide a qualitative method for detecting when state changes occur - they simply assume that a 'convex-hull' function will be available which determines when object boundaries meet.

Further, Cui, Cohn, and Randell discuss what topological state changes can occur as objects translate, but do not explicitly model motion. For example, they cannot determine which specific state change(s) can occur when given a specific direction of motion for an object. Their model also does not permit reasoning about changes in the area of overlap between objects based on the direction of motion of an object. The latter is crucial when reasoning about the effects of magnetic fields on conducting objects placed in the field, as in the example of Figure 1.1.

3.4.2 Our Model of Containment-Based Topological Relations

In our work, we consider the possibility of free translational and rotational motion. Thus, we require a 2-dimensional model of topological relations. For example, in addition to the concept of overlap, we also have overlap-top, overlap-bottom, overlap-right, overlap-left, overlap-top-right, overlap-top-left, overlap-bottom-right, and overlap-bottom-left, to identify intersections of the bounding box of one object across a specific edge or edges of the bounding box of another object. This is useful for determining if the area of overlap between two objects will increase, decrease, or will remain constant when given a topological state and the direction of motion of an object.

For example, $Overlap\text{-}left(A, B)$ is true if bounding box A overlaps the leftmost edge of bounding box B, but not its rightmost edge. Then, we can conclude that the motion of A in the positive X direction will tend to increase the area of overlap with B. However, if A overlaps both the rightmost and leftmost edges of B, then any motion of A parallel to the X-axis will not affect the area of overlap between the two *rectangular* bounding boxes until this spatial state changes.

Our model of containment-based topological relations is given below. Assume the vertices of a bounding box, A, are given by *((lm(A), tm(A)), (rm(A), tm(A)), (rm(A), bm(A)), (lm(A), bm(A)))* and that the vertices of a bounding box, B, are given by *((lm(B), tm(B)), (rm(B), tm(B)), (rm(B), bm(B)), (lm(B), bm(B)))*. Then:

- Disjoint(A,B) iff $Left\text{-}of(A, B) \vee Right\text{-}of(A, B) \vee Above(A, B) \vee Below(A, B)$.
- Touching(A,B) iff $Touching\text{-}left(A, B) \vee Touching\text{-}right(A, B) \vee Touching\text{-}top(A, B) \vee Touching\text{-}bottom(A, B)$.
- Touching-top(A,B) (A is touching the top of bounding box B) iff $Aligned\text{-}Y(A, B) \wedge bm(A) = tm(B)$.
- Touching-bottom(A,B) iff $Aligned\text{-}Y(A, B) \wedge tm(A) = bm(B)$.

- Touching-left(A,B) iff $\mathit{Aligned\text{-}X}(A,B) \wedge rm(A) = lm(B)$.
- Touching-right(A,B) iff $\mathit{Aligned\text{-}X}(A,B) \wedge lm(A) = rm(B)$.
- Overlapped(A,B) iff $\mathit{Aligned\text{-}X}(A,B) \wedge \mathit{Aligned\text{-}Y}(A,B)$.
- Partial-Overlap(A,B) iff

  $\mathit{Overlapped}(A,B))\wedge$

  $((tm(A) > tm(B) \wedge bm(A) > bm(B))\vee$

  $(bm(A) < bm(B) \wedge tm(A) < tm(B))\vee$

  $(lm(A) < lm(B) \wedge rm(A) < rm(B))\vee$

  $(rm(A) > rm(B) \wedge lm(A) > lm(B)))$.
- Overlap-left(A,B) iff $\mathit{Overlapped}(A,B) \wedge lm(A) < lm(B) \wedge rm(A) < rm(B)$.
- Overlap-right(A,B) iff $\mathit{Overlapped}(A,B) \wedge rm(A) > rm(B) \wedge lm(A) > lm(B)$.
- Overlap-top(A,B) iff $\mathit{Overlapped}(A,B) \wedge tm(A) > tm(B) \wedge bm(A) > bm(B)$.
- Overlap-bottom(A,B) iff $\mathit{Overlapped}(A,B) \wedge bm(A) < bm(B) \wedge tm(A) < tm(B)$.
- Overlap-top-right(A,B) iff $\mathit{Overlap\text{-}top}(A,B) \wedge \mathit{Overlap\text{-}right}(A,B)$.
- Overlap-top-left(A,B) iff $\mathit{Overlap\text{-}top}(A,B) \wedge \mathit{Overlap\text{-}left}(A,B)$.
- Overlap-bottom-right(A,B) iff $\mathit{Overlap\text{-}bottom}(A,B) \wedge \mathit{Overlap\text{-}right}(A,B)$.
- Overlap-bottom-left(A,B) iff $\mathit{Overlap\text{-}bottom}(A,B) \wedge \mathit{Overlap\text{-}left}(A,B)$.
- CoveredBy(A,B) iff

  $\mathit{Overlapped}(A,B)\wedge$

  $((tm(A) = tm(B) \wedge bm(A) > bm(B) \wedge lm(A) \geq lm(B) \wedge rm(A) \leq rm(B))\vee$

  $(bm(A) = bm(B) \wedge tm(A) < tm(B) \wedge lm(A) \geq lm(B) \wedge rm(A) \leq rm(B))\vee$

$(lm(A) = lm(B) \wedge rm(A) < rm(B) \wedge tm(A) \leq tm(B) \wedge bm(A) \geq bm(B)) \vee$

$(rm(A) = rm(B) \wedge lm(A) > lm(B) \wedge tm(A) \leq tm(B) \wedge bm(A) \geq bm(B)))$.

- Covers(A,B) iff $CoveredBy(B, A)$.
- Equal(A,B) iff $tm(A) = tm(B) \wedge bm(A) = bm(B) \wedge lm(A) = lm(B) \wedge rm(A) = rm(B)$.
- Inside(A,B) iff $tm(A) < tm(B) \wedge bm(A) > bm(B) \wedge lm(A) > lm(B) \wedge rm(A) < rm(B)$.
- Contains(A,B) iff $Inside(B, A)$.

3.4.3 Limitations of Our Model of Topological Relations

In our work, we only consider containment-based topological spatial relations, and do not explicitly model distances between objects. An inequality relation of the form *(q< rm(obj1) lm(obj2))* allows the conclusion that $obj1$ is to the left of $obj2$, but does not provide any information about how far apart they are.

Further, we cannot model the possibility of contact between specific surfaces of objects, or the extent of such contact. This would be required, for example, in reasoning about kinematics problems, as studied by Nielsen [61, 62]. Unfortunately, when shape abstractions are used as the basis for describing the relative positions of objects, the possibility of contact between specific surfaces of objects becomes inexpressible.

3.5 Modeling Translational Motion

Translational motion has three interesting effects that we need to model: (1) it changes the absolute position of an object. As mentioned in Section 3.1, it is necessary to be able to detect when an object has reached a world boundary to terminate the simulation of that behavioral path; (2) it can change the relative positions between objects; (3) when objects are overlapped, translational motion can change the area of overlap between objects. This is an important property in the magnetic fields domain since changes in the area of overlap between a magnetic field and a closed conducting object can change the magnetic flux passing through the object.

In our spatial model, the absolute position of an object, O, is maintained through the variables $rm(O)$, $lm(O)$, $tm(O)$, and $bm(O)$, which respectively model the X coordinate values of the rightmost and leftmost edges and the Y-coordinate values of the topmost and bottommost edges of its bounding box. The effect of translational motion of an object on its position variables can be easily described through a derivative relationship with the velocity of the object. More specifically, for an object O, ($d/dt\ rm(O) = X\text{-}velocity(O)$) and ($d/dt\ lm(O) = X\text{-}velocity(O)$), where *X-velocity(O)* is the X-component of the velocity vector of O. Further, ($d/dt\ tm(O) = Y\text{-}velocity(O)$) and ($d/dt\ bm(O) = Y\text{-}velocity(O)$), where *Y-velocity(O)* is the Y-component of the velocity vector of O.

Then, for example, if *X-velocity(O)* has a positive value, the values of $rm(O)$ and $lm(O)$ will both be increasing, and it can easily be detected that a world boundary has been crossed once the qualitative value of $rm(O)$ reaches $(Maxval, \uparrow)$.

3.5.1 Detecting Changes in the Relative Positions of Objects

Of more interest is the case when translational motion causes a change in the relative positions of objects. As explained in Section 3.2.1, in addition to modeling the absolute position of objects, a second component of our model of the spatial state of the world is to maintain 8 inequality constraints between the extremal edges of each pair of bounding boxes for the objects in the scene.

Since directional and topological spatial relations are modeled in terms of inequality constraints on the extremal edges of the bounding boxes of objects, it is straightforward to detect when a constraint that is currently satisfied is violated, and when a constraint that is currently violated becomes satisfied.

For example, the inequality constraints that define the spatial relation *Left-of(A,B)* and *Aligned-Y(A,B)* for two objects A and B are $rm(A) < lm(B)$ and $(rm(A) > lm(B) \wedge lm(A) < rm(B))$. If *Left-of(A,B)* is satisfied in a given spatial state, and if the qualitative value of $rm(A)$ is $((0, Maxval), \uparrow)$, and the qualitative value of $lm(B)$ is $((0, Maxval), \ominus)$, then because $rm(A)$ is increasing, it is clear that the constraint for *Left-of(A,B)* will be violated once the magnitude of $rm(A)$ equals the magnitude of $lm(B)$. Further, if the value of $rm(A)$ continues to increase, then the constraint for *Aligned-Y(A,B)* will be satisfied, since the magnitude of $rm(A)$ will exceed that of $lm(B)$.

3.5.2 Modeling Changes in Area of Overlap

A magnetic field only affects objects that are within the field. As in the example of Figure 1.1, one such effect is the induction of current in a conducting object through which the magnetic flux is changing. In our examples, we consider three sources of change in the magnetic flux passing through an object - (1) due to translational motion of an object through a steady, uniform magnetic field; (2) due to rotational motion on an object within a steady, uniform magnetic field; and (3) due to the placement of the object within a time-varying magnetic field.

For this discussion, only the first two cases are of interest. Case 1 involves a change in the *area of overlap* between a conducting object and a magnetic field. Case 2 involves a change in the *projected area of overlap* of a conducting object onto a magnetic field. We use the term, *coplanar-area-of-overlap*, to model the area of overlap between the bounding boxes of the two objects. The term *coplanar* is used since both bounding boxes are in X-Y plane. We use the term *projected-area-of-overlap* to model the projected area of an object onto another object in the X-Y plane.

In Chapter 4, we will discuss our methods for modeling the projected area of overlap of an object onto the X-Y plane. For now, let us consider how changes in the coplanar area of overlap of two rectangular bounding boxes, A and B, can be determined.

First, we note that if A overlaps both the left and right boundaries of B, then, any motion of A in the X direction *will not* change the area of overlap between A and B until there is a change in the spatial state. Similarly, if A overlaps both the top and bottom boundaries of B, then motion in the Y direction will not change the area of overlap of A and B until there is a change in the spatial state. Thus, we only need to consider the motion of A in the X direction in the cases where is there is overlap of A across the left or right boundary of B, but not both, and the motion of A in the Y direction when there is overlap of A across the top or bottom boundaries of B, but not both.

The relation $Overlap\text{-}left(A, B)$ covers the case where there is overlap of A across the left boundary of B, but not the right boundary. The relation $Overlap\text{-}right(A, B)$ covers the reverse case. Similarly, the relation $Overlap\text{-}top(A, B)$ covers the case where there is overlap of A across the top boundary of B, but not the bottom boundary, and the relation $Overlap\text{-}bottom(A, B)$ covers the reverse case.

If $Overlap\text{-}left(A, B)$ is true then the translational motion of A in the positive X direction would tend to increase the coplanar area of overlap of A and B, and motion in the negative X direction would tend to decrease the area of overlap. The inverse relationship exists between the direction of motion of A and the area of overlap between A and B when $Overlap\text{-}right(A, B)$ is true.

Similarly, if $Overlap\text{-}bottom(A, B)$ is true then the translational motion of A in the positive Y direction would tend to increase the coplanar area of overlap of A and B, and motion in the negative Y direction would tend to decrease the area of overlap. The inverse relationship exists between the direction of motion of A and the area of overlap between A and B when $Overlap\text{-}top(A, B)$ is true.

When one of $Overlap\text{-}top(A, B)$ or $Overlap\text{-}bottom(A, B)$ is true and one

of $Overlap\text{-}left(A, B)$ or $Overlap\text{-}right(A, B)$ is true, then translational motion in either the X or Y direction can change the area of overlap of A and B. These cases are covered by the relations $Overlap\text{-}top\text{-}right(A, B)$, $Overlap\text{-}top\text{-}left(A, B)$, $Overlap\text{-}bottom\text{-}right(A, B)$, and $Overlap\text{-}bottom\text{-}left(A, B)$. In these case, we independently model the extent of overlap of A across each boundary of B, and compute the coplanar area of overlap of A and B by multiplying the length of overlap of A across each boundary of B.
More formally, let:

- Overlap-left-length(A,B) $= rm(A) - lm(B)$ if $Overlap\text{-}left(A, B)$, else 0.
- Overlap-right-length(A,B) $= rm(B) - lm(A)$ if $Overlap\text{-}right(A, B)$, else 0.
- Overlap-top-length(A,B) $= tm(B) - bm(A)$ if $Overlap\text{-}top(A, B)$, else 0.
- Overlap-bottom-length(A,B) $= tm(A) - bm(B)$ if $Overlap\text{-}bottom(A, B)$, else 0.

Then, of the relations $Overlap\text{-}left(A, B)$, $Overlap\text{-}right(A, B)$, $Overlap\text{-}top(A, B)$, $Overlap\text{-}bottom(A, B)$, $Overlap\text{-}top\text{-}right(A, B)$, $Overlap\text{-}top\text{-}left(A, B)$, $Overlap\text{-}bottom\text{-}right(A, B)$, and $Overlap\text{-}bottom\text{-}left(A, B)$,

- If only $Overlap\text{-}left(A, B)$ is true, then:

  $Coplanar\text{-}area\text{-}of\text{-}overlap(A, B) \propto_Q Overlap\text{-}left\text{-}length(A, B)$, where $\propto_Q$ means 'qualitatively proportional to'.

- If only $Overlap\text{-}right(A, B)$ is true, then:

  $Coplanar\text{-}area\text{-}of\text{-}overlap(A, B) \propto_Q Overlap\text{-}right\text{-}length(A, B)$.

- If only $Overlap\text{-}top(A, B)$ is true, then:

  $Coplanar\text{-}area\text{-}of\text{-}overlap(A, B) \propto_Q Overlap\text{-}top\text{-}length(A, B)$.

- If only $Overlap\text{-}bottom(A, B)$ is true, then:

  $Coplanar\text{-}area\text{-}of\text{-}overlap(A, B) \propto_Q Overlap\text{-}bottom\text{-}length(A, B)$.

- If only $Overlap\text{-}top\text{-}left(A, B)$ is true, then:

  $Coplanar\text{-}area\text{-}of\text{-}overlap(A, B) = overlap\text{-}left\text{-}length(A, B) \times overlap\text{-}top\text{-}length(A, B)$.

- If only $Overlap\text{-}top\text{-}right(A, B)$ is true, then:

  $Coplanar\text{-}area\text{-}of\text{-}overlap(A, B) = overlap\text{-}right\text{-}length(A, B) \times overlap\text{-}top\text{-}length(A, B)$.

- If only $Overlap\text{-}bottom\text{-}left(A, B)$ is true, then:

  $Coplanar\text{-}area\text{-}of\text{-}overlap(A, B) = overlap\text{-}left\text{-}length(A, B) \times overlap\text{-}bottom\text{-}length(A, B)$.

- If only $Overlap\text{-}bottom\text{-}right(A, B)$ is true, then:

  $Coplanar\text{-}area\text{-}of\text{-}overlap(A, B) = overlap\text{-}right\text{-}length(A, B) \times overlap\text{-}bottom\text{-}length(A, B)$.

- By definition, $Overlap\text{-}left(A, B)$ and $Overlap\text{-}right(A, B)$ cannot be true at the same time. If A overlaps both the right and left boundaries of B, or if A overlaps neither boundary of B, both relations will not be true, and motion parallel to the X-axis cannot change the area of overlap of A and B.

- Similarly, $Overlap\text{-}top(A, B)$ and $Overlap\text{-}bottom(A, B)$ cannot be true at the same time. If A overlaps both the top and bottom boundaries of B, or if A overlaps neither boundary of B, both relations will not be true, and motion parallel to the Y-axis cannot change the area of overlap of A and B.

3.5.3 Race Conditions

During qualitative simulation, since quantitative values are not maintained for the distances between objects or the velocities of objects, it is possible for ambiguities to arise regarding which of a number of inequality constraints changes first. For example, the qualitative value of $rm(A)$ for some object A may be approaching $(Maxval, \uparrow)$ while at the same time, an object B could be moving with respect to object C such that the constraint $(rm(B) > lm(C))$ will be violated.

In many cases, there may not be a way to precisely determine which of these qualitatively significant changes will happen first. In such situations, qualitative simulations systems such as QPC [9, 17] and QSIM [42, 43] automatically consider all combinations and simulate the effects of each as a separate behavior path. In this example, there will be three branches in the simulation, where (1) object A reaches the world boundary first; (2) where the $Left-of(B, C)$ relation is violated first; and (3) where both events happen at the same time.

3.6 Implementation of our Relative Position Model

Our model of the relative positions of objects has been implemented in the form of Algernon [10] rules in the spatial KB, and as views in qualitative process theory [20, 17] for problems to be solved through qualitative simulation. In the sections below, a brief overview of the implementation is provided.

3.6.1 Algernon Rules for Modeling Relative Positions

The Algernon implementation is used by the Figure Understander [73] in processing references to spatial relations in text input. Some example Algernon if-needed (backward-chaining) rules are given in Figures 3.4 and 3.5. These rules partially implement the definitions of spatial relations given in Sections 3.3.1 and 3.4.2, and include definitions for the relations *Aligned-X*, *Left-of*, *Strictly-left-of*, *Touching-left*, *Adjacent-left*, *Overlap-left*, *Overlapped*, and *Equal*.

These rules are all associated with objects of the class `convex-2-d-objects`, or convex, 2-dimensional objects. As discussed in Section 2.4.4, the variable `?sco` represents the scope, or the spatial state or states over which an assertion in the knowledge base holds. The variable `*initial-model-name*` holds the name of the initial spatial state. The term `(:test <X>)` is a call to Lisp to evaluate `X`.

Each `:test` condition searches a list, `*initial-ordinal-relations*`, to see if an inequality relation of interest in satisfied in the initial spatial state. This list is compiled automatically by the Figure Understander as it processes a diagram, and will be discussed in greater detail in Chapter 6.

Incidentally, the relation *Adjacent-left* is applied only to a static scene, where the shape abstraction for all objects will be a minimum bounding rectangle. This relation states that two objects are adjacent at some boundary if they are touching, and if the touching surfaces are both edges.

3.6.2 Support for Reasoning with Other Frames of Reference

The Algernon implementation of our spatial model provides some limited support for reasoning with frames of reference other than the initial default Cartesian coordinate system chosen for the problem.

Choosing Whether the Y Axis Represents Height or Depth

We input a problem description through a 2-dimensional diagram for the spatial scene and text to describe any domain or problem specific properties of diagram objects that cannot be captured in the diagram itself. We allow the use of spatial relations in the text to identify the diagram objects of interest.

The default that we use is that the principle axes of the diagram, X and Y, represent breadth and height, respectively. Then, 'depth' relations in the text input, such as 'ahead-of' would not have any meaning in terms of the diagram, since

```
(:rules convex-2-d-objects
 ((aligned-x ?obj1 ?obj2 ?sco)
  <-
  (member convex-2-d-objects ?obj2)
  (:test (equal '?sco ',*initial-model-name*))
  (:test (member `(q< (bm-yval ,'?obj1) (tm-yval ,'?obj2))
                 *initial-ordinal-relations* :test 'equal))
  (:test (member `(q< (bm-yval ,'?obj2) (tm-yval ,'?obj1))
                 *initial-ordinal-relations* :test 'equal)))

 ((left-of ?obj1 ?obj2 ?sco)
  <-
  (member convex-2-d-objects ?obj2)
  (:test (equal '?sco ',*initial-model-name*))
  (:test (member `(q< (rm-xval ,'?obj1) (lm-xval ,'?obj2))
                 *initial-ordinal-relations* :test 'equal)))

 ((strictly-left-of ?obj1 ?obj2 ?sco)
  <-
  (left-of ?obj1 ?obj2 ?sco)
  (aligned-x ?obj1 ?obj2 ?sco))

  ; contact between obj1 and obj2 at the left boundary of obj2.
 ((touching-left ?obj1 ?obj2 ?sco)
  <-
  (member convex-2-d-objects ?obj2)
  (:test (equal '?sco ',*initial-model-name*))
  (:unp (overlapped ?obj1 ?obj2 ?sco))
  (:test (member `(q= (rm-xval ,'?obj1) (lm-xval ,'?obj2))
                  *initial-ordinal-relations* :test 'equal)))

 ((adjacent-left ?obj1 ?obj2 ?sco)
  <-
  (member convex-2-d-objects ?obj2)
  (touching-left ?obj1 ?obj2 ?sco)
  (has-rightmost-edge ?obj1 True ?sco)
  (has-leftmost-edge ?obj2 True ?sco)))
```

Figure 3.4: Example of Algernon rules for the spatial relations Aligned-X, Left-of, Strictly-left-of, Touching-left, and Adjacent-left.

```
(:rules convex-2-d-objects
 ((overlapped ?obj1 ?obj2 ?sco)
  <-
  (member convex-2-d-objects ?obj2)
  (aligned-x ?obj1 ?obj2 ?sco)
  (aligned-y ?obj1 ?obj2 ?sco))

 ((overlap-left ?obj1 ?obj2 ?sco)
  <-
  (member convex-2-d-objects ?obj2)
  (overlapped ?obj1 ?obj2 ?sco)
  (:test (equal '?sco ',*initial-model-name*))
  (:test (member `(q< (lm-xval ,'?obj1) (lm-xval ,'?obj2))
                 *initial-ordinal-relations* :test 'equal))
  (:test (member `(q< (lm-xval ,'?obj2) (rm-xval ,'?obj1))
                 *initial-ordinal-relations* :test 'equal)))

 ((equal ?obj1 ?obj2 ?sco)
  <-
  (member convex-2-d-objects ?obj2)
  (:test (equal '?sco ',*initial-model-name*))
  (:test (member `(q= (bm-yval ,'?obj2) (bm-yval ,'?obj1))
                 *initial-ordinal-relations* :test 'equal))
  (:test (member `(q= (tm-yval ,'?obj1) (tm-yval ,'?obj2))
                 *initial-ordinal-relations* :test 'equal))
  (:test (member `(q= (lm-xval ,'?obj2) (lm-xval ,'?obj1))
                 *initial-ordinal-relations* :test 'equal))
  (:test (member `(q= (rm-xval ,'?obj1) (rm-xval ,'?obj2))
                 *initial-ordinal-relations* :test 'equal))))
```

Figure 3.5: Example of Algernon rules for the spatial relations Overlapped, Overlap-left, and Equal.

```
((ahead-of ?obj1 ?obj2 ?sco)
 <-
 (:bind ?sce scenario-name)
 (principle-axes ?sce XZ)
 (above ?obj1 ?obj2 ?sco))
```

Figure 3.6: An Algernon rule which translates queries on a 'depth' relation, ahead-of, to one compatible with our breadth-and-height based spatial model.

all points of all objects would have a zero Z coordinate value. However, in some applications, particularly in natural language understanding, it may be preferable to treat the Y axis as the depth axis instead of the height axis. That is, instead of using terms such as 'above' and 'below' in the text input, it may be preferable to use terms such as 'ahead-of' and 'behind'.

One of the options provided by the Figure Understander, our input processing module, is that a user can request that the principle axes of the diagram should be treated as the breadth and depth axes so that queries on 'depth' relations such as *ahead-of* and *behind* will be allowed. It is assumed that in these cases, 'height' relations such as 'above' will not be used in the text input.

That is, if the assertion `(principle-axes ?sce XZ)` is made into the knowledge base, where `?sce` is a scope variable bound to the entire time of the simulation, it is an indication that 'depth' relations may be found in any text input or query, and that they should be interpreted in terms on the relative positions of objects along the Y-axis of the diagram. Queries on 'depth' relations are then automatically translated into an equivalent query compatible with our height-and-breadth based spatial model. An example Algernon rule, which translates queries on *ahead-of* to queries on *above*, is given in Figure 3.6.

Support for Reasoning with Intrinsic Frames of Reference

As noted by Mukerjee and Joe [57] and Retz-Schmidt [77], in some applications, it may be necessary to reason about directional relations between objects in terms of an intrinsic or deictic frame of reference. To support this ability, a method is required for converting spatial knowledge given in terms of an extrinsic frame of reference to be compatible with the intrinsic or deictic frame of reference of interest.

For reasoning about static scenes, this is accomplished through a query on the `dir-rel-with-FoR` (DIRectional RELation WITH a Frame Of Reference) predicate, `(dir-rel-with-FoR ?dir-rel ?obj1 ?obj2 ?sco ?FoR)`. This predicate accepts the name of a relation to prove, `?dir-rel`, the objects whose positions are being compared, `?obj1` and `?obj2`, a scope argument `?sco`, and an intrinsic or deictic front direction, `?FoR`. The query is resolved by proving an equivalent relation in terms of the extrinsic coordinate system chosen for the problem.

The equivalent relation is determined through a Lisp function. Currently, the Lisp function consults a lookup table, as illustrated by the excerpt in Figure 3.7, to obtain the equivalent relation. The lookup table only considers intrinsic and deictic front directions that are aligned with one of the axes of the Cartesian coordinate system chosen for the problem.

The lookup table treats the given intrinsic front direction as the Y-axis of the intrinsic coordinate system. The intrinsic front direction is given as a vector with direction, (X-comp, Y-comp, Z-comp), in terms of the extrinsic coordinate system chosen for the problem. In the partial lookup table in Figure 3.7, the intrinsic front vector, (1 0 0), specifies that it points toward the positive X-axis of the extrinsic coordinate system. Thus, the positive X-axis of the extrinsic coordinate system will be treated at the positive Y-axis of the intrinsic coordinate system. In the example of Figure 3.7, a query of *right-of* in terms of this intrinsic coordinate system would be treated as query of *below* in terms of the extrinsic coordinate system.

```
(((right-of (0 1 0)) right-of)
 ((right-of (0 -1 0)) left-of)
 ((right-of (1 0 0)) below)
 ((right-of (-1 0 0)) above)))
```

Figure 3.7: An entry from our lookup table which returns an equivalent spatial relation to prove in terms of the extrinsic frame of reference chosen for the problem when given a query to prove in terms of an orthogonal frame of reference.

3.6.3 Modeling Spatial Relations Through QPC Views

For problems involving motion, the QPC [9, 17] implementation of spatial model is used to solve the problem through qualitative simulation.

All parameters of interest in the qualitative simulation are modeled as QPC variables, or `quantity-types`. The declaration, `(quantity-type rm-xval convex-2-d-objects)`, names a QPC variable (e.g., `rm-xval`), and associates the variable with a class of objects (e.g., `convex-2-d-objects`). In our model, the quantity-type `rm-xval` represents the X-coordinate value of the rightmost edge of the bounding box of an object. The other quantity types which describe the absolute position of an object are `lm-xval`, for the X-coordinate value of the leftmost edge of its bounding box, and `tm-yval` and `bm-yval`, for the Y-coordinate value of the topmost and bottommost `bm-yval` edges of the bounding box.

An instantiated quantity type is called a *quantity*. For example, a convex 2-dimensional object, 'A', will have the quantity `rm-xval(A)`. As described previously in Section 3.1, the qualitative value of a quantity is given by the pair <*qmag, qdir*>, where *qmag*, its qualitative magnitude, is given by the mapping to its quantity space, and where *qdir*, its qualitative direction of change, can be increasing ($\uparrow$), decreasing ($\downarrow$), or steady ($\ominus$) [43].

In QPC, the relationships between the quantities of objects are described through model fragments, or *views*, in a model library. Figure 3.8 shows QPC views for the spatial relations Aligned-X, Aligned-Y, Left-of, and Overlapped. Figure

3.9 shows the QPC views for the spatial relations Overlap-left and Overlap-top-left. Before exploring further details about these specific views, let us examine the structure of a QPC view in general.

Overview of QPC Views

QPC views represent *model fragments* that could be active in a given qualitative state. QPC checks the activation conditions of each view against the knowledge base and assembles a unified model from the model fragments that are active during a given qualitative state. QPC then calls the QSIM [42, 43] qualitative simulation engine to determine the qualitative behaviors of this model. QSIM halts the simulation of a behavior path if a quiescent state (no changes) is achieved, or if a *model transition* is achieved. QPC models are assembled in such a way that a model transition takes place whenever the activation condition of a currently active view is violated, or if the activation condition of a currently inactive view is satisfied. When there is a model transition, QPC regains control and assembles a new model for further simulation based on the new state of the world.

A QPC view has three basic components. The `individuals` are the world objects to which the view applies. In Figure 3.8, the views `aligned-X-view` and `left-of-view` apply to any two convex 2-dimensional objects. In describing the individuals of interest, other conditions, called the *structural conditions* for the view [17, 20], may be specified.

For example, the view `bb-overlap-view` also applies to two convex, 2-dimensional objects, but has the additional conditions that one of the objects must be a solid object and one must be a permeable object. Further, because this view considers the possibility of motion, it requires that the solid object is a mobile object. There is another view in our model, `permanent-bb-overlap-view`, to model the overlap of an immobile solid object with a permeable object. Other structural conditions in this view are that the permeable object has to be in the X-Y plane, and that the views, `aligned-X-view` and `aligned-Y-view`, must be active for the pair of objects under consideration.

The second basic component of a QPC view is the specification of the `operating-conditions` for the model. These specify the conditions under which the view is active. If no operating conditions are given, as for the view `bb-overlap-view`, then the view is always active for any individuals that meet the structural conditions.

The operating conditions are specified as inequality relations on quantities. An operating condition can compare the magnitude of a quantity with one of the landmarks in its quantity space, or it can specify that a given inequality relationship has to hold between two quantities.

The third major component of a QPC view description is the `:relations` section. This section specifies the consequences of the view when it is active. These are assertions to be made into the knowledge base whenever the view is active. In Figure 3.8, the views `aligned-X-view` and `aligned-Y-view` make a simple assertion into the knowledge base that the views are active. Note that these assertions are referenced in the structural conditions of the `bb-overlap-view` view.

The QPC Model of Overlap

In Figure 3.8, the definitions of the views `Aligned-X-view`, `Aligned-Y-view`, and `Left-of-view` are all straightforward. The operating conditions of each of these views simply check to see if the given inequality relations hold between the extremal edges of the bounding boxes of two objects. If the inequality relations are satisfied, than so is the spatial relation modeled by the view.

Of more interest are the views which model the spatial relations Overlapped, Overlap-left, and Overlap-top-left, as given in Figures 3.8 and 3.9. For example, the relations section of the view `bb-overlapped-view` describes several properties that are true when two objects are overlapped. First, there is the simple assertion that the coplanar area of overlap of the two objects cannot be less than zero while there

```
(defview aligned-X-view ()
 :individuals ((obj1 :type convex-2-d-objects)
               (obj2 :type convex-2-d-objects))
 :operating-conditions ((q> (tm-yval obj1) (bm-yval obj2))
                        (q< (bm-yval obj1) (tm-yval obj2)))
 :relations ((is-active-two-role-view ?model 'v-aligned-X obj1 obj2)))

(defview aligned-Y-view ()
 :individuals ((obj1 :type convex-2-d-objects)
               (obj2 :type convex-2-d-objects))
 :operating-conditions ((q> (rm-xval obj1) (lm-xval obj2))
                        (q< (lm-xval obj1) (rm-xval obj2)))
 :relations ((is-active-two-role-view ?model 'v-aligned-Y obj1 obj2)))

(defview left-of-view ()
 :individuals ((left-obj :type convex-2-d-objects)
               (right-obj :type convex-2-d-objects))
 :operating-conditions ((q< (rm-xval left-obj) (lm-xval right-obj)))
 :relations ((not (are-overlapped ?model left-obj right-obj))))

(defview bb-overlap-view ()
 :individuals
  ((solid-obj :type convex-2-d-objects
              :conditions ((mobility-of solid-obj mobile)
                           (isa solid-obj solid-objects)))
   (perm-obj :type convex-2-d-objects
             :conditions ((isa perm-obj permeable-objects)
                          (in-XY-plane perm-obj TRUE ?model)
                          (is-active-two-role-view
                           ?model 'v-aligned-X solid-obj perm-obj)
                          (is-active-two-role-view
                           ?model 'v-aligned-Y solid-obj perm-obj))))
 :relations
  ((not (q< (coplanar-area-of-overlap solid-obj perm-obj) ZERO))
   (piecewise-continuous-variable
    ?model (coplanar-area-of-overlap-deriv solid-obj perm-obj))
   (mult (cos-theta-X solid-obj) (cos-theta-Y solid-obj)
         (proj-factor-XY-plane solid-obj))
   (mult (coplanar-area-of-overlap solid-obj perm-obj)
         (proj-factor-XY-plane solid-obj)
         (proj-area-of-overlap solid-obj perm-obj))
   (is-active-two-role-view ?model 'v-bb-overlap solid-obj perm-obj)))
```

Figure 3.8: Examples of QPC views that model the spatial relations Aligned-X, Aligned-Y, Left-of, and Overlapped.

```
(defview overlap-bb-left-view ()
 :individuals
  ((solid-obj :type solid-objects
              :conditions ((isa solid-obj convex-2-d-objects)))
   (perm-obj :type permeable-objects
             :conditions ((isa perm-obj convex-2-d-objects)
                          (mobility-of perm-obj immobile)
                          (is-active-two-role-view
                           ?model 'v-bb-overlap solid-obj perm-obj))))
 :operating-conditions ((q<= (rm-xval solid-obj) (rm-xval perm-obj))
                        (q< (lm-xval solid-obj) (lm-xval perm-obj)))
 :relations
  ((is-active-two-role-view ?model 'v-overlap-bb-left solid-obj perm-obj)
   (I+ (coplanar-area-of-overlap solid-obj perm-obj)(x-velocity solid-obj))
   (q+ (coplanar-area-of-overlap-deriv solid-obj perm-obj)
       (x-velocity solid-obj))
   (q>= (left-overlap-length solid-obj perm-obj) zero)
   (add (lm-xval perm-obj) (left-overlap-length solid-obj perm-obj)
        (rm-xval solid-obj))
   (q+ (coplanar-area-of-overlap solid-obj perm-obj)
       (left-overlap-length solid-obj perm-obj))
   (correspondence (coplanar-area-of-overlap solid-obj perm-obj) zero
                   (left-overlap-length solid-obj perm-obj) zero)))

(defview overlap-bb-top-left-view ()
 :individuals
  ((solid-obj :type solid-objects
              :conditions ((isa solid-obj convex-2-d-objects)))
   (perm-obj :type permeable-objects
             :conditions ((isa perm-obj convex-2-d-objects)
                          (mobility-of perm-obj immobile)
                          (is-active-two-role-view
                           ?model 'v-overlap-bb-top solid-obj perm-obj)
                          (is-active-two-role-view
                           ?model 'v-overlap-bb-left solid-obj perm-obj))))
 :relations ((mult (top-overlap-length solid-obj perm-obj)
                   (left-overlap-length solid-obj perm-obj)
                   (coplanar-area-of-overlap solid-obj perm-obj))))
```

Figure 3.9: Examples of QPC views that model the spatial relations Overlap-left and Overlap-top-left.

is overlap between them. While intuitively, it seems that the area of overlap should be greater than zero, we have to allow for the possibility that the area of overlap could become zero during the qualitative simulation of any active processes.

Second, the relationship between the coplanar area of overlap of the two objects and the projected area of overlap of one of the objects onto the other is given. This will be discussed in greater detail in Chapter 4. Third, as discussed in Section 1.4.2, since the value of the derivative of the coplanar area of overlap can change discontinuously, this variable is declared as *piecewise-continuous* variable. Our treatment of piecewise-continuous variables will be discussed in Chapter 7.

The views which model changes in the coplanar area of overlap of two bounding boxes due to translational motion are the views `overlap-bb-left-view` and `overlap-bb-top-left-view`. These views model the spatial relations $Overlap\text{-}left$ and $Overlap\text{-}top\text{-}left$, respectively.

In the relations section of the view, `overlap-bb-left-view`, the second and third relations relate the change in the coplanar area of overlap of the bounding boxes of two objects, `solid-obj` and `perm-obj`, to the X component of the velocity of `solid-obj`. The fourth and fifth relations of the view note the length of overlap of the bounding box of `solid-obj` across the left edge of the bounding box of `perm-obj`.

The view, `overlap-top-left-view`, models the case where there is partial overlap of a bounding box, A, over the top and left boundaries of a bounding box, B. In this case, motion of `solid-obj` along either the X or Y direction could affect the area of overlap between bounding boxes A and B. Therefore, the coplanar area of overlap of A and B is computed by multiplying the length of overlap of A across each of the left and top boundaries of B.

```
(defview constraints-on-mobile-objects ()
 :individuals ((obj :type mobile-objects))
 :relations ((d/dt (rm-xval obj) (x-velocity obj))
             (d/dt (lm-xval obj) (x-velocity obj))
             (d/dt (tm-yval obj) (y-velocity obj))
             (d/dt (bm-yval obj) (y-velocity obj))))
```

Figure 3.10: Modeling the effects of translational motion on position in QPC.

The QPC Model of Translational Motion

The model of the effects of translational motion on the position of an object is given by the view `constraints-on-mobile-objects` as shown in Figure 3.10. This view simply relates the derivative relationship between the X and Y components of the translational velocity of an object and its position quantities, as was described in Section 3.5.

3.7 Chapter Summary

This chapter has presented our model of the absolute position of an object (as approximated by its bounding shape), and the relative positions between objects. We also discussed our methods for determining the effects of translational motion on each.

In our spatial representation, position is modeled with respect to an extrinsic Cartesian coordinate system. Once a bounding shape abstraction is chosen for an object, we determine the boundaries of the rectangular bounding box that is projected by the bounding abstraction onto the X-Y plane, and use the bounding box to model the position of the object. Directional and topological spatial relations between objects are modeled through inequality relations on the X-coordinate values of the rightmost and leftmost edges of their bounding boxes and the Y-coordinate values of their topmost and bottommost edges. Numerical coordinate values are not required and all knowledge of the relative positions of two objects is maintained through the inequality relations described above. The effect of translational motion on a given spatial scene is determined by studying the changes in the truth values of the inequality constraints which describe the spatial relations that are true in the scene.

Finally, since qualitative simulation systems such as QPC [9] reason primarily with inequality constraints on model variables, we showed that our model of the relative positions of objects and the translational motion of objects could be implemented in a straight-forward fashion within the framework of these existing qualitative simulation systems.

Chapter 4

Modeling Orientation and Rotational Motion

This chapter describes our model of orientation and rotational motion. Unlike our model of relative positions, which is based on comparing the positions of the bounding boxes of objects, our model of orientation considers the orientation of the actual object, and not an approximation of the shape of the object. The restriction we impose is that all objects must be convex.

We will first present our model of the orientation of individual surfaces of objects, and then we will discuss our model of the projected area of an object onto the X-Y plane. We will then discuss our model of rotation, our model of vectors and vector operations, and will conclude the chapter with a discussion of the QPC-based implementation of our model of orientation and rotation. As in Chapter 3, all references to an extrinsic, Cartesian coordinate should be interpreted to mean the final extrinsic coordinate system which has been chosen for the problem.

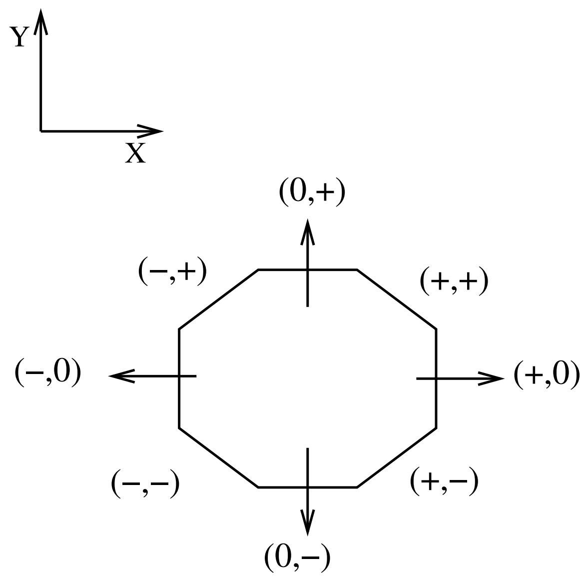


Figure 4.1: The 8 possible qualitative orientations for any surface of an object in the X-Y plane.

4.1 Modeling the Orientation of Surfaces of Objects

As discussed by Freksa [27], Mukerjee and Joe [58], and Nielsen [62], for a 2-dimensional object confined to the X-Y plane, there are 8 qualitative orientations for any surface of an object. These are shown in Figure 4.1. When all objects are convex, knowledge of the identities of the extremal vertices and edges of the object is sufficient to obtain the qualitative orientation of any surface of the object, as described below.

We define the **orientation** of an object as a union of pairs (s_i, d_i), denoting some surface of the object and its surface normal direction. The set of all s_i should exactly cover the surface area of the object. The directions, d_i, are defined by a qualitative vector (Xmag, Ymag) where each of Xmag and Ymag may carry a value from the set $\{-, 0, +\}$ to denote whether the direction vector has a component along each of the X and Y axes, respectively.

Given these definitions, the orientation of any surface of a two-dimensional convex object in the X-Y plane can be determined as follows:

- For a polygon, if there is an extremal edge, then the orientation of that edge will be (1 0) for a rightmost edge, (0 1) for a topmost edge, (-1 0) for a leftmost edge, and (0 -1) for a bottommost edge.
- For any surface which lies between the topmost and rightmost vertices or edges, the orientation vector is (1 1).
- For any surface which lies between the rightmost and bottommost vertices or edges, the orientation vector is (1 -1).
- For any surface which lies between the bottommost and leftmost vertices or edges, the orientation vector is (-1 -1).
- For any surface which lies between the leftmost and topmost vertices or edges, the orientation vector is (-1 1).

4.1.1 Modeling Surface Tangent Directions

Knowledge of the surface normal direction of a surface is sufficient for such applications as modeling the intrinsic or deictic front direction of an object [27, 58]. For modeling such properties as the direction of current flow along a surface of a conducting object, or to model the direction of a frictional force acting on an object [62], we need the surface tangent direction of each surface of interest. For objects in the X-Y plane, the surface tangent direction, in the clockwise direction as viewed from positive infinity on the Z-axis, can be obtained from knowledge of the surface normal direction as described below. The surface tangent direction in the counterclockwise direction may be obtained by reversing the signs of the X and Y components of the clockwise tangent vectors given below.

- If the surface normal direction of a surface is (-1 0), then its surface tangent direction (in the clockwise direction) will be (0 1).

- If the surface normal direction of a surface is (-1 1), then its surface tangent direction will be (1 1).
- If the surface normal direction of a surface is (0 1), then its surface tangent direction will be (1 0).
- If the surface normal direction of a surface is (1 1), then its surface tangent direction will be (1 -1).
- If the surface normal direction of a surface is (1 0), then its surface tangent direction will be (0 -1).
- If the surface normal direction of a surface is (1 -1), then its surface tangent direction will be (-1 -1).
- If the surface normal direction of a surface is (0 -1), then its surface tangent direction will be (-1 0).
- If the surface normal direction of a surface is (-1 -1), then its surface tangent direction will be (-1 1).

4.1.2 Definition of Clockwise and Counterclockwise

As encountered in the previous section, sometimes it is convenient to use the terms *clockwise* and *counterclockwise* in describing such properties as the direction of rotation of an object or the direction of current flow in a closed conducting object. These terms require a perspective, or vantage point. When a vantage point is not explicitly specified, such as 'there is clockwise current flow in the wire loop', we will assume that the vantage point is looking in from infinity on the positive Z-axis.

Following standard conventions [5], for modeling rotation, a counterclockwise rotation will considered positive, and a clockwise rotation will be considered negative, whereas, for modeling the direction of current flow, clockwise will be considered as positive, and counterclockwise will be considered as negative.

4.1.3 Modeling the Connectivity of an Object

To reason about the orientation of a surface of an object, there needs to be a method for modeling the surfaces of objects. The spatial KB contains primitives for storing the connectivity of a convex polygon and for maintaining knowledge of the vertices and edges that form the extremal points of an object. Objects that do not have distinguishable extremal points (i.e., a vertex or edge), such as a circular object, are handled as special cases, and there is no explicit representation of the surfaces of such objects.

The connectivity of a polygon is modeled by a list, $(V_0, E_0, V_1, E_1, ..., V_n, E_n)$, where each V_i is a vertex, and each E_i is the edge that is attached to that vertex in the clockwise direction as viewed from positive infinity on the Z-axis. Predicates of the form `(topmost-edge ?obj ?edge ?sco)` and `(topmost-vertex ?obj ?vert ?sco)` are used to hold the identities of the extremal edges and vertices of an object in the spatial state denoted by the scope variable `?sco`.

4.2 Orientation and Rotation in 3-dimensional space

Based on the definitions given in the previous section, to maintain the orientation of a rotating object, we must determine the changes in the identities of its extremal vertices and edges as it rotates. As an object rotates in 3-dimensional space, the surface normal direction of any given surface could have a non-zero Z component. Currently, we do not explicitly model the Z component of the surface normal vector of an object. Instead, for our purposes, we have found it sufficient to maintain the angle of rotation of each object with respect to each of the X, Y, and Z axes.

4.2.1 Modeling Rotation Around the X or Y axis

For the discussion below, the reader should remember that we only consider the rotation of a 2-dimensional object about an axis that passes through its center of gravity. Therefore, phrases such as 'rotation about the X-axis' should be interpreted as 'rotation about an axis passing through the center of gravity of the object which is parallel to the X-axis of the frame of reference for the problem'.

Although we start with the assumption that all objects are 2-dimensional and are initially in the X-Y plane, domain processes could cause rotation about the X, Y, or Z axis. Let us first consider rotation around the X and Y axis.

These rotations can change the projected area of an object onto another object in the X-Y plane, a property of great interest in the operation of such devices as generators and motors. Further, as an object rotates around the X or Y axis, it is possible for the object to become inverted, as viewed from positive infinity on the Z-axis. That is, during rotation around the X axis, the identities of the topmost and bottommost vertices or edges of the object could become reversed, while during rotation about the Y-axis, the identities of the rightmost and leftmost vertices or edges could be reversed. This needs to explicitly considered, for example, in describing the direction of current flow in a closed conducting loop. What appears to be clockwise current flow during one interval of rotation could suddenly appear to be undefined for an instant of time (when the projection of the conducting loop onto the X-Y plane is a line), and then appear to be counterclockwise in the next rotation interval.

We model the extent of rotation about the X and Y axes, respectively, through two angles measured counterclockwise from the X-Y plane to the plane of the object:

- θ_Y is the angle between the plane of an object and the Y axis.
- θ_X is the angle between the plane of an object and the X axis.

When θ_X or θ_Y is equivalent to 90 or 270 degrees, the projected area of an object onto the X-Y plane is zero. These angles also mark the point at which the orientation of the object changes. During counterclockwise rotation about the X-axis, when

θ_Y is in the interval 90 degrees to 270 degrees, the identities of the topmost and bottommost vertices or edges will be reversed. During rotation about the Y-axis, the identities of the leftmost and rightmost vertices or edges can be reversed. ;In Figure 1.2, considering object B, ;if the object has a value for θ_Y in the interval 90 to 270 degrees, ;then the orientation of edge E1 would have a positive X-component and a ;negative Y-component.

We use the Algernon relations, `(top-bottom-reversed ?obj <True/False> ?sco)`, `(right-left-reversed ?obj <True/False> ?sco)`, and `(orientation-type ?obj <Inverted/Not-Inverted> ?sco)`, to maintain knowledge of an object's orientation as viewed from positive infinity on the Z-axis. In our magnetic fields application, the need is to know whether a current flow in an object would appear to be in the clockwise or counterclockwise direction. If either `top-bottom-reversed` or `right-left-reversed` is true, then the object would appear to be inverted, and a formerly clockwise current flow would now appear to be counterclockwise, and vice versa. However, if neither is true, or if both are true, then the object would not appear to be inverted.

Modeling the Projected Area of an Object onto Another

The projected area of a 2-dimensional object rotating about the X or Y axis onto the X-Y plane is a function of the area of the object and the cosine of the angle between the plane of rotating object and the X-Y plane. We will henceforth refer to the cosine of this angle as the *projection factor* of the object onto the X-Y plane. When the cosine is zero, the projection of the object onto the X-Y plane is a line, and the projected area of overlap become zero. Otherwise, the projection of the rotating object onto the X-Y plane is a rectangle, which has a non-zero area.

Of more interest in our work is the projected area of a rotating 2-dimensional object onto another 2-dimensional object in the X-Y plane. Here, while it is again possible for the projected area of overlap to be zero when the objects are perpendicular to each other, it is also possible for the projected area of overlap to be zero if the rectangle projected onto the X-Y plane by the rotating object does not intersect the stationary object.

Unfortunately, as described in Chapter 2, in the general case, it is not possible to reason about the latter case with only qualitative information. Instead, in our work, we assume that the projection of a rotating 2-dimensional object always intersects another object in the X-Y plane as long as the rectangular bounding boxes projected by the bounding shapes of the objects onto the X-Y plane intersect.

Therefore, in our work, we only consider two causes of change in the projected area of overlap between two objects: (a) As one of the objects rotates, we consider the effect of the change in the projection factor between the two objects; and (b) we consider the effect of translational motion of either object.

To model the projection factor between a rotating object and the X-Y plane, we independently model the effects of rotation about the X and Y axes, and use qualitative multiplication to determine the net projection factor onto the X-Y plane. Then, as illustrated in the relations section of the view `bb-overlap-view` in Figure 3.8, the projection factor of an object, A, is simply $(\cos\theta_X(A) \times \cos\theta_Y(A))$.

We approximate the projected area of overlap of an object, A, onto an object, B, in the X-Y plane through the equation:

$Proj\text{-}area\text{-}overlap(A, B) = Coplanar\text{-}area\text{-}overlap(A, B) \times Projection\text{-}factor(A)$.

Here, the term $Coplanar\text{-}area\text{-}overlap(A, B)$ is the area of overlap between the bounding boxes projected onto the X-Y plane by the bounding shapes of objects A and B. While this definition does not give the actual projected area of overlap between A and B, it can be used to determine the qualitative direction of change in the projected area of overlap between A and B. It also allows reasoning about two situations when it is unambiguously known that the projected area of overlap between A and B must be zero: (1) when the projection factor between the two objects is zero; and (2) when the bounding boxes of the two objects are not overlapped.

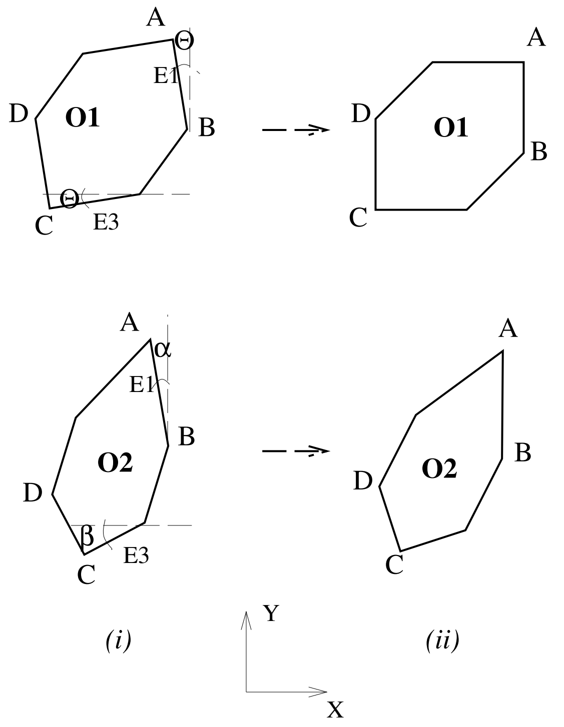


Figure 4.2: For reasoning about rotation around the Z-axis, a more precise model of orientation, with measures of the angles between each surface and each of the coordinate axes, may be necessary to overcome qualitative ambiguity.

4.2.2 Rotation in the X-Y Plane

Rotation about the Z-axis can only change the orientation of an object, and not the projected area onto the X-Y plane. However, unlike rotation around the X or Y axis, which can only *reverse* the identities of the topmost and bottommost vertices or edges, or the rightmost and leftmost vertices or edges, during rotation about the Z-axis, any vertex or edge could become an extremal point or surface.

Figure 4.2 illustrates that with a model of orientation based solely on the identities of the extremal edges and vertices, there can be qualitative ambiguity in determining the next qualitative spatial state that follows a given spatial state when an object is rotating about the Z-axis. In the figure, during a clockwise rotation of object O1 in the X-Y plane, the next qualitative spatial state that follows the state marked *(i)* is one where four of its edges become aligned with the coordinate axes. However, for object O2, which has the same qualitative orientation (in terms of the identities of the extremal edges and vertices) as object O1 in state *(i)*, the next spatial state during clockwise rotation in the X-Y plane is one where edge E1 alone becomes aligned with the Y-axis.

Some qualitative heuristics can be offered for resolving the ambiguity regarding which extremal point or surface of a convex polygon will change identity first during rotation about the Z-axis, as shown below.

- During rotation about the Z-axis, an edge can be an extremal surface for only an instant of time. In the next qualitative spatial state, that extremal point will be formed by the vertex that follows the edge in the direction of rotation.
- Once a given vertex becomes one of the rightmost, leftmost, topmost, or bottommost points, it will retain that role over an angle of rotation of (180 - θ) degrees, where θ is the internal angle at that vertex.
- Therefore, if the internal angle at a vertex is acute, that vertex will always be one of the rightmost, leftmost, topmost, or bottommost extremal points, since, once it takes the role of being a given extremal point, it will retain that role for an angle of rotation of greater than 90 degrees.
- If the internal angle at a vertex is a right angle, that vertex will always be an extremal vertex, or will be a point on an extremal edge.
- If the internal angle at a vertex is obtuse, then that vertex will be a given extremal point for less than a 90 degree rotation.

Although the heuristics given above appear to be promising, in the general case, when there are four extremal vertices, during rotation about the Z-axis, there can still be qualitative ambiguity regarding which of the four extremal vertices will be replaced by an edge first. For a general convex polygon, as shown in Figure 4.2, knowledge of the angle between each edge of the object and each of the X and Y axis of the coordinate system will be required to determine which extremal vertex will be replaced first.

In the figure, for object O, measuring clockwise, the angle between edge E1 and the Y-axis is the same as the angle between edge E3 and the X-axis. Therefore, both edges will become extremal edges at the same time. However, for object O1, these angles, α and β, are not the same. Since $\alpha < \beta$, edge E1 will become an extremal edge before edge E3 during clockwise rotation from the initial spatial state given by *(i)*.

Unfortunately, a method that requires knowledge of the angles between all the edges of all objects and the X and Y axis of the coordinate system is not an *intuitive* solution. We conjecture that for problems of this complexity, a human problem solver would resort to a diagram or some other quantitative aid. Therefore, at this time, we do not provide a general qualitative approach for reasoning about rotation in the X-Y plane.

4.3 Modeling Vectors and Vector Operations

Vectors are used frequently in describing the properties of magnetic fields. The direction of the field itself, the direction of any magnetic force that act on an object in the field, and the direction of motion of an object in the field are all vector quantities. The effects of a magnetic field on objects in the field are often described through vector cross product operations. For example, the magnetic force that acts at a given point on a current carrying conductor placed in a magnetic field is given by $\mathbf{F} = i\mathbf{l} \times \mathbf{B}$, where $\mathbf{F}$ is the magnetic force, i is the magnitude of the current flow, $\mathbf{l}$ is the direction of current flow at the point of interest, and where $\mathbf{B}$ gives the magnitude and direction of the magnetic field at the point of interest.

4.3.1 Modeling Vectors

In our spatial model, a vector is modeled as a triple, $(X\text{-}comp, Y\text{-}comp, Z\text{-}comp)$, where X-comp, Y-comp, and Z-comp denote the components of the vector in each of the X, Y, and Z directions. The qualitative values of each component can be any of $\{+, 0, -\}$, where $+$ denotes a positive value, $-$ denotes a negative value, and 0 denotes a zero component in a given direction.

A vector cannot be modeled directly as a QPC variable since it has three components whose values could be influenced by domain processes. Instead, each vector is implemented through an Algernon predicate(s) in the knowledge base, and through three QPC quantities for each of its components. For example, the translational velocity of an object is given through the predicate `(velocity-vector ?obj ?sco ?vect)` and three QPC quantities: `X-velocity`, `Y-velocity`, and `Z-velocity`. This makes it possible to model influences on the individual components of a vector. For example, the qualitative relations `(d/dt (X-velocity obj) (X-acc obj))` and `(q+ (X-acc obj) (net-force-X obj))` state that the derivative of the X component of the velocity of an object is given by the X component of its acceleration vector, and that the latter is monotonically influenced by the X component of the net force vector that is acting on the object.

The Algernon predicate form of a vector is not used during qualitative simulation. However, it can be referenced in the structural conditions and relations sections of QPC views to affect which views are active in a given spatial state, and also to constrain the initial values of other QPC quantities. For example, as shown in the relations section of the first view in Figure 4.3, the net magnetic force on an object is given through the Algernon predicate `(net-mag-force-vector ?obj ?sco ?vect)` and three additional predicates, `net-mag-force-x-comp`, `net-mag-force-y-comp`, and `net-mag-force-z-comp`, which store the individual components of this vector.

Three QPC quantities, `net-mag-force-x`, `net-mag-force-y`, and `net-mag-force-z` are used to model these components of the magnetic force vector during qualitative simulation. The second view in Figure 4.3 shows that the Algernon predicate form of a vector or component of a vector, in this case `net-mag-force-x-comp`, can be referenced in the structural conditions of a view, and be used to set or constrain the value of the corresponding QPC quantity or quantities. In this example, the relations section of the view `pos-x-dir-mag-force-on-loop` contains the restriction that because the X component of the magnetic force vector must initially be positive based on the structural conditions of view, then the qualitative magnitude of the QPC quantity `(net-mag-force-x loop)` has to be greater than or equal to zero. A zero value is allowed because the X component of the magnetic force vector could be decreasing, and thus, `net-mag-force-x` could become zero at the end of the simulation of the spatial state.

4.3.2 The Qualitative Vector Cross Product Operation

Weinberg, et. al. [88] present a model of qualitative vector algebra which covers the vector addition, substration, and dot product operations. In our work, we extend their results to cover the vector cross product operation.

We define the qualitative cross product operation of two vectors,

$(X_r\mathbf{i} + Y_r\mathbf{j} + Z_r\mathbf{k}) = (a\mathbf{i} + b\mathbf{j} + c\mathbf{k}) \times (d\mathbf{i} + e\mathbf{j} + f\mathbf{k})$,

as follows:

- Let $X1$ be bf, and let $X2$ be -ce.
- Let $Y1$ be -af, and let $Y2$ be cd.
- Let $Z1$ be ae, and let $Z2$ be -bd.
- Then X_r is $X1 + X2$, Y_r is $Y1 + Y2$, and Z_r is $Z1 + Z2$. ;
- Then, the resultant vector after the cross product operation is given ;by $X_r\mathbf{i} + Y_r\mathbf{j} + Z_r\mathbf{k}$.

```
(defview setup-magnetic-force-on-loop-in-field-XY-plane ()
 :individuals
  ((loop :type conducting-loops
         :conditions ((in-XY-plane loop TRUE ?model)
                      (exists-current-flow loop TRUE ?model)))
   (mag-fld :type magnetic-fields
            :conditions ((in-XY-plane mag-fld TRUE ?model)
                         (exists-external-flux-through-loop
                          ?model loop mag-fld))))
 :relations ((:bind ?net-mag-force-dir
                    (get-net-mag-force-dir 'loop 'mag-fld '?model))
             (:bind ?force-x (first '?net-mag-force-dir))
             (:bind ?force-y (second '?net-mag-force-dir))
             (:bind ?force-z (third '?net-mag-force-dir))
             (net-mag-force-vector loop ?model ?net-mag-force-dir)
             (net-mag-force-x-comp loop ?model ?force-x)
             (net-mag-force-y-comp loop ?model ?force-y)
             (net-mag-force-z-comp loop ?model ?force-z)
             (is-active-two-role-view
              ?model 'mag-force-on-loop-in-mag-fld loop mag-fld)
             (exists-mag-force-on-mob-obj loop ?model True)))

(defview pos-x-dir-mag-force-on-loop ()
 :individuals ((loop :type conducting-loops
                     :conditions ((net-mag-force-x-comp loop ?model 1))))
 :relations ((q>= (net-mag-force-x loop) zero)
             (q+ (net-mag-force-x loop) (mag-force-magnitude loop))
             (correspondence (net-mag-force-x loop) zero
                             (mag-force-magnitude loop) zero)))
```

Figure 4.3: Example QPC views which integrate the use of Algernon predicates and QPC quantities in modeling vector objects.

The terms X_r, Y_r, and Z_r are obtained through qualitative addition [43]. More specifically, if $C = A + B$, then:

- If A and B have the same sign, then so will C.
- If A and B have opposite signs, one $+$ and the other $-$, then the value of C will be ambiguous (?).
- If either of A or B is zero, then C will equal the other.
- If both A and B is zero, then C will be zero.
- Otherwise, if the value of either of A or B is ambiguous (?), then the value of C will be ?.

4.4 Implementation of Our Model of Rotation

Our model of rotation is implemented in the form of QPC views. In Section 4.2.1, we discussed that the view `bb-overlap-view`, as shown in Figure 3.8, models the projection factor of a rotating object onto the X-Y plane and the projected area of overlap of the rotating object onto another object in the X-Y plane.

Figure 4.4 shows two other QPC views, `rotation-around-X-axis` and `rotation-theta-y-0-to-90`, which partially model the effects of rotation of an object around the X-axis. In the QPC implementation, there are two quantity types, `theta-X` and `theta-Y`, which correspond to θ_X and θ_Y in our spatial model.

The view `rotation-around-X-axis` simply defines the derivative relationship between the X component of the rotational velocity vector of an object and its `theta-Y` quantity. The view `rotation-theta-y-0-to-90` models the effects of rotation around the X-axis where θ_Y for an object is between 0 and 90 degrees. There are additional views in the implementation to cover θ_Y values of 90 to 180 degrees, 180 to 270 degrees, and 270 to 360 degrees. The views which model rotation around the Y-axis are similar.

```
(defview rotation-around-X-axis ()
 :individuals ((obj :type mobile-objects))
 :operating-conditions ((not (q= (x-rot-velocity obj) ZERO)))
 :relations ((d/dt (theta-y obj) (x-rot-velocity obj))
             (is-active-one-role-view ?model
                                      'v-rotation-around-X-axis obj)))
(defview rotation-theta-y-0-to-90 ()
 :individuals ((obj :type mobile-objects))
 :operating-conditions ((q> (theta-y obj) (theta-y-0 obj))
                        (not (q> (theta-y obj) (theta-y-90 obj))))
 :quantity-types (theta-y-theta-y-90-dif theta-y-theta-y-0-dif)
 :relations
 ((add (theta-y-0 obj) (theta-y-theta-y-0-dif rotation-theta-y-0-to-90)
       (theta-y obj))
  (not (q< (theta-y-theta-y-0-dif rotation-theta-y-0-to-90) zero))
  (not (q> (theta-y-theta-y-0-dif rotation-theta-y-0-to-90) ninety))
  (add (theta-y obj) (theta-y-theta-y-90-dif rotation-theta-y-0-to-90)
       (theta-y-90 obj))
  (not (q< (theta-y-theta-y-90-dif rotation-theta-y-0-to-90) zero))
  (not (q> (theta-y-theta-y-90-dif rotation-theta-y-0-to-90) ninety))
  (q- (theta-y-theta-y-90-dif rotation-theta-y-0-to-90)
      (theta-y-theta-y-0-dif rotation-theta-y-0-to-90))
  (correspondence (theta-y-theta-y-0-dif rotation-theta-y-0-to-90) ninety
                  (theta-y-theta-y-90-dif rotation-theta-y-0-to-90) zero)
  (correspondence (theta-y-theta-y-0-dif rotation-theta-y-0-to-90) zero
                  (theta-y-theta-y-90-dif rotation-theta-y-0-to-90) ninety)
  ; model change in sin and cos theta as a function of theta.
  (U- (theta-y-theta-y-0-dif rotation-theta-y-0-to-90)
      (sin-theta-y obj) (:quote (ninety one)))
  (U- (theta-y-theta-y-0-dif rotation-theta-y-0-to-90)
      (cos-theta-y obj) (:quote (0 one)))
  (correspondence (theta-y-theta-y-0-dif rotation-theta-y-0-to-90) zero
                  (sin-theta-y obj) zero)
  (correspondence (theta-y-theta-y-0-dif rotation-theta-y-0-to-90) ninety
                  (cos-theta-y obj) zero)
  (top-bottom-reversed obj False ?model)
  (q+ (Z-comp-to-cog-from-orig-TM obj) (sin-theta-y obj))
  (q- (Y-comp-to-cog-from-orig-TM obj) (cos-theta-y obj))
  (correspondence (Z-comp-to-cog-from-orig-TM obj) zero
                  (sin-theta-y obj) zero)
  (correspondence (Y-comp-to-cog-from-orig-TM obj) zero
                  (cos-theta-y obj) zero)
  (is-active-one-role-view ?model 'v-rotation-theta-Y-0-to-90 obj)))
```

Figure 4.4: Example QPC views which model rotation around the X-axis.

The relations section of the view `rotation-theta-y-0-to-90` is complex. First, the QPC keyword `:quantity-types` is used to define two local quantity types for the view. These quantity types measure the difference between θ_Y and 0 degrees, and the difference between θ_Y and 90 degrees, as shown in the first nine assertions in the relations section of the view. The angles 0 degrees and 90 degrees are not given as raw numbers, and instead are defined through the quantity types `theta-y-0` and `theta-y-90`. The use of *variables* to model these angles allows us to use the view `rotation-theta-y-0-to-90` to not only model rotation around the X-axis where θ_Y is between 0 and 90 degrees, but any interval of rotation, such as 360 to 450 degrees, which is equivalent to the interval 0 to 90 degrees. We will discuss this in greater detail in Chapter 7, when we present our methods for modeling and reasoning about variables with *circular* quantity spaces.

The tenth through the thirteenth assertions in the relations section of the view provide qualitative constraints for the values of the sine and cosine of θ_Y. These are modeled through the quantity types `sin-theta-y` and `cos-theta-y`.

Next, the relations section contains the assertion that the identities of the topmost and bottommost extremal points/edges would not be reversed while $0 < \theta_Y \leq 90$. Finally, for use in problems involving reasoning about torque, we define two quantity types, `Z-comp-to-cog-from-orig-TM` and `Y-comp-to-cog-from-orig-TM`, to maintain the Y and Z component of the vector from the center of the edge or vertex that was the topmost edge/vertex in the initial spatial state to the center of rotation of the object. During rotation about the X-axis, we maintain two similar quantity types, `Z-comp-to-cog-from-orig-LM` and `X-comp-to-cog-from-orig-LM`, to maintain the X and Z component of the vector from the center of edge or vertex that was the leftmost edge/vertex in the initial spatial state to the center of rotation of the object. Our use of these quantity types will be described in greater detail in Chapter 8, when we present an example which simulates the operation of a motor.

4.5 Chapter Summary

This chapter has presented our methods for modeling and reasoning about the orientation of objects and the effects of rotational motion. Unlike the problem of modeling the relative positions of objects, where simplifying shape abstractions can be used to approximate their spatial extent, to model the orientation of an object, knowledge of the shapes of the actual surfaces of the object is often required.

Our model of the orientation of a convex object is based on knowledge of the identities of the vertices and edges that form its extremal points. We showed that this model could be used in a static world for reasoning about the qualitative surface normal or surface tangent directions of any surface of the object. For example, we showed that our model could be used for describing the intrinsic and deictic front directions of objects, and the direction of current flow at any given point on the surface of a convex object.

We also presented methods for modeling the area of the projection of an object onto the X-Y plane, and for determining the changes in area of the projection if the objects rotates. Other issues that were discussed include (a) determining when a rotating object may appear to be inverted when continuously viewed from the same vantage point, and (b) methods for modeling vectors and for performing the qualitative vector cross product operation.

Chapter 5

The Figure Understander - Fundamentals

Chapters 2-4 presented our spatial representation. The remaining chapters will focus on the implementation and the use of the spatial representation. This chapter describes our methods for providing a problem description and for processing the same. More specifically, ours is the only existing spatial reasoning system which starts with a diagram and text description of a problem and without any further user intervention, produces the qualitative behaviors that are achievable from the initial spatial state.

In this chapter, we will discuss why we use diagrams and text as input and the issues involved in integrating the differing information in the two forms of input into a unified knowledge base description. Then, we will present our solution, the *Picture Semantics* description, which greatly simplifies the task of integrating diagram and text input. We will illustrate the utility of the Picture Semantics description through a natural language understanding example involving understanding a description of an urban scene.

In Chapter 6, we will provide an in-depth presentation of the spatial reasoning features of the Figure Understander. Among the issues discussed will be methods for selecting bounding abstractions for objects, methods for selecting and rotating the frame of reference for the problem, and methods for producing a qualitative description of the initial spatial state of the problem.

5.1 Why Use Diagrams and Text as Input?

We did not appreciate the need for the Figure Understander early in our research. Our initial attempts involved the development of a text-based language to describe the spatial properties of an initial scene. Due to large amount of spatial information used internally, we soon found this approach to be cumbersome. Since many of the textbook magnetic fields problems we have considered as examples in our work had associated diagrams, we then considered the possibility of using diagrams as input to describe the spatial scene. However, since diagram convey only the spatial properties of objects, we needed some other means of inputing the domain-specific properties of the objects in the diagram, and decided to use text for this purpose.

More generally, Larkin and Simon describe the benefits of using both diagrams and text to describe a scene in their classic paper, "Why a Diagram is (Sometimes) Worth 10,000 Words" [48]. Pictures can be used to *compactly* describe the spatial properties of objects in a scene. This allows any accompanying textual descriptions to concentrate on the non-spatial domain-specific and problem-specific (henceforth, *contextual*) properties of objects. To clarify our terminology, the rectangular object in the diagram of Figure 5.1 has the *spatial* property that it is rectangular, and the *contextual* property that it is a magnetic field.

5.2 The Coreference Problem

Diagrams and text typically describe the same object in different ways. Figure 5.1 redisplays the diagram and text statement for the Physics problem [76] introduced in Chapter 1. The diagram objects are lines, polygons, and circles. The text problem

- **Diagram:**

754 / Faraday's Law of Induction

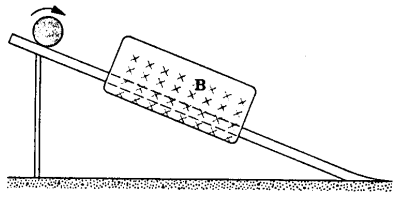


Figure 13 Question 7.

- **Text:** Figure 13 shows an inclined wooden track that passes, for part of its length, through a strong magnetic field. You roll a copper disk down the track. Describe the motion of the disk as it rolls from the top of the track to the bottom.

Figure 5.1: Figure 1.1 redisplayed.

statement uses domain-specific terms, such as 'disk' and 'track', to describe the same objects. Each form of input, diagram and text, contains unique information. The diagram shows the position and shape of the circular object, but it is the word 'disk' in the text that reveals that it is a solid object, and it is the word 'copper' that reveals that it is a conducting object.

The *coreference* problem [64] is to associate object references in text with the appropriate diagram objects. Solutions to the coreference problem are important because the unique information in each form of input, diagrams and text, can be integrated only after the mappings between diagram and textual objects are identified.

To solve the coreference problem, human readers can make use of a large "commonsense" knowledge base to quickly visualize the spatial properties of an object mentioned in a text description and identify the appropriate object(s) in a picture. Similar strategies have been adopted in previous work to automate the ability to solve the coreference problem. One approach is to use model libraries which map domain-specific terms to a spatial model for the object [64], such as the model that a 'car' is a rectangle with four circles attached to it. Another approach is to use an image database to directly associate a picture with a textual term [80, 84].

Both approaches face problems with the completeness of the model library or image database and ambiguities in assessing the quality of a match against a given input. Unfortunately, a given object or collection of objects in a diagram may map to several models in a model library, or objects in the input may have a different shape or orientation, and thus appear to be different, than objects in an image database.

For example, in interpreting the diagram in Figure 5.1, a human reader can easily recognize that the sloping portion of the triangular base forms the 'track', that the vertical piece forms the support for the track, and that the horizontal line is the ground. In a model library, a possible mapping for the word 'track' is that it is a line object in a diagram.

In the text of Figure 5.1, there is only one reference to a 'line' object - the track. The diagram contains three line objects - the track, the support, and the ground. In this case, we would either need a model for a 'supported track' in the library which would include all three line objects in the diagram, or the underlying system would have to perform complex reasoning about the information in the text to associate only the sloping line with the word 'track'.

5.2.1 Overview of Our Goals and Solution

In previous work, researchers have addressed the problem of automatically interpreting text and pictorial input *intended for human use*, such as Srihari's interests in identifying persons in newspaper photographs based on clues in an associated text caption [84], or Novak and Bulko's [64] interest in solving textbook Physics problems using the exact text problem statement and associated diagrams that are available to the human reader. These problem assumptions do not allow any automation-friendly modifications in either the diagram or text input, thus preserving the full complexity of the coreference problem, and making the solutions difficult to implement and use.

For example, in previous work, it is assumed that the pictorial input is provided in visual (raster) form, such as a camera image. In such cases, it becomes necessary to perform visual scene interpretation [4] simply to identify the pictorial objects of interest. There can be many ambiguities. In Figure 5.1, for example, it may be non-trivial to infer that the X's form a pattern to be associated with the rectangular object, or to infer that the label 'B' is attached to the rectangular object, as opposed to the track or to the diagram as a whole.

Our Approach for Solving the Coreference Problem

The Figure Understander addresses the question "Are there any simple, but unrestrictive, methods that allow diagrams and text to be used as computer input?". The goal is to take advantage of the power of diagrams for describing the spatial state [48, 60] and yet retain the ability to use text-based input to describe other contextual properties of interest.

We have developed a novel solution to the coreference problem based on two observations: (1) Annotations are often used in drawings to allow human readers to associate non-spatial contextual information with objects. For example, Figure 5.1 contains two means to identify the rectangular object as a magnetic field: the label, B, and the pattern of Xs; (2) Graphical drawing tools provide several methods, such as colors, filling patterns, and brush types, to highlight objects.

Our solution to the problem of compactly providing both spatial and non-spatial information as input to a given reasoning system has been to automate the ability to associate contextual information with the use of highlighting colors, patterns, and brush types for objects in diagrams drawn through graphical drawing tools. We have a developed a special description language, the *Picture Semantics*, for this purpose.

Advantages of Our Approach

Our approach has two favorable consequences for the coreference problem: (1) The need for additional text input can sometimes be completely eliminated, thus trivially solving the coreference problem. (2) When the need for additional text input persists, our approach allows constraints on both the spatial and contextual properties of objects to be used when matching textual and diagram objects.

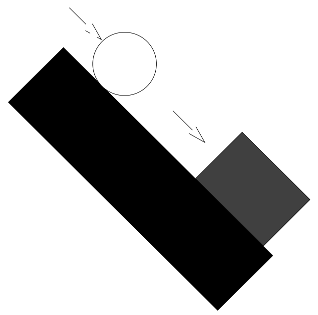
Our solution is straight-forward to implement and use. The assumption that the pictorial input is given through a graphical editor greatly reduces the complexity of diagram interpretation - anything drawn as a distinct object in the diagram can be readily identified as a distinct object from the output of the graphical editor. Further, properties such as brush types, filling patterns, and colors can be directly associated with objects, thus eliminating a source of ambiguity present in interpreting visual input in raster form.

5.2.2 Related Research Issues

The Figure Understander focuses on a very specific problem - providing an efficient method for using text and diagrams as computer input. Some related research issues include the generation of explanations using both text and graphical input [18], generating a pictorial description from a text description of a scene [29, 49, 65], and generating a text description from a visual image of a scene [53].

The solutions to these issues focus less on associating pictorial objects with text objects, and instead, concentrate on the modeling and use of spatial relations. For example, Gapp [29], Latecki and Pribbenow [49], and Olivier, et. al., [65] have all pointed out that the interpretation of distance-based topological relations, such as near and far, and those relating two paths, such as along, can be highly context dependent. They have defined both heuristic and quantitative methods for

- **Diagram:**



- **Text:** The loop is moving down the track.

Figure 5.2: Input given to the Figure Understander for the textbook problem shown in Figure 5.1.

evaluating the applicability of such topological relations to a set of objects in a given spatial scene.

5.3 The Picture Semantics Description

The diagram and text input to the Figure Understander for the problem of Figure 5.1 is given in Figure 5.2. We have modified the problem such that a circular 'loop' is moving down the track instead of a conducting 'disk', as used in Figure 5.1. This is simply to reduce the number of types of objects in the magnetic fields domain model. Conducting loops of wire appear frequently in the other magnetic fields problems we consider, such as those demonstrating the principles of operation of transformers, oscillators, generators, and motors.

Further, we use the term 'moving' out of the field to reason about the motion as a translational motion instead of a rolling motion, since we do not support reasoning about any types of rotational motion other than rotation of an object about an axis passing through its center of gravity. Although there are differences in the solutions of the problems in Figures 5.1 and 5.2, such as whether reasoning about torques and angular velocities is required, the basic magnetic fields properties that

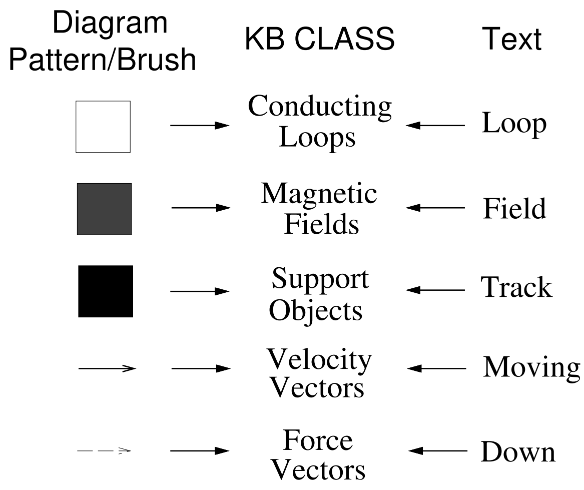


Figure 5.3: Diagram and text semantics for Figure 5.2.

affect the behavior of the moving object are the same in both examples. Further, since the moving object is circular in shape in both problems, we do not have to be concerned about differences in the net effect of the magnetic field on the moving object in each case due to differences in the orientation of the object as it moves or rolls through the field.

In Figure 5.2, the cruder diagram and greatly simplified text form are sufficient to capture all the necessary input information for the problem. This simplification is possible because of the power of our Picture Semantics description, illustrated in Figures 5.3 and 5.4, which allows contextual properties to be associated with the use of brush types, filling patterns, and colors in diagrams.

Incidentally, text labels could also be used in the Picture Semantics if the graphical editor provides a means to directly attach the label to other diagram objects. Otherwise, the task of associating the text label with a diagram object can itself be a challenging problem [32] particularly when the diagram contains overlays. Since the graphical editor we use, the `InterViews` drawing editor, treats text as a separate object, we not use text labels in the Picture Semantics description.

In our approach, we first process the diagram before processing the text input.

The result is a partial knowledge base with instantiations for each diagram object under the appropriate knowledge base class. For the diagram in Figure 5.2, we use the Picture Semantics form to provide the following associations between diagram objects and knowledge base (KB) classes, as illustrated in Figure 5.3: white objects are conducting loops, objects with a dark rectangular mesh pattern are magnetic fields pointed into the page, black objects are supports, vectors with the light, solid brush type are velocity vectors, and vectors with a light, broken brush type are force vectors.

In our application, we always provide a vector in the diagram to provide the direction of motion of objects, intrinsic front directions, and the net force acting on an object. In the example of Figure 5.2, the diagram does not specify which object is in motion, since the velocity vector (solid vector) is floating in free space. This information is obtained from the text input, "the loop is moving down the track".

5.3.1 Solving the Coreference Problem in Figure 5.2

After processing the text input in Figure 5.2, we learn that there must be a mobile, conducting loop in the diagram which is in motion (from 'loop is moving'). Further, we learn that there is contact between the loop and a supporting track (from 'down the track'). Since the loop is in motion, we also expect a vector in the diagram to provide the direction of motion of the loop. Finally, because of the use of the word 'down', we infer that gravity is a factor in this problem, and expect a force vector in the diagram which gives the direction of the initial net force acting on the object.

The coreference problem in processing the diagram and text in Figure 5.2 is to identify the conducting loop, velocity vector, force vector, and supporting track in the diagram, even though the diagram primitives are only lines, circles, and polygons.

The problem is quickly solved by use of the Picture Semantics description, since, when processing the diagram, we have already extracted the information that there are instances of conducting loops, magnetic fields, supporting objects, velocity vectors, and force vectors in the diagram. Thus, once the text processing identifies the need for finding a conducting loop, velocity vector, force vector, and supporting track, we can directly search against the knowledge base for diagram objects with these contextual properties.

Since there is only one object of each KB class in the diagram, the matching process becomes straightforward. We can then assert the direction of motion specified by the velocity vector and the net force direction given by the force vector into the KB description of the loop. In more complex scenes, when there are multiple instances of a given object type in the diagram, spatial constraints in the text, such as the clue that the conducting loop and supporting track must be in contact, can be used to isolate the specific instances of interest. In Section 5.5, we will present an urban scene interpretation example where there are multiple instances of each object type.

5.3.2 Features of the Picture Semantics Description

Figure 5.4 shows a partial Picture Semantics description for our magnetic fields problems. There are four main blocks. Under `:patterns-for-2d-objects` and `:brush-types-for-vector-objects`, the user can assign a meaningful label, such as the label 'mobile-conducting-loop' assigned to the solid-white pattern, for each of the color, filling pattern, and vector types allowed by the graphical editor used to draw the diagram.

The second block (`:object-groups`) allows the user to organize individual labels into groups and to specify the root to be used for the names of the knowledge base instantiations of diagram objects of this group. For example, the group 'magnetic-fields' is defined to include diagram objects with a dark rectangular mesh pattern and those with a horizontal line pattern. The first diagram object encountered with this pattern will have a knowledge base instantiation with the name 'F'.

```
(defpict pictures-for-mag-fields
 :patterns-for-2d-objects
          ; Graphical Editor Pattern      ; Label For Object
           ((dark-rect-mesh                field-into-page)
            (horizontal-line               horizontal-field)
            (solid-black                   track)
            (solid-white                   mobile-conducting-loop)
            (light-cross-mesh              immobile-conducting-loop))

 :brush-types-for-vector-objects
          ; Graphical Editor Brush Type   ; Label For Object
           ((vl-s-vector-out               mot-dir-vect)
            (vl-b-vector-out               perm-force-dir-vect))

 :object-groups
     ; Label For Group  ; Root Name For KB objects  ; Objects In Group
     ((magnetic-fields       F     (field-into-page horizontal-field))
      (conducting-loops      D     (mobile-conducting-loop
                                    immobile-conducting-loop))
      (motion-vectors        MV    (mot-dir-vect))
      (force-vectors         FV    (perm-force-dir-vect)))

 :object-group-semantics
     ((conducting-loops ; Group Label
         ; Properties of group to be asserted into the KB.
         ((member conducting-loops ?obj) ; assert membership in KB class.
          (is-solid ?obj TRUE)))
      (magnetic-fields
       ; Fields in the initial scene are uniform, steady, and permanent.
        ((member steady-uniform-magnetic-fields ?obj)
         (source-of-field ?obj permanent)))
      (motion-vectors
        ((member velocity-vectors ?obj))))

 :individual-object-semantics
     ((conducting-loop  ; Group Label
       (mobile-conducting-loop  ; Object Label
        ; special properties of object type to assert into the KB.
        ((mobility-of ?obj mobile))))
      (magnetic-fields
       (field-into-page
        ((field-direction-of ?obj '(0 0 -1))))))

 :special-procedures (connect-vectors-to-objects))
```

Figure 5.4: A partial Picture Semantics description for our magnetic fields problems.

The second will be named F-1, the third F-2, and so on.

The third block specifies the knowledge base assertions to be made for each individual object type (`:individual-object-semantics`) and for groups as a whole (`:object-group-semantics`). All contextual information is expressed in terms of predicates in the Algernon knowledge representation language [10].

For example, under `:object-group-semantics`, we have specified that all diagram objects of the group 'conducting-loops' are members of the knowledge base class 'conducting-loops', and that they have the property of being solid objects. Under `:individual-object-semantics`, we specify that diagram objects with a solid-white pattern (label 'mobile-conducting-loops') have the additional property that they are mobile.

The fourth block (`:special-procedures`) specifies user-defined, special-purpose procedures to be run after diagram processing is completed. For example, the special procedure 'connect-vectors-to-objects' associates a vector in the diagram with the object which contains the tail of the vector whenever an unambiguous match can be found. The vectors in Figure 5.2 are intentionally not attached to the conducting loop. However, we could have eliminated the need for text input completely by moving the tail of the vectors into the loop, allowing the velocity and force vectors for the loop to be extracted directly from the diagram!

5.4 The Figure Understander

The Figure Understander is an implemented research tool for integrating the information in text and *graphically drawn* diagrammatic input into a single knowledge base description. The intended use of the Figure Understander is to support applications, such as map understanding systems, story understanding systems, or automated problem solving systems, that will benefit from the ability to *compactly* describe the spatial state of the world through a diagram, and yet may require additional text-based input to describe the domain properties of the diagram objects and any dynamic changes that are taking place.

The Figure Understander takes as input a `Postscript` file for a diagram produced using the `Interviews` graphical drawing tool, a Picture Semantics description for the diagram, one or more sentences of descriptive text, and an Algernon [10] knowledge base. It outputs an integrated description of the information in each form of input, diagram and text, into the knowledge base.

5.4.1 Diagram Processing

The strength of the Figure Understander lies in its diagram interpretation capabilities. The Figure Understander extracts and reasons with a *qualitative* internal description of the initial spatial state. The internal description is in terms of the spatial representation described in Chapters 2-4.

As the diagram is processed, a knowledge base (KB) object is created for each diagram object. Both the contextual information given in the Picture Semantics and the qualitative spatial properties of interest are attached to the KB description of the object. The spatial properties extracted include the spatial extent (size, shape, and position), connectivity, and orientation of individual objects, and topological and directional spatial relations between objects. For example, the connectivity of a polygon is stored by creating knowledge base structures for all the edges and vectors, and storing the connections between the edges and vertices. Incidentally, some human errors in drawing the diagrams are automatically corrected by the Figure Understander, such as adjusting for an unintentional gap of a few pixels between objects intended to be touching each other.

5.4.2 Text Processing

The task of attaching semantic information given in any accompanying text sentences to diagram objects requires one to first locate the appropriate diagram objects to which the information is to be attached. We require that the text must

```
(defparameter *semantic-functions*
   '((loop (lambda ()
             (let ((?v1 (make-variable))
                  `(:the ,?v1 (member conducting-loops ,?v1)))))

     (velocity (lambda ()
             (let ((?v1 (make-variable))
                  `(:the ,?v1 (member velocity-vectors ,?v1)))))

     (force (lambda ()
             (let ((?v1 (make-variable))
                  `(:the ,?v1 (member force-vectors ,?v1)))))

     (track (lambda ()
              (let ((?v1 (make-variable))
                   `(:the ,?v1  (member support-objects ,?v1))))) ))
```

Figure 5.5: Examples of semantic interpretation functions used to obtain the conceptual class for textual objects.

provide sufficient information to *uniquely* locate the diagram objects to which the information is to be attached.

The Figure Understander performs text processing after processing the diagram. Since the Picture Semantics provides the conceptual class for each diagram object, if we can obtain the conceptual class of each text object, text and diagram objects could be matched based on their conceptual class. We use domain-dependent semantic interpretation functions, as illustrated in Figure 5.5, to provide the conceptual class for any nouns or pronouns that may be used in the text as reference objects. In Figure 5.5, the function `make-variable` is used to return a variable name, such as ?V1 or ?V2, to be substituted into the `:the` from.

Algernon's [10] definite description facility `(:the)` is used to connect object references in text to objects in the knowledge base (KB). The form `(:the ?v1 (member conducting-loops ?v1))` asks Algernon to find a *unique* object, to be bound to the variable ?v1, that is a member of the knowledge base class

```
Interpreting text sentence:
 (THE LOOP IS MOVING DOWN THE TRACK)

 The following Algernon path shall be asserted:

((:THE ?V1 (MEMBER CONDUCTING-LOOPS ?V1) (MOBILITY-OF ?V1 MOBILE))
 (:THE ?V2 (MEMBER SUPPORT-OBJECTS ?V2) (TOUCHING ?V1 ?V2 P-INIT))
 (:THE ?V3 (MEMBER VELOCITY-VECTORS ?V3)
          (VECTOR-DIRECTION-OF ?V3 P ?VEL-VECT))
 (:THE ?V4 (MEMBER FORCE-VECTORS ?V4)
          (VECTOR-DIRECTION-OF ?V4 P ?FORCE-VECT))
 (VELOCITY-VECTOR ?V1 P-INIT ?VEL-VECT)
 (PERM-FORCE-VECTOR ?V1 P ?FORCE-VECT))
```

Figure 5.6: Algernon query and assert form generated for the text in Figure 5.2.

`conducting-loops`. If it is the case that there is more than one conducting loop in the knowledge base, we can use any additional constraints, such as spatial relations, obtained from processing the text to narrow the search of the knowledge base.

The text processing is accomplished by first parsing a sentence through an ATN grammar, and then interpreting the parse output through a domain-dependent semantic interpreter. For each supported sentence type, the semantic interpreter examines the subject, verb, direct object, etc., and assembles a query-and-assert form in the syntax of the Algernon language for locating the appropriate diagram objects in the knowledge base (query) and attaching any new information given in the text about the object (assert).

Figure 5.6 shows the Algernon form generated after processing the sentence 'The loop is moving down the track'. This sentence was interpreted by a function which processes sentences of the form: (⟨det⟩ ⟨subject⟩ IS ⟨action verb⟩ ⟨clause⟩*).

To process the sentence, the conceptual class of the subject is obtained through a semantic interpretation function, as shown in Figure 5.5. Then, if the action verb is 'moving' or 'rolling', the additional constraint that the subject must be mobile is added, as shown in the first line of the Algernon path in Figure 5.6. The conceptual class of the direct object, the track, is then determined through the semantic interpretation function, and another clause is added to the Algernon query, as shown by the second line of the Algernon path in Figure 5.6. From the phrase 'down the track', the additional constraint that the subject and the direct object must be touching is added to the query. Finally, by convention, the presense of an action verb triggers the search for a velocity vector in the diagram, as shown by the third line of the Algernon path in Figure 5.6, and the presense of the word 'down' triggers a search for a force vector, as shown by the fourth line.

Summarizing, the resulting Algernon path shown in Figure 5.6 contains four queries (preceded by `:the`) and two new assertions. The queries search for diagram objects using both spatial properties and contextual properties. The form, `(:the ?v1 (member conducting-loops ?v1) (mobility-of ?v1 mobile))`, is a query to locate a *unique* diagram object which is a member of the knowledge base (KB) class 'conducting-loops', and which is also a mobile object. The second query is to locate a member of the KB class 'support-objects' such that the conducting loop and support object are touching each other. The third query searches for a velocity vector and retrieves its direction, and the fourth query searches for a force vector and retrieves its direction. The asserts then assign the direction of the velocity vector as the velocity vector of the moving loop, and the direction of the force vector as the net force vector acting on the loop. This force vector is assumed to be a permanent force, as indicated by the use of the scope argument 'p' (problem), instead of 'p-init' (problem initial state).

5.5 A Scene Interpretation Example

We have discussed how Picture Semantics descriptions can be used to provide contextual information, such as the knowledge base (KB) class for each diagram object, directly within diagrams. We showed that when each diagram object is of a unique KB class, the use of Picture Semantics descriptions greatly simplifies the solution to the coreference problem of matching objects in text and diagram input.

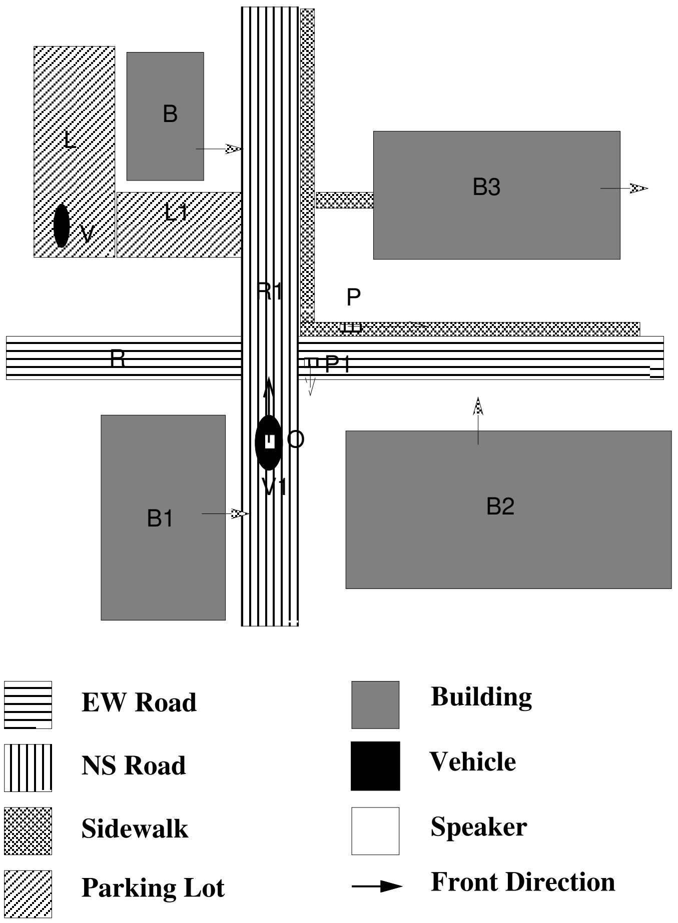


Figure 5.7: An urban scene - buildings, streets, vehicles, and pedestrians. The labels identify the internal names of the knowledge base objects created for each diagram object. The key shows the domain class for each diagram object.

In this section, we consider solution strategies for the coreference problem when there are multiple objects of the same contextual class in the diagram. Essentially, in such cases, we include additional constraining information in the text to uniquely identify the appropriate diagram object. One useful source of additional constraints is the use of spatial relations in the text.

Figure 5.7 shows an input diagram for an urban scene that consists of many lines, rectangles, and ellipses. To assist the reader, Figure 5.7 includes a key which visually presents the Picture Semantics for the diagram. The Picture Semantics form shown in Figure 5.8 provides the contextual knowledge that the objects are buildings, streets, vehicles, and pedestrians. Our goal for this example is to process the text sentences in Figure 5.9 to attach additional contextual properties to descriptions of the diagram objects.

The text sentences in Figure 5.9 make heavy use of spatial relations to assist the reader in locating the diagram objects of interest. For example, to interpret sentence 2, “The name of the building that is to my right is Welch Hall”, we can use the fact that the word ‘my’ refers to the speaker, locate the only speaker (O) in the diagram, and then use the spatial relation ‘building strictly to the right of the speaker’ to locate the only building (B2) that satisfies the relation and label that as ‘Welch Hall’.

5.5.1 Intrinsic/Deictic Frames of Reference

Retz-Schmidt [77] discusses that in human use of spatial prepositions, extrinsic, intrinsic, or deictic frames of reference could be used. The Figure Understander enforces the convention that all objects of interest are mentioned explicitly in the text sentences. Thus, in processing sentence 2, it uses the deictic frame of reference of the speaker as the frame of reference, based on the use of the phrase ‘my right’. However, in processing sentences 3-5, the intrinsic frames of reference of Welch Hall, Taylor Hall, and the Aerospace-Engineering building, respectively, are used instead, since those sentences contain such phrases as ‘in front of Welch Hall’ and ‘behind Taylor Hall’.

The Figure Understander reasons about spatial relations with respect to the extrinsic frame of reference of the diagram. However, to allow reasoning about deictic and intrinsic frames of reference, the Figure Understander’s spatial reasoning module allows queries about directional spatial relations to carry a frame of reference argument, and automatically determines an equivalent relation to prove in terms of the extrinsic frame of reference of the diagram. In Figure 5.7, the vectors define the

```
(defpict roads-and-buildings
 :patterns-for-2d-objects  ;Graphical Pattern  Domain Object Type
                          ((solid-white          speaker)
                           (solid-black          vehicle)
                           (solid-gray           building)
                           (slash                parking-lot)
                           (horizontal-line      NS-road)
                           (vertical-line        EW-road)
                           (wide-rect-mesh       pedestrian)
                           (light-cross-mesh     sidewalk))
 :brush-types-for-vector-objects  ;Graphical Brush Type, Domain Object Type
                                 ((vl-s-vector-out      int-front-vect))

 :object-groups  ; Domain-Class-Name    Abb-Name   Member-Object-Type
                 ((sidewalks             S          (sidewalk))
                  (speakers              O          (speaker))
                  (pedestrians           P          (pedestrian))
                  (roads                 R          (EW-road NS-road))
                  (buildings             B          (building))
                  (int-front-vectors     IF-V       (int-front-vect)))

 :object-group-semantics     ; Semantics given as Algernon predicates.
                         ((buildings    ((member buildings ?obj)))
                          (roads        ((member roads ?obj)))
                          (pedestrians  ((member people ?obj)))
                          (speakers     ((member speakers ?obj))))

 ; format is (class-name (object-type semantics) (object-type semantics)..)
 :individual-object-semantics
                          ((roads
                            (EW-road ((direction-of ?obj EW)))
                            (NS-road ((direction-of ?obj NS)))))
 :special-procedures ((connect-vectors-to-objects)
                      (integrate-partial-objects))
 :principle-axes XZ)
```

Figure 5.8: Example of a partial Picture Semantics for urban scene interpretation.

1. My name is Raman.
2. The name of the building that is to my right is Welch Hall.
3. The name of the building that is in front of Welch Hall is Taylor Hall.
4. The name of the building that is behind Taylor Hall is the Aerospace-Engineering building.
5. The name of the building that is to the right of the Aerospace-Engineering building is the Biology building.
6. I am traveling on Twenty-Fourth street.
7. Speedway street intersects Twenty-Fourth street.
8. Ben is crossing Speedway street.

Figure 5.9: A collection of descriptive sentences for the scene depicted in Figure 8.

```
(interpret '(the name of the building that is to my right is Welch Hall))

 The following Algernon path shall be asserted:

((:THE ?V1 (MEMBER SPEAKERS ?V1))
 (:THE ?V2 (MEMBER BUILDINGS ?V2)
           (:BIND ?VECT (GET-INT-FRONT-DIR-IF-ANY '?V1))
           (DIR-REL-WITH-FOR STRICTLY-RIGHT-OF ?V2 ?V1 SCENE-INIT ?VECT))
 (NAME-OF ?V2 '(WELCH HALL)))
```

Figure 5.10: Example of the Algernon query and assert form generated by the text interpreter after processing sentence 2 in Figure 5.9.

intrinsic front direction for each object. Currently, the Figure Understander only considers intrinsic and deictic frames of reference that are orthogonal to that of the diagram.

Figure 5.10 shows the Algernon form generated after processing the sentence 'The name of the building that is to my right is Welch Hall'. This sentence was interpreted by a semantic interpretation function which processes sentences of the form: (⟨property⟩ of ⟨subject⟩ that ⟨phrase with spatial relation and reference object⟩ is ⟨value⟩).

There are two queries and an assertion. The first form, `(:the ?v1 (member speakers ?v1))` is a query to locate a *unique* diagram object whose conceptual type is 'Speakers'. This was obtained from the semantic interpretation function for the reference object, 'my'. The second (long) form is a query to find a unique building that is strictly right of the speaker (based on intrinsic front direction of the speaker). The assertion in the Algernon path of Figure 5.10 specifies a new fact to be added, the name of the building, should the query identify the appropriate building. In this case, building B2 is named as Welch Hall.

The form `(GET-INTrinsic-FRONT-DIRection-IF-ANY)` returns the intrinsic front direction of an object if one exists. The form `(prove-DIRectional-RELation--WITH-given-FrameOfReference)` accepts a directional spatial relation, the identities of two objects, a scope argument, and a frame of reference argument and proves a corresponding relation with respect to the extrinsic frame of reference of the diagram.

5.5.2 Annotated Demonstration

In the remainder of this section, we shall demonstrate the combined use of contextual and spatial constraints in text to solve the coreference problem in processing the diagram in Figure 5.7 and the text sentences in Figure 5.9. All computer generated output is shown in typewriter font.

```
> (parse-figure "speedway-24th.ps"       ; input diagram
'((my name is Raman))    ; input text
'roads-and-buildings)    ; picture semantics form

***********************************************
***            Parsing The Figure           ***
***********************************************

Extracting spatial information from the figure.
Executing user defined special procedures.
Interpreting text sentence:
 (MY NAME IS RAMAN)

 The following Algernon path shall be asserted:
((:THE ?V1 (MEMBER SPEAKERS ?V1))
 (NAME-OF ?V1 'RAMAN))
```

The Figure Understander is invoked through the `parse-figure` function. The diagram is provided in the form of a PostScript file produced by the InterViews drawing editor. The input also includes a list of the sentences to be processed and the picture semantics form to be used. As Figure Understander processes the diagram, it creates knowledge base objects for all the diagram objects, and stores both the spatial information about the objects and the semantic information given through the picture semantics.

After processing the diagram and the text sentence 'My name is Raman', database entries will exist for all the objects in the diagram, and their internal names will correspond to the abbreviations given in the picture semantics. The internal names for the larger objects in the scene are given as labels in Figure 5.7.

Let us now query the entire set of two dimensional objects in the knowledge base to identify the objects with bindings for the name-of relation. This is accomplished through Algernon's `ASK` function. A list of bindings for the variables (symbols with the initial character,'?') used within the `ASK` function is retrieved using the `:collect` argument.

```
algy> (ask '((member convex-2-d-objects ?x)
             (name-of ?x ?y)) :collect '(?x ?y))
  ((O RAMAN))
```

Note that the only object with a domain-specific name is the speaker, O, whose name is Raman. Next, we shall proceed to process sentences 2-5 from Figure 5.9. Below, we show the Algernon query-and-assert form for the sentence 'The name of the building that is behind Taylor Hall is the Aerospace-Engineering building.' Note that since the intrinsic front of Taylor Hall, given by the vector attached to object B3 in Figure 5.7, points to the right, the relation 'behind' is determined with respect to this new front direction to identify the Aerospace Engineering building, B.

```
algy> (interpret '(The name of the building that is behind Taylor Hall is
                  the Aerospace-Engineering building))

 The following Algernon path shall be asserted:

((:THE ?V1 (MEMBER CONVEX-2-D-OBJECTS ?V1) (NAME-OF ?V1 '(TAYLOR HALL)))
```

```
 (:THE ?V2 (MEMBER BUILDINGS ?V2)
           (:BIND ?VECT (GET-INT-FRONT-DIR-IF-ANY '?V1))
           (GET-DIR-REL-WITH-FOR STRICTLY-BEHIND ?V2 ?V1 ?VECT))
 (NAME-OF ?V2 '(AEROSPACE-ENGINEERING BUILDING)))
```

Set of named objects after processing sentences 2-5 in Figure 5.9.

```
algy> (ask '((member convex-2-d-objects ?x) (name-of ?x ?y))
           :collect '(?x ?y))
  ((B (AEROSPACE-ENGINEERING BUILDING)) (B1 (BIOLOGY BUILDING))
   (B2 (WELCH HALL)) (B3 (TAYLOR HALL)) (O RAMAN))
```

Now let us proceed to name the streets. Below, we show the Algernon query forms generated for sentences 6 and 7 in Figure 5.9. Notice that there are no attempts to identify an intrinsic frame of reference since the spatial relations to be proven, enclosed-in and intersecting, are not directional relations. The 'ask' query at the end reveals that streets R and R1 have been named.

```
algy> (interpret '(I am traveling on Twenty-Fourth street))
 The following Algernon path shall be asserted:

((:THE ?V1 (MEMBER SPEAKERS ?V1))
 (:THE ?V2 (MEMBER ROADS ?V2) (ENCLOSED-IN ?V1 ?V2))
 (NAME-OF ?V2 '(TWENTY-FOURTH STREET))
 (ACTION-OF ?V1 'TRAVELING))

algy> (interpret '(Speedway street intersects Twenty-Fourth street))
 The following Algernon path shall be asserted:

((:THE ?V1 (MEMBER ROADS ?V1) (NAME-OF ?V1 '(TWENTY-FOURTH STREET)))
 (:THE ?V2 (MEMBER ROADS ?V2) (INTERSECTING ?V2 ?V1))
 (NAME-OF ?V2 '(SPEEDWAY STREET)))
```

Set of named roads in the KB after processing sentences 6-7 in Figure 5.9.

```
algy> (ask '((member convex-2-d-objects ?x) (isa ?x roads) (name-of ?x ?y))
   :collect '(?x ?y))
((R (SPEEDWAY STREET)) (R1 (TWENTY-FOURTH STREET)))
```

Let us now consider the last sentence in Figure 5.9, 'Ben is crossing Speedway Street'. This sentence names one of the pedestrians after uniquely identifying the street that the pedestrian is crossing. Note that in addition to the name of the street, Speedway, we could have instead specified the direction of the street, NS or EW, since that is also a known property (given in the picture semantics) that distinguishes the two streets in Figure 5.7. Since it can be inferred that Ben is not the speaker, the query looks for a person other than the speaker.

```
algy> (interpret '(Ben is crossing speedway street))
 The following Algernon path shall be asserted:

((:THE ?V1 (MEMBER ROADS ?V1) (NAME-OF ?V1 (:QUOTE (SPEEDWAY STREET))))
 (:THE ?V2 (MEMBER PEOPLE ?V2)
       (:UNP (ISA ?V2 SPEAKERS)) (ENCLOSED-IN ?V2 ?V1))
 (NAME-OF ?V2 (:QUOTE BEN)))

algy> (ask '((member people ?x) (name-of ?x ?y)) :collect '(?x ?y))
  ((P1 BEN) (O RAMAN))
```

5.6 On Composite Objects

A limiting assumption of our Picture Semantics approach is that diagram objects must be in one-to-one correspondence with the conceptual objects to be created in the knowledge base. Sometimes, it is advantageous to use composite objects in a diagram, such as drawing a car as a rectangular block with four attached circles.

We have implemented a limited capability to include composite objects in the diagram. In Figure 5.7, the parking lot and the sidewalk are drawn as several connected rectangular blocks. The special procedure `(integrate-partial-objects)` is used to automatically assert that the individual pieces are parts of a single object.

Basically, any objects of the same type which are adjacent (contact between pairs of edges) to each other in the diagram are assumed to be parts of a composite object. One of the subparts is selected to be representative of the composite object, and the others are marked explicitly as subparts of the whole. For example, in Figure 5.7, the parking lot, marked with the slash pattern, is composed of two subparts. If L1 is chosen to represent the composite, it may become a member of the KB class parking-lots, while L is marked as a subpart of parking lot L1.

The difficulty is in handling spatial relations involving composite objects. To process a containment relation, such as inside, we should check not only the diagram object selected to represent the composite object, but all of its parts as well. I.e., a car is in the parking lot if it's in either L or L1, not just L1. In such cases, the relation on the composite object becomes an 'OR' of the relation on the subparts. A relation such as 'equals' requires an 'AND' operation on the subparts. Relations such as 'right-of' and 'front-of' are more problematic, and the correct interpretation may be domain dependent. For example, in Figure 5.7, is building 'B' to the right of the parking lot when the parking lot is composed of both L and L1? If we opt to answer such queries through an 'OR' operation on the subparts, then, since building 'B' is the right of 'L', we would answer YES to the query, even though building 'B' is not to the right of 'L1'.

Let us now consider the following sentences:

- John is jogging on the sidewalk.
- My car is in the lot.

Then, since sidewalks are treated as composite objects, and since all the sidewalk segments in the diagram are interconnected, there is only one composite sidewalk in the diagram and it may be uniquely identified. Similarly, there is only one parking lot, and it may be identified uniquely to locate the car belonging to the speaker. The Algernon query and assert forms produced from the interpretation of the two sentences are shown below, followed by the outputs of knowledge base queries to show all named objects and all objects owned by someone.

```
algy> (ask '((member sidewalks ?x) (member parking-lots ?y))
           :collect '(?x ?y))
((S2 L1))

algy> (interpret '(John is jogging on the sidewalk))
 Algy path =

((:THE ?V1 (MEMBER SIDEWALKS ?V1))
 (:THE ?V2 (MEMBER PEOPLE ?V2) (ENCLOSED-IN ?V2 ?V1 SCENE-INIT))
 (NAME-OF ?V2 (:QUOTE JOHN)) (ACTION-OF ?V2 (:QUOTE JOGGING)))
```

```
algy> (interpret '(my car is in the lot))
 Algy path =

((:THE ?V1 (MEMBER PARKING-LOTS ?V1))
 (:THE ?V2 (MEMBER VEHICLES ?V2) (ENCLOSED-IN ?V2 ?V1 SCENE-INIT))
 (:THE ?V3 (MEMBER SPEAKERS ?V3)) (OWNER-OF ?V2 ?V3))

algy> (ask '((member convex-2-d-objects ?x) (name-of ?x ?y))
           :collect '(?x ?y))
((R (SPEEDWAY STREET)) (R1 (TWENTY-FOURTH STREET))
 (B (AEROSPACE-ENGINEERING BUILDING)) (B1 (BIOLOGY BUILDING))
 (B2 (WELCH HALL)) (P JOHN) (P1 BEN) (B4 (TAYLOR HALL)) (O RAMAN))

algy> (ask '((member convex-2-d-objects ?x) (member convex-2-d-objects ?y)
             (owner-of ?x ?y)) :collect '(?x ?y))
((V O))
```

5.7 Evaluation of the Picture Semantics Approach

The Figure Understander addresses the question "Are there any simple, but unrestrictive, methods that allow diagrams and text to be used as computer input?". The goal is to take advantage of the power of diagrams for describing a spatial state [48, 60] and yet retain the ability to use text-based input to describe other contextual properties of interest.

Previous work in the area of integrating text and diagram input to a given problem solving system has resulted in the development of methods for automating the ability to understand text and diagrammatic input intended for *human communication* [32, 64, 80, 84]. This assumption makes the coreference problem particularly complex and difficult for several reasons: (1) There can be hidden assumptions on the background knowledge of the human reader which may be not be available to an automated reasoning system; (2) Even the task of independently understanding each form of input may be difficult. Understanding pictorial input in visual (raster) form requires visual scene interpretation [4], and processing text intended for human-to-human communication can require a sophisticated natural language processing system to resolve the ambiguities that may be present.

We have described a simpler methodology, the Picture Semantics description, and an implemented tool, the Figure Understander, which allows multimedia input to a knowledge base in the form of a graphically-drawn diagram and a collection of text sentences to be integrated into a single unified description.

By using graphical editors to provide diagrammatic input, we gain two advantages over systems that have to deal with visual input. First, graphical editors directly identify the objects of interest by providing a *geometric description* of the objects in the scene. If the additional constraint that diagram objects must be in one-to-one correspondence with the objects to be created in the knowledge base is enforced, there is no need for visual scene interpretation.

Second, graphical editors allow additional properties to be attached to objects such as patterns, colors, and brush types. Our system uses a special input form, the Picture Semantics, to associate contextual knowledge with these graphical properties. This approach provides a simple and elegant solution to the coreference problem by allowing text and diagram objects to be matched based on both spatial and domain-specific contextual information.

5.7.1 Applicability of Our Solution

We impose strict restrictions on the contents of both the diagram and the text. For example, unlike the work of Srihari [84], we do not consider the possibility that there may be object references in the text for which there is no corresponding object in the diagram. To avoid ambiguity in establishing coreference, we also require that there is at least one unique object in the diagram which can be matched against a text object based on its Picture Semantics description alone. We further assume that it will be possible to use spatial relations in the text between such uniquely identifiable objects and the non-unique diagram objects to isolate any specific non-unique diagram object.

The reader may ask, 'Aren't these requirements too restrictive?'. The answer is yes, if, as with the work of Srihari [84], Rowe [80], and Novak [64], the goal is to automatically process text and diagram input intended for human use, such a problem statement in a textbook, or a newspaper photograph. That is why their solutions to the coreference problem are necessarily more complex.

The solutions that we provide are intended for users who simply wish to take advantage of diagrams to compactly input the many spatial properties of a spatial scene, but who need another means, such as text, to provide additional domain-specific information about the objects in the diagram. In these situations, it is quite reasonable to assume that the user has a great interest in providing ambiguity-free input descriptions.

Just as we are making use of the Figure Understander to process diagram and text descriptions of the spatial reasoning problems used to test and evaluate this dissertation research, the work of Hari Narayanan, et al. [60] and He, et al. [32] could also benefit from the Picture Semantics-based approach for using diagrams and text as computer input.

Chapter 6

The Figure Understander - Implementation

This chapter describes the implementation of the Figure Understander. The intent is to provide the reader with an in-depth understanding of the internal qualitative description of the initial spatial state that is extracted from an input diagram. Among the topics discussed are the selection of frames of reference for a problem, the selection of bounding shapes for objects, and the Algernon model (spatial KB) for the spatial properties of objects.

6.1 The Input PostScript File

The Figure Understander processes diagrams drawn through an object-oriented drawing editor, such as the InterViews drawing editor. Currently, its capabilities are limited to processing vectors and closed convex objects, such as circles and polygons, drawn as a single object through one of the drawing primitives of the editor. For example, while a rectangular object drawn through a rectangle icon is allowed, a rectangle drawn as four independent, connected lines is not allowed.

Each object drawn through a drawing primitive is readily identifiable in the PostScript file generated by the drawing editor. Figure 6.1 shows a block of code generated for the rectangle that represents the magnetic field in Figure 5.2. When processing this block, the Figure Understander learns that the object is a

```
Begin %I Rect
%I b 65535
0 0 0 [] 0 SetB
%I cfg Black
0 0 0 SetCFg
%I cbg White
1 1 1 SetCBg
%I p
0.75 SetP
%I t
[ 0.725396 -0.725396 0.725396 0.725396 12.0325 456.445 ] concat
%I
265 204 313 237 Rect
End
```

Figure 6.1: The internal description generated by the InterViews drawing editor for the rectangle that represents the magnetic field in Figure 5.2.

rectangle from the word `Rect` in the first line, and learns that the object has a white pattern from the `0.75 SetP` value for the object's pattern. The last two lines give the coordinates for the vertices and a rotation matrix for the orientation of the rectangle.

6.2 Selecting the Frame of Reference for the Problem

The Figure Understander has the capability to automatically choose a Cartesian frame of reference for a problem. If the chosen frame of reference is different from that of the input diagram, then the diagram is rotated to conform to the newly chosen frame of reference, and all further spatial reasoning is done with respect to the rotated diagram.

6.2.1 Restrictions on Automatic Selection of Frames of Reference

The determination of whether the Figure Understander should choose a frame of reference for a problem is a user option. This feature of the Figure Understander should only be used for problem descriptions where there is no need to process spatial

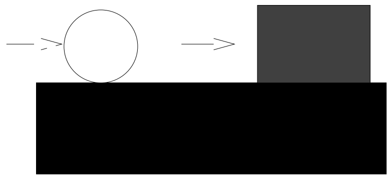


Figure 6.2: Diagram in Figure 5.2 redisplayed within a frame of reference chosen to maximize the number of edges aligned with the coordinate axes.

relations in any accompanying text input. This is because the task of translating a qualitative spatial term such as 'right-of' used with respect to the frame of reference of the diagram to one compatible with a rotated frame of reference is inherently ambiguous in the general case. This has been demonstrated formally by Freksa [27].

For example, the diagram in Figure 5.2 is automatically rotated by the Figure Understander to produce the one in Figure 6.2. The criterion used by the Figure Understander to choose an orientation for the coordinate system is to maximize the number of vectors and edges of objects that are aligned with the coordinate axes. To meet this criterion, any input diagram need not be rotated more than ninety degrees in either the clockwise or counterclockwise direction.

Let us now consider the diagram in Figure 5.2, and assume that it is to be rotated counterclockwise by an angle of less than ninety degrees. Then, the 'left-of and above' relationship between the loop and the field could remain the same, become only 'left-of', or become 'left-of and below' after such a rotation. Although, in this example, we could use the fact that the conducting loop and magnetic field are contacting the track to obtain a unique result, the example still illustrates the ambiguity that can arise in the general case.

In the general case, if the Figure Understander's ability to automatically select a frame of reference for a problem is to be used, then spatial relations cannot be specified in any text sentences that accompany the diagram. To circumvent this restriction, we have adopted the approach of providing vectors in the diagram to specify directions. In our work, we have used vectors to specify the direction of motion of objects, intrinsic front directions, the directions of magnetic fields, and the directions of forces acting on objects. With this approach, if the diagram is rotated, the vectors will also be rotated. For example, for the input diagram and text in Figure 5.2, the reference to the spatial relation, 'down', in the text is not used to obtain the direction of motion of the conducting loop. Instead, it is obtained from the velocity vector given in the diagram. As shown in rotated diagram in Figure 6.2, for the input diagram in Figure 5.2, the velocity vector is also rotated. Therefore, it continues to correctly specify the direction of motion of the loop.

6.2.2 Criteria for Automatic Selection of Frames of Reference

The Figure Understander uses three heuristics to select a frame of reference for a problem, as shown below:

- The origin is chosen such that all objects initially lie in the positive quadrant of the chosen coordinate system.
- The orientation is chosen such that the number of vectors and edges that are aligned with the coordinate axes is maximized.
- When there are conflicting possibilities, priority is given to aligning vectors and the edges of *immobile* objects with the axes of the coordinate system, since their orientation will remain constant.

6.2.3 Benefits of Automatic Selection of Frames of Reference

It is well known that in solving many spatial reasoning problems, the proper selection of the frame of reference for the problem can reduce the complexity of the solution. This is particularly true of problems involving reasoning about vectors and vector operations. For example, in the diagram of Figure 5.2, the qualitative direction of the motion vector of the loop is $(+, -)$. Given this *qualitative* direction of motion, it is possible for the loop to intersect the magnetic field, or to pass above the field. Further, if the loop intersects the magnetic field, there are two possibilities: the loop could intersect only the left edge of the field as it enters, or it could intersect both the top edge and the left edge of the field.

However, if we start with the rotated diagram of Figure 6.2, there is only one possibility. Since the direction of the motion vector is $(+, 0)$, the loop will move to the right, intersect the left edge of the field, and exit to the right (assuming the magnetic field processes acting on the loop do not stop the motion of the loop). In this case, it is clear that the alternate choice for the frame of reference greatly simplifies the solution to the problem. This is because the qualitative vector $(+, 0)$ is more precise than the vector $(+, +)$.

Another benefit of maximizing the number of vectors and edges of objects that are aligned with the coordinate axes of the frame of reference is to reduce the ambiguity in performing the qualitative cross product operation. As shown in Figure 6.3, in case (6.1), when the two vectors in the cross product operation both have non-zero components along each of the X, Y, and Z directions, the X, Y, and Z components of the resultant vector could all be ambiguous (if bf, ce, cd, af, ac, and bd are all positive, for example). In case (6.2), when one of the vectors in the cross product operation has non-zero components along only two of the X, Y, and Z directions, then there can be ambiguity regarding only one component of the resultant vector. In case (6.3), when one of the vectors in the cross product operation is parallel to one of the axes of the coordinate system, then there is no ambiguity in determining the resultant vector.

6.2.4 Implementation of Frame of Reference Selection

The Figure Understander determines the coordinates of the endpoints of all vectors and the vertices of all polygons using the numerical coordinates provided for each object in the input PostScript file. Then, the slope of each vector and each edge of a polygon is determined, and parallel and perpendicular vectors and edges are placed into buckets. The bucket with the maximum number of entries is selected, and the frame of reference of the diagram is rotated based on the slope of the vertices and

1. General Case: Full Ambiguity.

$$(a\mathbf{i} + b\mathbf{j} + c\mathbf{k}) \times (d\mathbf{i} + e\mathbf{j} + f\mathbf{k}) = (bf - ce)\mathbf{i} + (cd - af)\mathbf{j} + (ae - bd)\mathbf{k}. \quad (6.1)$$

2. One Vector Has Only Two Components: Partial Ambiguity.

$$(a\mathbf{i} + b\mathbf{j} + c\mathbf{k}) \times (d\mathbf{i} + e\mathbf{j}) = (ae - bd)\mathbf{k} + cd\mathbf{j} - ce\mathbf{i}. \quad (6.2)$$

3. One Vector Has Only One Component: No Ambiguity.

$$(a\mathbf{i} + b\mathbf{j} + c\mathbf{k}) \times (d\mathbf{i}) = cd\mathbf{j} - bd\mathbf{k}. \quad (6.3)$$

Figure 6.3: Ambiguities associated with the qualitative cross product operation.

edges in the bucket. If there are conflicts between buckets, the one with the largest number of vectors and edges of immobile objects is selected, since their orientation is guaranteed to remain constant.

Once a frame of reference has been selected, the input diagram is numerically rotated and a new PostScript file is generated with all coordinates adjusted to reflect the new positions and orientations of objects. This file is then used as the 'input' diagram for all further spatial reasoning.

6.3 Qualitative Description of the Initial Spatial Scene

The main task for the Figure Understander is to convert the numerical description of the spatial scene provided in the input PostScript file (the original one or the one generated by the Figure Understander) to an equivalent qualitative knowledge base description.

First, an Algernon knowledge base object is created for each diagram object, and its shape (rectangle, polygon, circle, etc.) is recorded. Then, based on the patterns and brush types used for the diagram, any associated contextual information given in the Picture Semantics description for each diagram object is attached to the knowledge base description for the object. Then, individual Algernon objects are created for each edge and vertex of a diagram object and the perimeter of each polygon is recorded.

Next, the coordinates of the extremal points/surfaces of each object are determined, and for convex polygons, the identities of the extremal edges and vertices are recorded. Then, based on the identities of the extremal edges and vertices, the qualitative orientation of all other edges is determined, as defined in Chapter 4.

Next, the bounding shapes of objects are chosen, their projected rectangle onto the X-Y plane is computed, and the relative positions of objects in the initial spatial state are determined. Once these tasks are completed, then any accompanying text sentences are processed as described in Chapter 5, and any resulting new assertions are made into the knowledge base. Finally, for problems to be solved through qualitative simulation, a QPC-compatible description of the initial state of the problem (the `scenario` description) is determined, and QPC is invoked to solve the problem.

The spatial KB description of the rectangle that models the magnetic field in Figure 6.2 is shown in Figure 6.4. The relations shown in the figure are described in greater detail in the following sections.

6.3.1 Spatial KB Description of Object Boundaries and Orientation

Whether a problem involves reasoning about a static scene or a dynamically changing spatial scene, the starting point of the reasoning process is an Algernon knowledge base description of the spatial properties of the objects in the initial scene. The knowledge base class hierarchy (`taxonomy`) and the relations (`slots`) of interest for spatial reasoning are defined in an Algernon-based spatial knowledge base (spatial KB), as was described in Section 1.4.4.

In the spatial KB, the connectivity of a convex polygon is described in terms of the slots shown in Figure 6.5. The slot `Perimeter-of` is used to store a list of the edges and vertices of a polygon in clockwise order. The list is of the form (V_0, E_0, V_1, E_1, ..., V_n, E_n), where vertex V_0 is the clockwise left endpoint of edge E_0,

```
algy> vf f
F:
  name:       ((f) nil)
  isa:        (f-selfset nil) (convex-polygons nil) (convex-2-d-objects nil)
              (permeable-objects nil) (immobile-objects nil)
              (magnetic-fields nil) (steady-uniform-magnetic-fields nil)
              (objects nil) (world-objects nil) (things nil)
  number-of-vertices: (4 nil)
  is-rectangular: (true nil)
  mobility-of: (immobile nil)
  is-conducting-object: (false nil)
  field-type-of: (uniform nil) (steady nil)
  is-solid:   (false nil)
  field-direction-of: ((0 0 1) nil)
  edge-of:    (f-e-4 nil) (f-e-3 nil) (f-e-2 nil) (f-e-1 nil)
  vertex-of:  (f-v-4 nil) (f-v-3 nil) (f-v-2 nil) (f-v-1 nil)
  perimeter-of: ((f-v-1 f-e-1 f-v-2 f-e-2 f-v-3 f-e-3 f-v-4 f-e-4) nil)
  connected-edges: ((f-e-3 f-v-3 f-e-4) nil) ((f-e-2 f-v-3 f-e-3) nil)
                   ((f-e-1 f-v-2 f-e-2) nil) ((f-e-4 f-v-1 f-e-1) nil)
  connected-vertices: ((f-v-4 f-e-4 f-v-1) nil) ((f-v-3 f-e-3 f-v-4) nil)
                      ((f-v-2 f-e-2 f-v-3) nil) ((f-v-1 f-e-1 f-v-2) nil)
  cog-xval:   ((751/2 p-init) nil)
  cog-yval:   ((432 p-init) nil)
  has-bottommost-edge: ((true p-init) nil)
  bottommost-edge: ((f-e-4 p-init) nil)
  bottommost-vertex: ((f-v-1 p-init) nil) ((f-v-4 p-init) nil)
  bottommost: ((351 400 415 p-init) nil)
  has-leftmost-edge: ((true p-init) nil)
  leftmost-edge: ((f-e-1 p-init) nil)
  leftmost-vertex: ((f-v-2 p-init) nil) ((f-v-1 p-init) nil)
  leftmost:   ((351 415 449 p-init) nil)
  has-topmost-edge: ((true p-init) nil)
  topmost-edge: ((f-e-2 p-init) nil)
  topmost-vertex: ((f-v-3 p-init) nil) ((f-v-2 p-init) nil)
  topmost:    ((351 400 449 p-init) nil)
  has-rightmost-edge: ((true p-init) nil)
  rightmost-edge: ((f-e-3 p-init) nil)
  rightmost-vertex: ((f-v-4 p-init) nil) ((f-v-3 p-init) nil)
  rightmost:  ((400 415 449 p-init) nil)
  no-torque-point: ((f-e-4 edge-center) nil) ((f-e-3 edge-center) nil)
                   ((f-e-2 edge-center) nil) ((f-e-1 edge-center) nil)
  box-bounding-shape: (true nil)
```

Figure 6.4: Example of the internal spatial KB description generated after processing the rectangle that represents the magnetic field in Figure 6.2.

```
(:slot Perimeter-of (convex-polygons nil))
(:slot Edge-Of (convex-polygons edges))
(:slot Vertex-Of (convex-polygons vertices))
(:slot Connected-edges (convex-polygons edges vertices edges))
(:slot Connected-vertices (convex-polygons vertices edges vertices))
(:slot Q1EV (convex-polygons perimeter scopes))
(:slot Q2EV (convex-polygons perimeter scopes))
(:slot Q3EV (convex-polygons perimeter scopes))
(:slot Q4EV (convex-polygons perimeter scopes))

; Are the topmost/bottommost points reversed due to rotation?
; Are the rightmost/leftmost points reversed due to rotation?
(:slot top-bottom-reversed (convex-2-d-objects booleans scopes))
(:slot right-left-reversed (convex-2-d-objects booleans scopes))

; Does an object appear inverted when viewed from +infinity on the Z-axis?
; Orientation-types can be inverted or not-inverted
(:slot orientation-type (convex-2-d-objects orientation-types scopes))
```

Figure 6.5: Slots in the spatial KB for modeling the connectivity of a polygon object and the orientation of its surfaces.

vertex V_1 is the clockwise right endpoint of edge E_0, and so on.

To assist in reasoning about a particular surface of a polygon, the slot `Connected-Edges` stores the edges connected by a vertex, and the slot `Connected-Vertices` stores the vertices that form the endpoints of a given edge. For example, for a convex polygon O, whose perimeter is formed by the vertices and edges (V_0, E_0, V_1, E_1, ..., V_n, E_n), then the assertions (`Connected-Edges O` E_0 V_1 E_1) and (`Connected-Vertices O` V_0 E_1 V_1) will both be made into the knowledge base by the Figure Understander. These relations can be used, for example, to determine the sequence of changes in the identities of the edges and vertices that make up the the extremal points/surfaces of an object as it rotates in the X-Y plane.

Spatial KB Model of the Orientation of an Object

To store the orientation of the surfaces of an object, there are four slots, `Q1EV`, `Q2EV`, `Q3EV`, and `Q4EV`, which respectively store the names of the edges and vertices which face the first quadrant of the extrinsic coordinate system (i.e., have a qualitative orientation of $(+,+)$), the second quadrant (i.e., orientation of $(-,+)$), the third quadrant (i.e., orientation of $(-,-)$), and the fourth quadrant (i.e., orientation of $(+,-)$). These slots have a scope argument, so that as objects rotate, we can maintain the qualitative orientation of each surface in each qualitatively distinct spatial state.

It is possible that during rotation, an object can appear to be inverted when viewed from positive infinity on the Z-axis. In such situations, it becomes important to adjust the signs for such variables as the current flow in a closed conducting loop and the projected area of an object onto the X-Y plane to reflect the change in orientation.

To recognize situations when an object may appear to be inverted, we use three slots, `top-bottom-reversed`, which states that the topmost and bottommost points are reversed in comparison with the initial spatial state, `right-left-reversed`, which states that the rightmost and leftmost points are reversed, and `orientation-type`, which models whether an object would appear to be inverted or not in a given spatial state. Figure 6.6 shows the Algernon rules in the spatial KB for determining whether an object would appear to be inverted when viewed from positive infinity on the Z-axis based on the predicates `top-bottom-reversed` and `right-left-reversed`.

Modeling the Extremal Points/Surfaces of Objects

The identities of the edges and vertices which form the extremal points/surfaces of an object are maintained through 8 slots, as shown in Figure 6.7. We use four additional slots, `Has-Topmost-Edge`, `Has-Bottommost-Edge`, `Has-Rightmost-Edge`, and `Has-Leftmost-Edge`, to indicate, for a given spatial state, whether any of the extremal points/surfaces of an object is formed by an edge.

```
(:rules convex-2-d-objects
  ((orientation-type ?obj reversed ?sco)
   <-
   (top-bottom-reversed ?obj True ?sco)
   (right-left-reversed ?obj False ?sco))

  ((orientation-type ?obj reversed ?sco)
   <-
   (top-bottom-reversed ?obj False ?sco)
   (right-left-reversed ?obj True ?sco))

  ((orientation-type ?obj not-reversed ?sco)
   <-
   (top-bottom-reversed ?obj False ?sco)
   (right-left-reversed ?obj False ?sco))

  ((orientation-type ?obj not-reversed ?sco)
   <-
   (top-bottom-reversed ?obj True ?sco)
   (right-left-reversed ?obj True ?sco)))
```

Figure 6.6: Rules in the spatial KB for determining whether an object would appear to be inverted when viewed from infinity on the positive Z-axis.

6.3.2 Spatial KB Description of the Relative Positions of Objects

As described in Chapter 3, the relative positions between objects are modeled in terms of the relative positions of the rectangles projected onto the X-Y plane by the bounding shapes chosen for objects. In this section, we will first describe how bounding shapes are chosen for objects and then will present how the relative positions of the projected rectangular bounding boxes are derived.

Choosing Bounding Shapes for Objects

The heuristics for choosing a bounding shape for an object are straightforward. If the domain model for object reveals that it is immobile, then we know that the object cannot rotate, and a minimum bounding rectangle is used to approximate its position and spatial extent. For mobile objects, the default is to use a bounding sphere abstraction since it is possible for that these objects could rotate.

```
(:slot Topmost-Edge (convex-polygons edges scopes))
(:slot Topmost-Vertex (convex-polygons vertices scopes))
(:slot Bottommost-Edge (convex-polygons edges scopes))
(:slot Bottommost-Vertex (convex-polygons vertices scopes))
(:slot Rightmost-Edge (convex-polygons edges scopes))
(:slot Rightmost-Vertex (convex-polygons vertices scopes))
(:slot Leftmost-Edge (convex-polygons edges scopes))
(:slot Leftmost-Vertex (convex-polygons vertices scopes))

;;; some boolean flags regarding the extremal points of an object
(:slot Has-Topmost-Edge (convex-polygons booleans scopes))
(:slot Has-Bottommost-Edge (convex-polygons booleans scopes))
(:slot Has-Rightmost-Edge (convex-polygons booleans scopes))
(:slot Has-Leftmost-Edge (convex-polygons booleans scopes))
```

Figure 6.7: Slots in the spatial KB for tracking the identities of the edges and vertices that form the extremal points/surfaces of an object.

However, as part of the input to the Figure Understander, the user may set the values of the flags `*no-translation*` and `*no-rotation*` to true to indicate that for the given problem, translational and/or rotational motion is not possible. These flags reflect known mechanical constraints on the mobile objects in a problem and are used to restrict the behavioral paths that are explored during qualitative simulation.

For example, in a motor, there are mechanical constraints on the rotor such that translational motion is not possible. Similarly, in a transformer, neither translational motion or rotational motion is possible. In any event, while selecting bounding shapes for objects, if the `*no-rotation*` flag is set to `True`, then minimum bounding rectangles are used as the abstract shape for all objects.

As described in Chapter 2, if a bounding sphere is used for an object, its center of gravity is computed numerically, and the furthest distance from the center of gravity to any point on the surface of the object is computed. This is used as the radius of the bounding sphere, and the coordinates of the extremal points of the bounding sphere are determined by adding or subtracting the length of the radius

```
(:rules convex-2-d-objects
      ; compare a rightmost and a leftmost point.
     ((rightmost ?obj1 ?xval1 ?ylow1 ?yhigh1 ?sco)
      (:test (eq (quote ?sco) ,initial-model-name))
      (member convex-2-d-objects ?obj2)
      (leftmost ?obj2 ?xval2 ?ylow2 ?yhigh2 ?sco)
      (:neq ?obj1 ?obj2)
      (:test (epsilon> ?xval2 ?xval1))
      ->
      (:lisp (setq *initial-ordinal-relations*
                   (cons '(q< (rm-xval ?obj1) (lm-xval ?obj2))
                         *initial-ordinal-relations*)))))
```

Figure 6.8: Example of an Algernon rule in the spatial KB for extracting a qualitative description of the relative positions of two objects from a quantitative description of the positions of their bounding boxes.

to the X and Y coordinate values of the center of gravity.

Determining the Relative Positions of Objects

As described in Section 2.4.4, the Algernon relations `Rightmost`, `Leftmost`, `Topmost`, and `Bottommost` are used to store the physical coordinate values of the extremal edges of the rectangles projected onto the X-Y plane by the bounding shapes of objects. Then, production rules of the form described in Figure 6.8 are applied to this quantitative description to extract a qualitative description of the relative positions of objects in the initial scene.

The rule in Figure 6.8, from the spatial KB, is expressed in the syntax of the Algernon [10] knowledge representation language. Recall from Section 2.4.4 that the predicates `Rightmost` and `Leftmost` hold the physical coordinate values for the rightmost and leftmost points of the bounding boxes computed for objects. This rule finds two convex 2-dimensional objects and compares the X-coordinate values of the rightmost and leftmost points of the bounding boxes of the objects. The test for the the binding for the scope variable, `?sco`, ensures that only initial values are being compared.

The comparison operation is performed by the function `epsilon>`. This function does not directly compare two physical coordinate values. Instead, it takes into account the possibility of human error in drawing the diagram, and allows two physical coordinate values to be considered as equal if there are within a predefined epsilon measure. Currently, the epsilon measure used is 6 pixels.

The result of executing rules such as the one in Figure 6.8 is the generation of two lists, `*initial-ordinal-relations*` and `*permanent-ordinal-relations*`. The former contains all the inequality relations on the extremal points of the bounding boxes of objects that are true in the initial state. The latter contains all inequality relations on the extremal points of the bounding boxes that must be true in all states, such as `(q< (rm-xval obj) (lm-xval obj))`.

Incidentally, note that the syntax of the operating conditions in the QPC views, `aligned-X-view` and `left-of-view`, as shown in Figure 3.8 in Chapter 3, precisely matches the syntax of the qualitative description of the spatial state stored in the lists `*initial-ordinal-relations*` and `*permanent-ordinal-relations*`. Thus, not only are these two lists used by the Algernon implementation of our model of relative positions, as illustrated in the rules of Figures 3.4 and 3.5, they can also be passed directly to QPC to describe the relative positions of objects in the initial spatial scene.

6.4 Summary of Chapters 5 and 6

Chapters 5 and 6 have described the Figure Understander, our tool for processing diagram and text input provided to a knowledge base. Diagrams are useful for compactly describing a spatial scene, and text is useful for describing any non-spatial contextual properties of the objects in the scene. Integrating diagram and text input can be difficult because each form of input describes the objects in the scene in a different way. The problem of establishing *coreference* is to determine the correspondence between diagram objects and textual references to objects, so that the properties of the object given in each form of input can be integrated into a single unified knowledge base description.

Our solution to the coreference problem was to provide a method, Picture Semantics descriptions, for directly specifying domain-specific contextual information in the diagram. When each diagram object is of a unique domain class, we showed that the picture semantics descriptions alone are often sufficient for solving the coreference problem. In cases where there are multiple instances of objects of a single domain class in the diagram, we showed that the Picture Semantics descriptions are still useful for narrowing the set of diagram objects that need to be matched against a text object. Then, we showed how additional spatial information in the text could be used to isolate the specific diagram object of interest.

The Figure Understander outputs a qualitative knowledge base description of the spatial properties of the objects in an input scene in terms of the spatial representation described in Chapters 2, 3, and 4. In Chapter 5, we described our method for providing a Picture Semantics description for a domain, and discussed how the Figure Understander processes text sentences.

In this chapter, we described the Figure Understander's algorithms for choosing bounding shapes for objects, for automatically choosing frames of reference for a diagram, and for compensating for simple human errors in drawing the diagram. We also described the internal structures of the spatial knowledge base (spatial KB) that we have developed to describe the spatial properties of objects. The spatial KB contains primitives for describing the connectivity of polygons, the bounding shapes of objects, the relative positions of objects, and the orientation of objects.

Chapter 7

Enhancements For Qualitative Modeling

This chapter describes three important extensions to the modeling capabilities of QPC [17, 16]. These extensions all address general modeling needs that are insufficiently supported by the existing qualitative modeling and simulation systems, such as QPC [16], QPE [20], and QSIM [42]. We found the modeling capabilities of these systems to be insufficient for modeling the following features of a given domain.

First, whereas QPC and QPE require model variables to have continuous values, in many domains, the values of model variables can change discontinuously. To meet the need for including such variables in a qualitative model, we have developed an extension to QPC for modeling piecewise-continuous variables.

Second, while both QPC and QPE allow the creation of new entities during a simulation, once created, there is no means to state that an entity ceases to exist. Thus, in these systems, it is possible for the structural conditions of a view to satisfied although, in reality, one of the individuals required for the view to be active has ceased to exist. To avoid erroneous predictions, the modeler must test the existence of an entity indirectly in the operating conditions of the view. That is, the only way to infer that an entity has ceased to exist is to detect that the value of some continuous parameter of the entity has reached a value that indicates that the entity no longer exists, such as the amount of an entity reaching a value of zero.

In our work, we have to model spatially distributed entities which can appear and disappear discontinuously. For example, the magnetic field around a current carrying conductor can vanish if the current flow stops in the current carrying conductor. The current flow could stop *discontinuously* due to a physical break in the circuit, or as with the example of the disk rolling down the track in Figure 1.1, the induced current flow in a closed conducting object can start or stop discontinuously due to changes in the spatial state of the world. When the existence of an entity can change discontinuously (e.g., through an external action such as toggling a switch), it may be insufficient to model the existence of the entity based on the value of one of its continuous parameters. Therefore, we have developed an extension to QPC for directly modeling piecewise-continuous entities.

Third, all of the existing qualitative modeling system only allow parameters with linear quantity spaces to be modeled. That is, these systems map the magnitude of the qualitative value of a variable to point on the *real number line* that represents a qualitative landmark value of interest in the simulation, or to a point in an interval between two landmark values of interest. However, for some variables, such as the angle of rotation of an object around a given axis, the space of qualitative values of interest is cyclic. Whereas the angle of rotation may start at 0 degrees, and increase through 90, 180, 270, 360, 450, etc., it is often important, and sufficient, to capture the fact that a rotation of 0 degrees is essentially equivalent to a rotation of 360 degrees (one exception is when the number of revolutions made by an object is an issue). To successfully model such variables in QPC, we have added the ability to directly assert that some variables have a *circular quantity space*.

7.1 Modeling Piecewise Continuous Variables

Several qualitative modeling tools, such as QPC [17] and QPE [20], require model variables to be continuous. For many problems, it is possible for certain model variables to be piecewise-continuous. For example, consider Forbus's example [20]

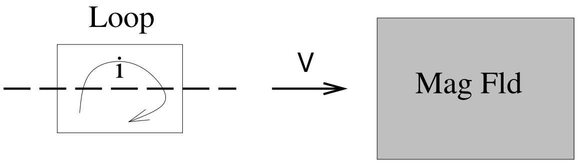


Figure 7.1: If the loop moves into the magnetic field, the derivative of the area of overlap between the two could change discontinuously. If the loop rotates around the axis shown, the direction of the current flow in the loop can appear to change discontinuously from clockwise to counterclockwise when viewed from a point on the positive Z-axis.

of boiling water in a closed container. In any qualitative reasoning system that assumes that all model variables are continuous, it will only be possible to determine that due to increasing pressure, the container could burst. However, it will not be possible to carry the simulation beyond this point since the state of existence of the container, and the pressure in the container, both change discontinuously once the container bursts.

Piecewise-continuous variables also exist in problems involving reasoning about dynamic change in the spatial properties of a system. For example, Figure 7.1 shows a rectangular wire loop moving towards a magnetic field with rectangular boundaries. When the loop enters the field, the derivative of the area of overlap between the two objects will be positive and proportional to the velocity of the loop. Once the loop is fully enclosed in the field, the derivative of the area of overlap will drop discontinuously to zero. This derivative is a *piecewise-continuous* variable - one whose value is continuous within a given world model, but which may change discontinuously at a transition point, when there is a qualitatively significant change in the world state.

For another example of a piecewise-continuous variable, assume that the wire loop in Figure 7.1 has an initial clockwise current flow, and consider what happens as the loop rotates about the axis shown in the figure. By convention, initially, the value of the current flow variable will be positive since the current flow is in the clockwise direction.

After a rotation of ninety degrees, the projection of the wire loop onto the plane of the figure will be reduced to a line, and for an instant of time, the direction of current flow will be undefined when we continue to view the figure from above. As the loop continues to rotate, in the rotation interval of 90 degrees to 180 degrees, the current flow in the loop would appear to be counterclockwise, and therefore, would have a negative value. Thus, the value of a constant current flow in a rotating wire loop can transition piecewise-continuously from a positive value to a negative value due to a change in the orientation of the wire loop.

7.1.1 Modeling Piecewise-Continuous Variables in QPC

We have implemented a solution in QPC for modeling *piecewise-continuous* variables, variables whose value can change discontinuously, but only after a transition point in the qualitative state. Before studying this solution, let us first briefly digress and reexamine the operation of QPC.

Given an initial state and a library of model fragments that capture the behavioral laws for some domain, QPC determines which model fragments are applicable, and forms a unified model which includes all the known constraints that govern the behavior of a domain over the time interval when the model is active. From this collection of constraints, a *qualitative differential equation*, or QDE, is created and passed to QSIM for qualitative simulation. The QDE includes information about the transition points of the model, i.e., points where a qualitatively significant change in the world state takes place. From the point of view of QPC, the transition points are those states of the qualitative simulation where currently instantiated model fragments could become inactive and where currently inactive model fragments could become instantiated. QSIM stops the simulation when it reaches a transition point, and passes the values of all variables back to QPC. QPC uses this new set of values to recompute the set of applicable model fragments and forms a new model and QDE for further simulation.

Unfortunately, if QPC inherits the value of a piecewise-continuous variable into the new description of the world state after the simulation of a previous model reaches a transition point, there could be a contradiction regarding the value of the variable in the next model constructed by QPC. For example, in Figure 7.1, considering the transition point when the rectangle in motion becomes fully enclosed within the second rectangle, the derivative of the area of overlap between the two objects will have a positive value in the QPC model prior to the transition, and a zero value in the model after the transition. If the positive value for the derivative of the area of overlap is inherited into the new QPC model after the transition, there will be a contradiction regarding the value of this variable in the new model.

Our solution was to provide the ability to explicitly declare that a variable is piecewise-continuous within a QPC view, or model fragment. Then, we changed the behavior of QPC at a transition point so that the qualitative values of piecewise-continuous variables and any inequality constraints involving such variables are never inherited into the description of the new world state after a transition point.

More specifically, the declaration `(piecewise-continuous-variable ?model ?quantity)` can now be used in the relations section of a QPC view to define that the variable `?quantity` is piecewise-continuous within any model where the view is active. The variable `?model` is automatically bound by QPC to the current model. Then, any values of this variable or inequality constraints involving this variable are never inherited into the description of the new world state after a transition point. An example of the use of this declaration may be found in the view `bb-overlap-view` in Figure 3.8.

This technique may be used for any dependent variable, since by definition of a *dependent* variable, its value can be recomputed from the values of the independent variables in any subsequent model. The technique can also be used for any independent variable whose new value is explicitly given in the subsequent model. In the current implementation, variables that are dependent on piecewise-continuous variables must also be modeled as piecewise-continuous (by the model builder) since their values could also change discontinuously. The results of Brajnik [6] could be used to remove this restriction on the model builder. Brajnik is using knowledge of dependency relations between variables to automatically infer, given a set of user-defined piecewise-continuous variables, the set of other model variables that must also be modeled as piecewise-continuous variables.

The ability to directly model piecewise-continuous variables will allow many new situations to be modeled in QPC. For example, if Forbus's example [20] of heating water inside a covered pot were implemented in QPC, previously, it would have been possible only to simulate the fact that the cover could be blown off the pot. The consequences of this event could not be simulated since the pressure inside the pot would change discontinuously. Our extensions now allow such piecewise-continuous variables to be modeled directly. In future work, the applicability of this technique for modeling changes in discrete variables, such as the position of a switch, could also be investigated.

7.2 Modeling Piecewise-Continuous Entities

Similar to the definition of piecewise-continuous variables, piecewise-continuous entities are those whose existence can change discontinuously after a transition point in a qualitative simulation. In our work, an example of a piecewise-continuous entity that is encountered is the magnetic field generated by a current flow in a conductor. The field can appear and disappear discontinuously based on whether there is a current flow in the conductor.

QPC provides two methods, the `defentity` macro, and the knowledge base declaration `(entity ?model ?object)` for defining that an object is an active entity in a model. The limitation is that once an entity has been defined, there is no means for declaring that it no longer exists. The adverse consequence of this behavior of QPC is that the structural conditions in views involving the entity could be satisfied

```
(defview boiling-example ()
 :individuals ((liq :type liquids)
               (pot :type containers
                    :conditions ((contained-in liq pot ?model))))
 :operating-conditions ((q> (amount liq) zero))
 :relations (...))
```

Figure 7.2: A example of a view whose operating conditions are used to monitor the existence of one of the individuals.

even though the entity no longer exists.

For example, consider the process of boiling, as modeled in Figure 7.2. Boiling is a process that acts on a liquid entity and transforms it to a gas. Once the liquid is gone, boiling can no longer take place.

In QPC, there is no direct means to model that the liquid is gone. Once it is declared that a liquid entity exists in the knowledge base, QPC assumes that is always exists. Thus, the structural conditions of the boiling view shown in Figure 7.2 will always be satisfied. To avoid the problem that the boiling view could erroneously remain active once the liquid is gone, as shown in Figure 7.2, the model builder must reference the value of a continuous parameter of the liquid (e.g., its amount) in the *operating conditions* of the view to infer when the liquid no longer exists.

This technique can be used only when the existence of an entity can be associated with a continuous parameter of the entity. Now consider the example of Figure 1.1 in Section 1.3.4 of Chapter 1. This problem showed a conducting disk rolling into a magnetic field. Alternately, the reader may consider the scene in Figure 7.1 with the additional provision that there is no initial current flow in the wire loop. In either case, initially, there is no current flow in the conducting object. Once the disk/loop begins to enter the permanent magnetic field shown in the respective figures, an induced current flow appears in the conducting object, and a second magnetic field is *discontinuously* created. This new magnetic field ceases

```
(defview create-field-from-current-in-loop ()
 :individuals ((loop :type conducting-loops
                     :conditions ((exists-current-flow loop TRUE ?model))))

 :relations ((:forc ?cf-fld
                    (source-of-field ?model ?cf-fld loop)
                    (isa ?cf-fld current-flow-magnetic-fields)
                    (piecewise-continuous-entity ?model ?cf-fld))))
```

Figure 7.3: A view in the magnetic field domain model which makes use of piecewise-continuous entities.

to exist *discontinuously* once the induced current flow stops. This happens once the disk/loop is fully enclosed in the permanent magnetic field. This example illustrates a case where it is not possible to model the existence of an entity using a continuous property of the entity. For these cases, we have developed a more direct means for modeling in QPC that the existence of an entity is piecewise-continuous.

7.2.1 Modeling Piecewise-Continuous Entities in QPC

We now allow the QPC model builder to directly assert that a newly created entity may exist only piecewise-continuously. This is done through the declaration `(piecewise-continuous-entity ?model ?entity)`. Its usage is shown in the QPC view in Figure 7.3. This view creates a piecewise-continuous magnetic field entity when there is current flow in a closed wire loop. In the view, the Algernon query `(:forc ?var ?cond1 ?cond2 ... ?condn)` *f*inds *or* *c*reates a knowledge base object, `?var`, such that all the conditions `?cond1` to `?condn` are satisfied.

As with our solution for modeling piecewise-continuous variables, to model piecewise-continuous entities, we have modified the behavior of QPC after a transition point is reached during qualitative simulation. Previously, entities were always inherited into the description of the world state after a transition point. Now, all knowledge of a piecewise-continuous entity is ignored at a transition point, and is never inherited into the description of the subsequent world state.

7.2.2 A Limitation of Our Solution

One limitation of our solution for modeling piecewise-continuous entities is that there isn't any means to declare that two piecewise-continuous entities created in two different models are the same. In other words, our solution can sometimes be too restrictive. By never inheriting information about piecewise-continuous entities after a model transition, we do not allow the possibility that the same entity continues to exist in a subsequent model. Instead, with our solution, a new piecewise-continuous entity will always be created in a subsequent model.

7.3 Modeling Circular Quantity Spaces

A third, ubiquitous modeling restriction imposed by the existing qualitative modeling systems is that the range of values for a model variable can be given only through a *linear* quantity space. For modeling such phenomena as rotation, it is beneficial to model some variables through a circular quantity space to capture their cyclic nature.

Consider the variable θ_Y, which, as defined in Chapter 4, measures the angle of rotation of an object with respect to the X-axis. The qualitatively significant values for θ_Y include 0, 90, 180, 270, and 360 degrees. We would like to directly state that the value of 360 degrees is functionally equivalent to zero degrees (resulting in a circular quantity space), instead of having to model, in a linear quantity space, that 450, 540, etc., are also significant values. This allow us to describe the effects of rotation of an object around the X-axis *compactly* through only 4 views; one each for the rotation intervals 0 to 90 degrees, 90 to 180 degrees, 180 to 270 degrees, and 270 to 360 degrees, as described earlier in Section 4.4 of Chapter 4.

A linear quantity space cannot be used to model the fact that a θ_Y value of 360 degrees is essentially equivalent to a value of 0 degrees, since it is impossible to assert that $(0 < 270)$, $(270 < 360)$, and $(0 = 360)$ in a linear quantity space without a contradiction.

7.3.1 Definition of a Circular Quantity Space

Every variable that has a circular quantity space also has a linear quantity space. A linear quantity space, $(l_1, l_2, ..., l_n)$, for a variable V is a totally ordered set of the qualitatively significant landmark values, l_i, for V where each landmark value, l_i, maps to a single point on the real number line.

A circular quantity space, $(lc_1, lc_2, ..., lc_k)$, for a variable V is a circular set of at least three landmark value classes, lc_i, where lc_1 follows lc_k, but where $lc_1 < lc_2 < ... < lc_k$. Each landmark values class, lc_i, is a set of functionally equivalent landmark values from the linear quantity space for V. Two landmark values l_a and l_b of a variable, V, are *functionally equivalent* if in the domain of interest, a qualitative magnitude for V of l_a can be treated as being equivalent to a qualitative magnitude of l_b, even though when mapped onto the real number line, one may be greater than the other. For example, we treat the values 0, 360, and 720 for θ_Y as being functionally equivalent.

Two further restrictions that we impose on the definition of a circular quantity space for a variable, V, are that (a) each landmark value l_i in the linear quantity space of V must be mapped on a one-to-one basis to a landmark value class in its circular quantity space; and (b) if the landmark value l_i is a member of the landmark value class lc_i, we require that if lc_{i+1} follows lc_i in the circular quantity space of V, and if l_{i+1} follows l_i in its linear quantity space, then l_{i+1} must be a member of the landmark value class lc_{i+1}.

Example of a Circular Quantity Space

To illustrate the concepts introduced above, let us now examine the circular quantity space for the variable θ_Y. A partial linear quantity space for θ_Y is (..., 0, 90, 180, 270, 360, 450, 540, 630, 720, ...). Since the values 0, 360, and 720 are functionally equivalent, we can place them together in a landmark value class, lc_1. This forces the landmark value class lc_2 to contain the landmark values 90 and 450. Then, the value class lc_3 is forced to contain the values 180 and 540, and the value class lc_4 is forced to contain the values 270 and 630. Then, the only circular quantity space definition for θ_Y that satisfies all the conditions given in the previous section is (lc_1, lc_2, lc_3, lc_4).

7.3.2 Use of a Circular Quantity Space

A variable with a circular quantity space can have an infinite number of landmarks in its linear quantity space. However, each of the landmark value classes in its circular quantity space represents one of a finite number of *relative values* for the variable. Our treatment of variables with circular quantity spaces proceeds under the assumption that all reasoning about such variables can be done in terms of their value with respect to the relative values represented by each of the landmark value classes. I.e., although a landmark value class can contain an infinite number of landmark values, since they are all functionally equivalent, we treat each landmark value class as representing a single relative value for a given variable.

Then, during qualitative simulation, we can treat landmark value classes in a circular quantity space like landmark values in a linear quantity space. For example, given a circular quantity space $(lc_1, lc_2, lc_3, ...lc_n)$ for a variable V, if the value of V lies between lc_2 and lc_3, then, if V is increasing, then the next qualitatively significant value of V will be the relative value represented by lc_3. If V is decreasing, it will decrease towards the relative value represented by lc_2.

Thus, in any QPC model, we only require a description of a linear subset of the circular quantity space of a variable. I.e., given the circular quantity space $(lc_1, lc_2, ..., lc_n)$, while it is the case that asserting both $lc_1 < lc_2 < .. < lc_n$ and $lc_n < lc_1$ into a QPC model will result in a contradiction, by asserting only a linear subset of the circular quantity space into any given QPC model, we avoid the contradiction.

More formally, given a variable V with a circular quantity space given by $(lc_1, lc_2, ..., lc_n)$, in any qualitative state, there are only two possible cases for the magnitude of V: (a) It either equals one of the landmark values in one of the landmark value classes in the circular quantity space; or (b) it lies between two landmark values that map to two consecutive landmark values classes in the circular quantity space.

We can model the value of V in case (a) as $V = lc_i$, where lc_i contains the landmark value that defines the actual value of V. Then, we only have to include the inequality relations $V \geq lc_{i-1} \wedge V \leq lc_{i+1}$ in a QPC model to capture all the possible values that V can reach during a qualitative simulation starting from the state where $V = lc_i$. For case (b), we can express the value of V as $lc_a \leq V \leq lc_b$, where lc_a and lc_b are the landmark values classes that contain the landmark values between which V lies. These inequality relations are again sufficient for capturing all the possible values that V can reach before a qualitative simulation reaches a transition point regarding the value of V.

The remaining requirement is a method for controlling the inheritance of the values of circular quantity space variables into the new world state after the qualitative simulation reaches a transition point. For example, given the circular quantity space, (lc_1, lc_2, lc_3), if a QPC model contains the assertions (i.e., linear subset) $lc_1 < lc_2 < lc_3$, and the qualitative simulation of the model reaches a transition point when the value of a variable V reaches lc_3, then in the next model after the transition, we will need to assert the linear subset $lc_2 < lc_3 < lc_1$ to describe the circular quantity space. This would contradict the assertions in the previous model.

To resolve this problem, in a QPC model, we treat the landmark values classes in a circular quantity space as constant valued, but piecewise-continuous, parameters. Thus, inequality relations involving the relative values defined by the landmark value classes in a variable's circular quantity space are never inherited. Instead, the linear subset of a circular quantity space that is appropriate for a variable in each QPC model is computed and asserted directly in each model.

7.3.3 QPC Implementation of Circular Quantity Spaces

In QPC, the declaration `(quantity-type ?type-name ?domain-class)` is used to define `?type-name` as a quantity type for the domain class or classes given by `?domain-class`. A linear quantity space may then be defined for this new quantity type which includes all the qualitative landmark values of interest.

Similarly, we use the declaration `(circular-qspace-quantity ?type-name ?domain-class ?landmark-value-classes)` to define a variable `?type-name` that has a circular quantity space. The landmark value classes for this variable are given directly by the argument `?landmark-classes`, which is a list (cls1, cls2, .., clsn), where the magnitude of the quantity `?type-name` can increase from a landmark value in `cls1`, through a landmark value in `cls2`, up to a value in `?clsn`, but after which the magnitude once again becomes equal to one of landmark values in the landmark value class `cls1`.

For example, the declaration

```
(circular-qspace-quantity
  theta-y convex-2-d-objects
  (theta-y-0 theta-y-90 theta-y-180 theta-y-270))
```

defines the variable `theta-y` as a quantity for the domain class `convex-2-d-objects`. It has a circular quantity space with four landmark value classes represented by `theta-y-0`, `theta-y-90`, `theta-y-180`, and `theta-y-270`, where each of these contains an infinite set of landmark values from the linear quantity space of `theta-y`.

We treat `theta-y-0`, `theta-y-90`, `theta-y-180`, and `theta-y-270` as piecewise-continuous variables that are constant within any QPC model. We maintain the inequality relationships between θ_Y and each of θ_{Y0}, θ_{Y90}, θ_{Y180}, and θ_{Y270} to model the relative value of θ_Y. For example, the same inequality relations, ($\theta_Y > \theta_{Y0}$) and ($\theta_Y < \theta_{Y90}$), can be used to capture the fact that θ_Y is between 0 and 90 degrees or between 360 and 450 degrees.

We avoid introducing contradictions in a QPC model by asserting only a linear subset of the circular quantity space into each model. We only consider the relationship between a circular-valued variable, such as θ_Y, and three of the landmark values classes in its circular quantity space. For example, assume for a rotating object, we have that θ_Y is equal to θ_{Y0}, is less than θ_{Y90}, and is greater than θ_{Y270}. We may model θ_Y to be greater or less than θ_{Y180}, and not both, to linearize the circular quantity space since we know that θ_Y cannot equal θ_{Y180} without first crossing through θ_{Y90} or θ_{Y270}, assuming that a model transition occurs at those points.

We use our ability to model piecewise-continuous variables to capture the fact that the inequality relationships between a variable such as θ_Y and any of its landmark values classes, such as θ_{Y0}, θ_{Y90}, θ_{Y180}, and θ_{Y270}, can change discontinuously at a transition point. The appropriate inequality relations are automatically asserted into the subsequent QPC model after a transition. For example, assume that an object is rotating in a counterclockwise direction about the X-axis and that there is a model transition when θ_Y equals θ_{Y90}. Then, at the transition point, we ignore all previous inequality relations between θ_Y and the landmark values classes in its circular quantity space, and insert into the subsequent QPC model after the transition that $(\theta_Y = \theta_{Y90})$, $(\theta_Y < \theta_{Y180})$, and $(\theta_Y > \theta_{Y0})$. The relationship between θ_Y and θ_{Y270} is insignificant in the new model and may be ignored, or modeled as $<$ or $>$, since the relationship in the previous model ($>$) will not have been inherited.

7.4 Chapter Summary

In this chapter, we described three important extensions to the modeling capabilities of the QPC [17, 16] qualitative modeling system. These include: (a) the ability to model piecewise-continuous variables - variables whose values can change *discontinuously* at a transition point during qualitative simulation; (b) the ability to model piecewise-continuous entities - model entities whose existence can change discontinuously at a transition point; and (c) the ability to model variables that have a circular quantity space - where a linear quantity space with an infinite number of landmarks can be collapsed into a finite circular set of landmark value classes each containing an infinite number of functionally equivalent landmark values from the linear quantity space. These extensions will not only allow a variety of new situations to be modeled and simulated using QPC, these ideas can also be used to enhance the modeling capabilities of other qualitative modeling systems as well.

Chapter 8

Demonstration and Evaluation

The previous chapters have described our methods for processing the text and diagram description of a problem to be solved, for selecting bounding abstractions for objects, and for modeling and reasoning about the relative positions and orientation of objects. We also described several extensions to the modeling capabilities of the QPC qualitative modeling and simulation system to support qualitative reasoning about the effects of dynamic change in the spatial configuration of a system, such as that caused by translational and rotational motion.

This chapter presents the experiments that we conducted to empirically evaluate the usefulness of the tools we have developed for qualitative spatial reasoning. Our experiments involved using our implemented tools for solving a number of problems from two different domains, understanding urban scenes and magnetic fields problems. The solutions to these problems fully cover the spatial reasoning issues addressed in this dissertation.

The complete set of examples demonstrates our claim that we are able to smoothly integrate text and diagram input given to a system, and that we are able to reason about the spatial properties of position and orientation in both static and dynamically changing worlds where translational or rotational motion may be taking place. Further, these examples demonstrate the claim that the spatial representation and the tools we have developed allow the smooth integration of the ability to reason qualitatively about simultaneous dynamic change in both the spatial and non-spatial properties of a physical system.

The remainder of this chapter is organized as follows. First, we shall briefly describe the examples used to evaluate our methods for addressing spatial reasoning problems which concern a static world. Then, the bulk of the chapter will concentrate on problems involving reasoning about dynamically changing worlds. The examples used in the evaluation may be grouped in eight classes of similar problems, where the solutions for each class of problems requires addressing a specific set of spatial reasoning issues. We will present an overview of the full set of examples used in the evaluation process, and highlight the spatial reasoning issues covered by each example class.

In the latter half of this chapter, and in appendix A, we will present the solutions for a representative set of five problems, including the solution to the problem of the loop moving down the track, and solutions to problems which require reasoning about the principles of operation of transformers, oscillators, and generators.

8.1 Static World Examples

Our spatial representation has been implemented as production rules in Algernon [10], and as model fragments in a QPC [9, 16] model library. The rule-based implementation is intended for reasoning about problems involving static worlds, and forms the basis of the spatial reasoning component of the Figure Understander.

We have also used the rule-based, static-world, implementation of our spatial model in two other applications. First, as demonstrated in Chapter 5, we have used this model to interpret natural language text and diagram descriptions of an urban scene. Although we provided clues about the conceptual classes of objects in the diagram through the use of Picture Semantics descriptions, because there were multiple instances of some classes of objects in the diagram, an alternate means was required to solve the coreference problem of associating diagram and text objects. For this problem, we used spatial relations in the text to assist in uniquely identifying a specific diagram object. The rule-based implementation of our spatial representation was used to prove spatial relations between objects in the diagram.

Second, the static-world spatial model has also been used for reasoning about the configuration of road networks, such as determining the number of lanes available for travel in a given direction for a given road. This was accomplished by using the spatial reasoning capabilities of the Figure Understander to associate lane objects with road objects based on the spatial relationships between the objects in a diagram of a network of roads. By studying the conceptual type of the lane objects (e.g., right lane, middle lane, left lane, turning lane) and the directions of vectors which gave the direction of motion along a given lane, the number of lanes available for travel in a given direction was determined.

These examples served two purposes: first, the power of our techniques for integrating diagram and text input to a knowledge base was demonstrated. Second, the power and generality of our Algernon rule-based implementation of our spatial model was tested. We not only demonstrated the ability to reason about both topological and directional relations between objects, we also showed the ability to reason about changes in orientation due to the on-demand use of a deictic or intrinsic frame of reference, as opposed to the extrinsic frame of reference of the diagram.

The remainder of this chapter will present problem sets which require reasoning about dynamic change in the spatial properties of a physical system.

8.2 The Implemented QPC Model Libraries

Our methods for spatial reasoning about dynamic change in the relations positions and orientation of objects, including methods for reasoning about the effects of translational and rotational motion, have been implemented as model fragments (views) in the QPC [9, 17] qualitative modeling and simulation system. These views form a domain-independent model library for qualitative spatial reasoning, and may be referenced in any given domain model.

We have also constructed an extensive QPC view-based domain model for reasoning about the properties of magnetic fields. More specifically, we consider problems involving the induction of current flow in closed conducting objects where there is a change in the total magnetic flux passing through the object (henceforth, Φ). We consider changes in Φ due to translation and rotational motion of the conducting object within the magnetic field, and changes due to a time varying magnetic field.

If a current flow is established in the conducting object, we consider the effects of any magnetic forces that may act on the object. These effects may cause either, or both, translational and rotational motion. We also model the effects of the magnetic field that is dynamically created by current flow.

In addition to the magnetic fields domain model, we have also constructed a QPC domain model for reasoning about the motion of automobiles. Based on this domain model, we constructed examples that successfully demonstrated our ability to reason about possible collisions between multiple moving automobiles.

8.3 Overview of the Magnetic Fields Examples

The magnetic fields examples are the most complex in the set of examples used to evaluate our results. The magnetic fields domain was chosen as a test domain because many of the problems in this domain require reasoning about simultaneous dynamic change in both spatial and non-spatial properties. More specifically, in these problems, changes in the spatial state often lead to the activation of domain processes which in turn oppose the change in the spatial state. The magnetic fields examples also served to test our methods for performing the qualitative vector cross product operation, our methods for modeling and reasoning about piecewise continuous variables and entities, and our methods for modeling and reasoning about variables with circular quantity spaces.

8.3.1 Classes of Magnetic Fields Problems

In our experiments, we considered magnetic fields examples that involve reasoning about scenes with no motion, but where there are changes due to time-varying magnetic fields, examples that involve translational motion through a steady, uniform magnetic field, and examples that involve rotation within a steady, uniform magnetic field.

The first class of magnetic fields problems of interest involves reasoning about situations where these is no motion, but where there is dynamic change in the spatial configuration of a system due to the dynamic creation or destruction of spatially distributed objects, such as the magnetic field created around a current carrying conductor. One specific example from this class of problems is reasoning about the principles of operation of a transformer. Here, a changing current flow in one closed conducting object generates a time-varying magnetic field which in turn causes changes in the magnetic flux passing through a second closed conducting object. The dynamic change in the magnetic flux passing through the second object results in the induction of current flow in that object.

The second class of magnetic fields problems of interest involves reasoning about the effects of translational motion of a closed conducting object through a steady, uniform magnetic field. These problems require the ability to reason about changes in the relative positions of objects due to translational motion, particularly changes in the area of overlap between objects. These problems also require the ability to reason about the orientation of the individual surfaces of objects, so that the direction of any magnetic forces which may act on the conducting object can be determined.

Two examples in this problem class include reasoning about the effects on a conducting loop as it moves down a track and through a magnetic field, as discussed in Chapter 5, and reasoning about the operation of an oscillator. In the latter problem, if a current carrying closed conducting loop is placed partially within a magnetic field, magnetic forces can act on the loop and cause translational motion of the loop. In cases where there is no loss in energy, for certain spatial configurations of the loop and the field, and given a particular direction of current flow in the loop, it is possible for the conducting loop to oscillate back and forth within the magnetic field. For other spatial configurations and directions of current flow in the loop, it is possible for the loop to simply be flung out of the field.

The third class of magnetic fields problems of interest are those which involve rotational motion. More specifically, we considered the operation of generators, where a current flow is generated when rotation causes changes in the magnetic flux passing through a closed conducting object, and the operation of a motor. In a motor, the magnetic forces that acts on a current carrying object placed in a magnetic field cause a net torque on the loop, which results in rotational motion. These examples served to demonstrate our ability to reason about changes in the projected area of overlap between two objects, and the effects of changes in orientation due to the rotation of objects.

8.4 Spatial Reasoning Isses Covered by the Examples

Table 8.1 summarizes the spatial reasoning methods highlighted by each of our eight example sets. During testing, we considered many variations on these examples, involving differences in the directions of permanent magnetic fields, differences in the shapes of objects (we considered circular and rectangular objects to take advantage of their special properties), differences in the initial position and orientation of objects, differences in the direction of motion of objects, and differences in the initial direction of current flow in objects.

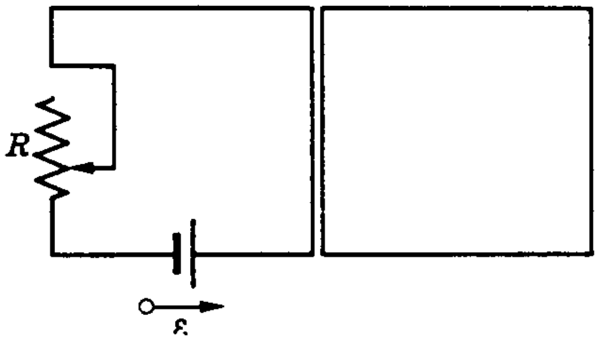
In the sections below, we will present four illustrative examples, covering the operation of transformers, oscillators, and generators. We will also present the solution for the problem of the loop moving down a track into a magnetic field,

Problem Features, Methods Tested	ScUnd	RdUnd	Auto	Trns	Osc	Rmp	Gen	Mtr
Static World Problem	X	X						
Dynamic Change Problem			X	X	X	X	X	X
Diagram Processing	X	X	X	X	X	X	X	X
Text/Diagram Integration	X			X		X	X	
Int/Deictic Ref Frames	X							
Alt Frame Ref Selection						X		
Relative Positions	X	X	X	X	X	X	X	X
Orientation of Surfaces					X	X	X	X
Orientation of Vectors	X	X	X	X	X	X	X	X
Translational Motion			X		X	X		
Rotational Motion							X	X
Projection onto XY Plane							X	X
Inversion of Objects							X	X
Magnetic Forces					X	X	X	X
Torque							X	X
Vector Cross Products					X	X	X	X
Piecewise-Cont Vars				X	X	X	X	X
Piecewise-Cont Objects				X	X	X		
Circular Quantity Spcs							X	X
Multiple Moving Objects			X					
Motion In Multiple Dirs			X		X	X		

Table 8.1: Table showing the spatial reasoning features that were highlighted by each of the problem classes we have considered in evaluating our work. The problem classes are: ScUnd - urban scene understanding, RdUnd - understanding road configurations, Auto - motion of automobiles, Trns - transformers, Osc - oscillators, Rmp - problems involving translational motion of a conducting object through a magnetic field, Gen - generators, Mot - motors.

as described in Chapter 5. Due to space limitations, we will provide only a partial presentation of the output produced for these examples by our spatial reasoning system, the Motion Understander. A complete presentation of the output of the Motion Understander for a fifth example, determining the results of pulling a conducting loop out of a magnetic field, is given in Appendix A. It may be helpful to the reader to skim the contents of Appendix A to obtain a more complete understanding of the operation of QPC, the qualitative simulation process, and the equations which describe the magnetic fields processes of interest in our work, before studying the examples presented in this chapter.

- **Diagram:**



- **Text:** The resistance R in the left-hand circuit of Fig. 20 is being increased at a steady rate. What is the direction of the induced current in the right-hand circuit?

Figure 8.1: A magnetic fields problem which illustrates the basic principles of operation of a transformer. This is problem 15, page 754, in Resnick and Halliday's Physics text, *Fundamentals of Physics* [76].

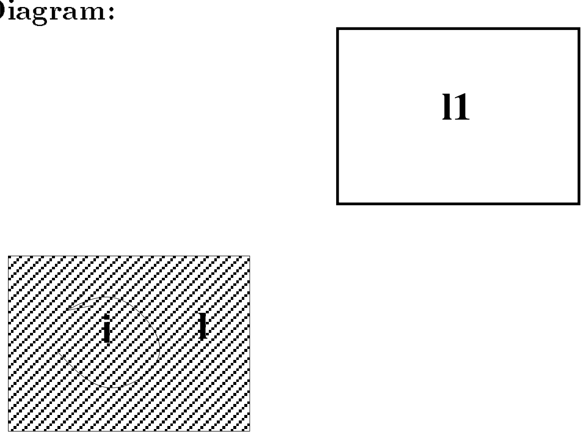
8.5 A Transformer Example

This section presents an example which illustrates the basic principles of operation of a transformer. The example is similar to the problem from Resnick and Halliday's Physics textbook [76], as shown in Figure 8.1. The input we gave to the Figure Understander for this problem is shown in Figure 8.2. We have altered the relative positions of the two loops simply to illustrate the fact that we model the relative positions of objects along both the X and Y axes.

8.5.1 Problem Description

The diagram in Figure 8.2 shows a conducting loop, labeled **l**, where there is an active current flow. Henceforth, we shall refer to this as the active loop. The diagram also shows a passive conducting loop, labeled l1, which is disconnected from the active loop. The magnitude of the current flow in the active conducting loop is decreasing

- **Diagram:**



- **Text:** The current flow in the loop is increasing.

Figure 8.2: Input given to the Figure Understander as our version of the Physics problem described in Figure 8.1. The text says the current flow is increasing because the *negative* counterclockwise current flow is increasing towards zero. In the diagram, the labels for the objects and the arrow for the direction of current flow have been added to assist the reader.

due to increasing resistance in the loop. This causes a time-varying magnetic field to be generated which results in an induced current flow in the passive loop.

In our magnetic fields domain model, we provide the knowledge that for a closed conducting object, the field generated by current flow will have one direction within the object, and the opposite direction outside the object. The direction of current flow is given by the right hand rule [76], as stated below:

Right Hand Rule: Grasp a conductor with the right hand with the thumb pointing in the direction of current flow. The fingers will curl around the conductor in the direction of B, the magnetic field.

In this example, for a counterclockwise current flow in the first loop, the net magnetic field outside the loop will be pointed into the page, in the negative Z direction.

The magnetic field outside the object is not uniform, and in fact, decreases rapidly with distance from the object. However, this property is not a factor in determining the direction of the induced current flow in the second loop, and may be ignored. If we were interested in calculating the strength of the current flow induced in the second object, the distance between the two objects will be a factor.

8.5.2 What Does This Example Illustrate?

This example is of interest because it illustrates our ability to reason about a dynamically created spatial object - in this problem, a magnetic field with infinite spatial extent. To reason about the direction of the magnetic field that passes through the passive conducting loop, we simply study the spatial relationship between the active (l) and passive (l1) conducting loops. Once we have the knowledge that the passive loop lies outside the active loop, we can deduce that the magnetic field through the passive loop will be in the negative Z direction.

Since the strength of the field will be weakening due to the increasing resistance in the active loop, the magnetic flux passing through the passive loop will change over time. Then, an electromotive force (emf) will be induced in the passive loop such that the resulting current flow will oppose the change in the magnetic flux passing through the loop. For this, the current flow must generate a magnetic field whose direction is -Z within the passive loop. This is accomplished by the generation of a clockwise current flow in the passive conducting loop.

In our domain model, we do not consider self-induction, and therefore, the effects of the magnetic field generated by the induced current flow in the passive loop are ignored. However, we do record the existence of the second magnetic field generated by the newly established current flow in the passive conducting loop.

8.5.3 Input Processing For This Example

The following output shows the results of processing the text and diagram input shown in Figure 8.2. In the resulting description of the active conducting loop (l), the fact that the object is a conducting loop, is immobile, that there is an initial counterclockwise current flow, and that there is a variable resistance in the loop is obtained from the Picture Semantics description for the conducting loop. After interpreting the text sentence, we obtain the additional knowledge that the *negative* current flow in the loop is increasing towards zero due to the increasing resistance in the loop.

In the descriptions of two conducting loops given below, the assertions made into the knowledge base based on Picture Semantics descriptions are marked with a dash (-). The assertions made after processing the text sentence are marked with a star (*). The remaining assertions are those made by the Figure Understander as it processed the diagram.

```
****************************************************************
***                   Parsing The Figure                     ***
****************************************************************

Choosing the new frame of reference.

The old coordinate system is being rotated by 0 degrees.

Determining new vertex coordinates for all objects.

Extracting qualitative information about the figure.

Interpreting text sentence:
 (THE CURRENT IN THE LOOP IS INCREASING)

The ATN parse of the sentence:
 `` (THE CURRENT IN THE LOOP IS INCREASING) '' is:

(S (NP (DET THE) (NOUN CURRENT)
       (PP (PREP IN) (NP (DET THE) (NOUN LOOP))))
 (VP (AUX IS) (VERB INCREASING)))

The text description accompanying the figure has been processed.

 The following Algernon path shall be asserted:

 Algy path =

((:THE ?V1 (MEMBER CONDUCTING-LOOPS ?V1)
  (LOOP-CURRENT-FLOW-DIRECTION ?V1 ?DIR P-INIT))
 (LOOP-CURRENT-FLOW-DERIVATIVE ?V1 + P))

Creating a postscript file for the figure after the change in the
frame of reference.

The processing of the input figure has been completed.
NIL
```

```
> (algy)

Algernon 1.2.4 of 16 December 1992.

algy> vf l  (partial display of actual output)
L:
  name:       ((l) nil)
  isa:        (convex-polygons nil) (immobile-objects nil)
              (convex-2-d-objects nil) (solid-objects nil)
              (field-sources nil)
-              (conducting-loops nil)
              (objects nil) (world-objects nil) (things nil)
  is-rectangular: (true nil)
-  is-conducting-object: (true nil)
-  is-solid:   (true nil)
-  mobility-of: (immobile nil)
-  variable-resistance-in-loop: (true nil)
-  directional-change-in-resistance: ((+ p) nil)
-  loop-current-flow-direction: ((ccw p-init) nil)
-  current-flow-source: (permanent nil)
-  exists-current-flow: ((true p-init) nil) ((true p) nil)
  bottommost: ((248 336 416 p-init) nil)
  leftmost:   ((248 416 480 p-init) nil)
  topmost:    ((248 336 480 p-init) nil)
  rightmost:  ((336 416 480 p-init) nil)
  box-bounding-shape: (true nil)
*  loop-current-flow-derivative: ((+ p) nil)

algy> vf l1  (partial display of actual output)
L1:
  name:       ((l1) nil)
  isa:        (convex-polygons nil) (mobile-objects nil)
              (convex-2-d-objects nil) (solid-objects nil)
              (field-sources nil) (conducting-loops nil)
              (objects nil) (world-objects nil) (things nil)
  is-rectangular: (true nil)
-  is-conducting-object: (true nil)
-  is-solid:   (true nil)
-  mobility-of: (mobile nil)
  current-flow-source:
     (non-value) (permanent nil)
  bottommost: ((368 456 499 p-init) nil)
  leftmost:   ((368 499 563 p-init) nil)
  topmost:    ((368 456 563 p-init) nil)
  rightmost:  ((456 499 563 p-init) nil)
  box-bounding-shape: (true nil)
```

8.5.4 The Models Produced by QPC

The following shows a partial list of the QPC model fragments (view descriptions) from our spatial and domain models that are found to be active by QPC in the initial state of this example. Note that the last view takes into account the

```
(defview create-field-from-current-in-loop ()
 :individuals ((loop :type conducting-loops
                     :conditions ((exists-current-flow loop TRUE ?model))))
 :relations ((:forc ?cf-fld
                    (source-of-field ?model ?cf-fld loop)
                    (isa ?cf-fld current-flow-magnetic-fields)
                    (piecewise-continuous-entity ?model ?cf-fld))))

(defview properties-of-field-from-CCW-current-flow ()
 :individuals ((loop :type conducting-loops
                     :conditions
                     ((loop-current-flow-direction loop CCW ?model)))
               (cf-fld :type current-flow-magnetic-fields
                       :conditions
                       ((source-of-field cf-fld loop))))
 :relations
  ((q+ (fld-strength cf-fld) (loop-current-flow loop))
   (not (q> (fld-strength cf-fld) zero)) ; outside the source loop.
   (q<= (fld-strength cf-fld) zero) ; outside the source loop.
   (correspondence (fld-strength cf-fld) zero
                   (loop-current-flow loop) zero)
   (piecewise-continuous-variable ?model (fld-strength cf-fld))

   ; call to Lisp function which determines the direction of the field.
   (:bind ?fld-dir-val
          (determine-CF-mag-fld-dir 'cf-fld 'loop 'CCW '?model))
   ; Store the fact that the field direction differs inside and outside
   ; the conducting loop where the current flow is taking place.
   (:bind ?inside-dir-val (car '?fld-dir-val))
   (:bind ?outside-dir-val (cadr '?fld-dir-val))
   (field-dir-inside-src-loop cf-fld ?inside-dir-val)
   (field-dir-outside-src-loop cf-fld ?outside-dir-val)))
```

Figure 8.3: Examples of views in the magnetic fields domain model which dynamically create a magnetic field object given that there is current flow in a conducting object, and which model the properties of this newly created field. Note that the field object itself is declared as a piecewise-continuous entity, and that its field strength quantity is declared as a piecewise-continuous variable.

fact that the orientation of the loop is not reversed. If the orientation of the loop was reversed, the direction of current flow in the first loop would actually appear to be clockwise, and the model fragments that model the properties of a clockwise current flow in a conducting loop would be invoked instead of those that model the properties of a counterclockwise current flow. The last view also takes into account that the second loop lies outside the conducting loop which generates the magnetic field.

Since our purpose for this example is to illustrate our methods for reasoning about dynamically created spatial objects, in Figure 8.3, we present a view description from our magnetic fields domain model which is invoked to dynamically create a piecewise-continuous magnetic field object given the knowledge that there is current flow in a closed conducting loop. We have also included a view description for the properties of this newly created magnetic field.

```
Instantiating view PERMANENT-CCW-CURRENT-FLOW-IMMOBILE-LOOP.
  L -- ABSTRACT-LOOP5
Instantiating view CREATE-FIELD-FROM-CURRENT-IN-LOOP.
  L -- ABSTRACT-LOOP22
Instantiating view PROPERTIES-OF-FIELD-FROM-CCW-CURRENT-FLOW.
  L.P-INIT.SOURCE-OF-FIELD -- ABSTRACT-CF-FLD1
  L -- ABSTRACT-LOOP7
Instantiating view ABOVE-VIEW.
  L -- ABSTRACT-BOTTOM-OBJ
  L1 -- ABSTRACT-TOP-OBJ
Instantiating view LEFT-OF-VIEW.
  L1 -- ABSTRACT-RIGHT-OBJ
  L -- ABSTRACT-LEFT-OBJ
Instantiating view
FLUX-THROUGH-NON-REVERSED-LOOP-IN-CF-FIELD-OUTSIDE-SOURCE.
  L1 -- ABSTRACT-LOOP9
  L.P-INIT.SOURCE-OF-FIELD -- ABSTRACT-CF-MAG-FIELD
  L -- ABSTRACT-CF-MAG-FIELD-SRC
```

At this point, since there is no direct assertion in the knowledge base about the direction or strength of the dynamically created magnetic field (the equations which provide this information are not evaluated until qualitative simulation takes place), QPC's model creation process branches on the assumption that the direction of the field passing through the second loop could be into the page, out of the page, and that its strength could also be zero. The correct assumption is that the direction of the field is pointed into the page within the second loop. The models created based on false assumptions are later proven to be contradictory by QPC, and are any further simulation of these models is stopped. This is illustrated by the pruned branches in the tree of possible behaviors for this example, as shown in Figure 8.4.

```
QPC has been unable to determine whether 2 views should be instantiated.
  0. (increasing-flux-through-loop-pos-fld l.p-init.source-of-field l1)
    0. (q> (:quote l.p-init.source-of-field.fld-strength) (:quote zero))
  1. (increasing-flux-through-loop-neg-fld l.p-init.source-of-field l1)
    0. (q< (:quote l.p-init.source-of-field.fld-strength) (:quote zero))
Break to examine ambiguous views? n
Branching to create 3 new models due to op-conditions-unknown.
  Created M-1 under initial assumption:
        (region-active (:slot q<)
                       (:quote l.p-init.source-of-field.fld-strength)
                       (:quote zero))
  Created M-2 under initial assumption:
        (region-active (:slot q>)
                       (:quote l.p-init.source-of-field.fld-strength)
                       (:quote zero))
  Created M-3 under initial assumption:
        (region-active (:slot q=)
                       (:quote l.p-init.source-of-field.fld-strength)
                       (:quote zero))
```

The model fragments that are active in model M-1, the branch based on the correct assumption that the magnetic field in the second loop (l1) is pointed into the page, are given below. Model M-1 takes into account that the magnetic flux through the second loop is increasing (negative value is increasing towards zero), and that this results in the induction of a clockwise current flow in the second loop. Once it is asserted into the knowledge base that a current flow exists in the second loop, the models fragments (as in Figure 8.3) which create and model the properties of the magnetic field due to this current flow are invoked. The model fragments which describe the magnetic forces that act on current carrying objects placed in a magnetic field are also invoked. In this example, the magnetic force vectors which act on each edge of the second loop cancel each other, and there is no net torque. Therefore, there is no motion.

```
Instantiated views for M-1:
 0. (properties-of-field-from-ccw-current-flow l l.p-init.source-of-field)
 1. (above-view l1 l)
```

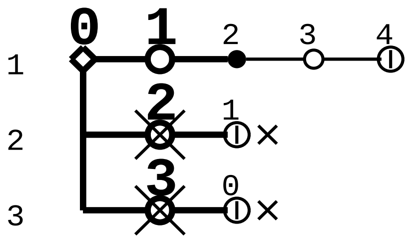


Figure 8.4: Tree of all behaviors produced for the problem given in Figure 8.2.

```
2. (left-of-view l l1)
3. (no-rotation l1)
4. (flux-through-non-reversed-loop-in-cf-field-outside-source
    l l.p-init.source-of-field l1)
5. (increasing-flux-through-loop-neg-fld l.p-init.source-of-field l1)
6. (cw-induced-emf l1)
7. (create-field-from-current-in-loop l1)
8. (properties-of-field-from-cw-current-flow l1 l1.m-1.source-of-field)
9. (setup-magnetic-force-on-loop-in-field-xy-plane
    l1 l.p-init.source-of-field)
10. (zero-x-dir-mag-force-on-loop l1)
11. (zero-y-dir-mag-force-on-loop l1)
12. (zero-z-dir-mag-force-on-loop l1)
13. (force-on-loop l1)
14. (resolve-induced-cw-current-flow-in-loop-1
     l.p-init.source-of-field l1)
15. (constraints-on-mobile-objects l1)
16. (zero-theta-z l1)
17. (permanent-ccw-current-flow-immobile-loop l)
18. (create-field-from-current-in-loop l)
```

8.5.5 The Qualitative Behaviors For This Example

The tree of behaviors produced for this example is shown in Figure 8.4. The only qualitative behavior for this example, produced by the simulation of model M-1, is shown in Figure 8.5. In the figure, the variables l.p-init.source-of-field.fld-strength and l1.p-init.source-of-field.fld-strength model the strength of the magnetic fields dynamically created by the current flows in the conducting loops l and l1.

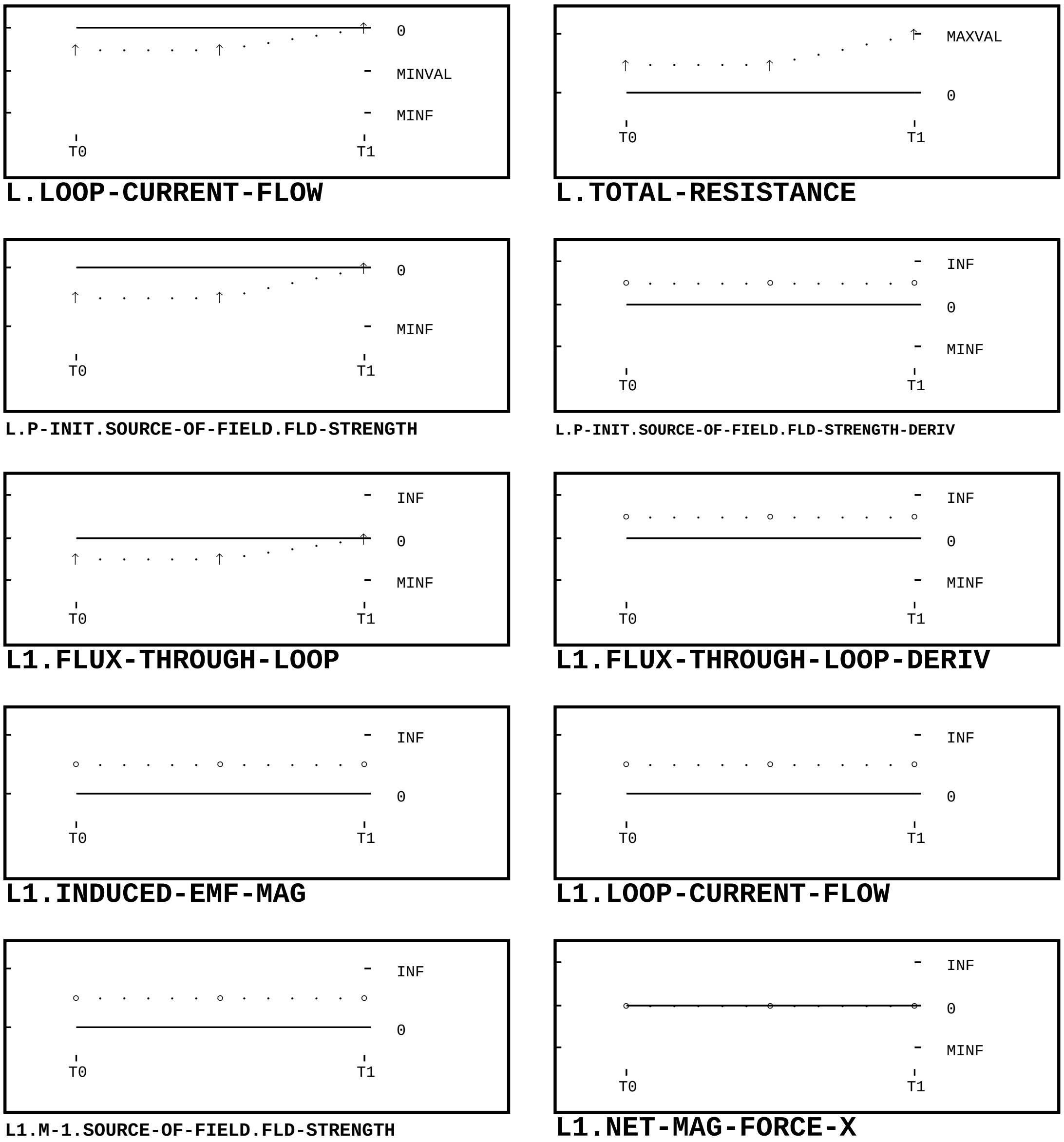


Figure 8.5: The only qualitative behavior for the problem given in Figure 8.2.

8.5.6 Final Knowledge Base Descriptions for the Objects

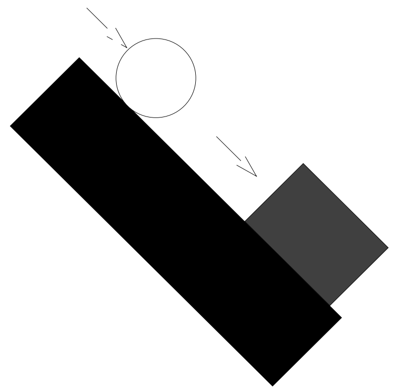
The following shows partial knowledge base descriptions for the dynamically created magnetic fields in the initial state (models P-init and M-1) due to the current flow in loops l and l1. We also show the most relevant assertions made to the knowledge base description of loop l1, the initially passive conducting loop, during the qualitative simulation of models P-init and M-1.

```
algy> vf l1
L1: [slot]
  name:        ((l1) nil)
  in-xy-plane: ((true m-1) nil) ((true p-init) nil)
  orientation-type: ((not-reversed m-1) nil) ((not-reversed p-init) nil)
  loop-current-flow-direction: ((cw m-1) nil)
  exists-current-flow: ((true m-1) nil)
  induced-emf-dir: ((cw m-1) nil)
  net-mag-force-vector: ((m-1 (0 0 0)) nil)

algy> vf l.p-init.source-of-field  (magnetic field from current flow in l)
L.p-init.source-of-field: [slot]
  isa:         (l.p-init.source-of-field-selfset nil) (slots nil)
               (current-flow-magnetic-fields nil) (permeable-objects nil)
               (things nil) (objects nil) (world-objects nil)
               (immobile-objects nil) (magnetic-fields nil) (roles nil)
  in-xy-plane: ((true m-1) nil) ((true p-init) nil)

algy> vf l1.m-1.source-of-field (magnetic field from current flow in l1)
L1.m-1.source-of-field:
  isa:         (l1.m-1.source-of-field-selfset nil)
               (current-flow-magnetic-fields nil) (permeable-objects nil)
               (things nil) (objects nil) (world-objects nil)
               (immobile-objects nil) (magnetic-fields nil)
  in-xy-plane: ((true m-1) nil)
```

8.6 The Loop Moving Down the Track Example

This section presents the solution to the problem of describing what happens as a loop moves down a track through as magnetic field, as discussed earlier in Chapters 1 and 5. The text and diagram input to the Figure Understander for this problem is shown in Figure 8.6. The Figure Understander automatically chooses a new frame of reference for this problem to align the maximal number of objects in the diagram with the coordinate axes. The rotated diagram, shown in Figure 8.7, is used for all subsequent spatial reasoning.

- **Diagram:**



- **Text:** The loop is moving down the track.

Figure 8.6: A magnetic fields problem which asks the reader to describe what happens as the conducting loop moves down the track and through the magnetic field.

8.6.1 Problem Description

In this problem, considering the rotated diagram, the conducting loop is initially to the left of the magnetic field, and is moving towards the magnetic field. It is accelerating due to the action of the external force vector shown in the diagram. The next qualitatively interesting state occurs when the loop begins to enter the magnetic field. This causes the magnitude of the magnetic flux passing through the loop to increase, resulting in an induced electromotive force (emf) in the loop which

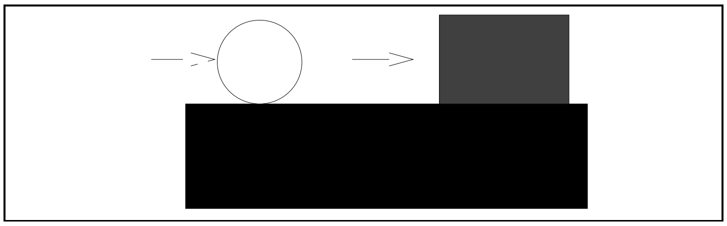


Figure 8.7: Rotated diagram produced by the Figure Understander for the problem described in Figure 8.6.

in turn causes a counterclockwise current flow. A magnetic force, whose magnitude depends on the strength of the field and the strength of the generated current flow, then acts on the right side of the loop, opposing the motion of the loop into the field.

The next qualitatively interesting state occurs when the loop is fully enclosed in the field. Because the flux through the loop no longer changes, no dynamic processes, other than motion, are active until the loop reaches the right boundary of the field, and begins to exit the field. At this point, the area of overlap begins to decrease, and therefore the magnitude of the flux through the loop decreases. An induced emf is established which causes a clockwise current flow in the loop. A resultant magnetic force, opposite to the direction of motion, now acts on the left side of the loop, which slows the acceleration of the loop as it moves out of the field.

An additional effect is that a new magnetic field is created whenever there is current flow in the loop. This magnetic field serves to oppose the change in flux through the loop. Thus, while the loop is entering the permanent field given in Figure 8.6, the new magnetic field generated by the current flow in the loop will have a direction pointed out of the page within the loop, and pointed into the page outside the loop. This new magnetic field is not permanent, and ceases to exist once the loop is fully enclosed in the permanent magnetic field and the current flow in the loop stops.

When the loop is exiting the permanent magnetic field, and a clockwise current flow is established in the loop, a new magnetic field is again created, with its direction pointed into the page inside the loop, and out of the page outside the loop.

8.6.2 What Does This Example Illustrate?

This example is of interest because it illustrates our ability to reason about translational motion and the forces that can affect translational motion. It tests our models for reasoning about the relative positions of objects, including detecting changes in the area of overlap between objects. It also tests our model of the orientation of objects, since we have to determine the orientation of the parts of the conducting loop that are within the magnetic field to determine where the net magnetic force will act on the loop. The resultant direction of the net magnetic force is determined by applying our methods for performing the qualitative vector cross product operation.

More specifically, in this example, as the loop enters the field, it overlaps only the left edge of the field. This, and the fact that the loop is circular in shape, is used to infer that the net magnetic force will act at the rightmost point of the loop as it enters the field. Since the direction of current flow at this point is +Y, and since the direction of the magnetic field at this point is -Z, the cross product operation $\mathbf{F_m} = i\mathbf{l} \times \mathbf{B}$ is applied to determine that the direction of the net magnetic force on the loop is -X.

This example also contrasts our method of reasoning about the orientation of an object, which uses knowledge of actual surfaces of an object, and our method of reasoning about the relative positions of objects, which makes use of simplying shape abstractions to approximate the spatial extent of an object. We just discussed how the fact that the loop in circular in shape is used to determine where the net magnetic force acts on the loop. In contrast, since the spatial extent of the circular loop is approximated by a rectangular bounding box, the area of overlap between the loop and the magnetic field is modeled simply as a monotonically increasing function of the velocity of the loop.

Further, as the loop enters and exits the magnetic field, the value of the derivative of the area of overlap between the bounding box for the loop and that of the field changes discontinuously from a non-zero value to zero. This feature of the problem tests our methods for modeling and reasoning about piecewise-continuous variables. Finally, as the loop enters and exits the field, and a current flow is induced in the loop, magnetic fields are dynamically created. This further tests out ability to model piecewise-continuous entities.

8.6.3 Input Processing For This Example

The following output shows the results of processing the text and diagram input shown in Figure 8.6. Note that the Figure Understander numerically rotates the diagram counterclockwise by an angle of 45 degrees to obtain the final diagram to be used as the initial spatial scene. As shown in Figure 8.7, since the vectors that provide the direction of the velocity and the net force on the conducting loop are also rotated, the relative motion of the loop through the field is not altered, and the magnetic fields processes that are actived remain the same. The spatial reasoning process is simplified since there is only a need to reason about motion along the X-axis instead of both the X and Y axis.

The fact that the moving object is a conducting loop and is mobile is obtained from the Picture Semantics description for this diagram, along with the knowledge that the magnetic field has a direction pointed into the page, that the track is a support object, and that there are velocity and force vectors in the diagram. These assertions are marked with a dash (-) in the knowledge base descriptions of the objects in the trace output shown below.

After interpreting the text sentence, we obtain the additional knowledge that the velocity and force vectors are associated with the conducting loop. These assertions are marked with a star (*) in the display of the knowledge base description of the conducting loop (l). Note that the fact that the conducting loop and track are touching has also been proven by the Figure Understander. The Figure Understander uses the rule-based, static-world implementation of our spatial model for proving spatial relations about the initial scene.

Incidentally, we currently do not consider the spatial reasoning problems requiring reasoning about mechanical constraints between objects. Therefore, in the scenario (initial world) description that is generated for use by QPC, all information about support objects (the track) is automatically ignored. This step is taken to improve the efficiency of the Motion Understander, since otherwise, spatial relations between the other objects and the track will be maintained by QPC.

However, we do provide some support to allow the ability to reason about mechanical constraint issues to be added in future releases of the Motion Understander. For example, when processing a diagram, we automatically extract much of the information used by Nielsen [61, 62] in his work on reasoning about mechanical constraint problems. In the knowledge base description of the conducting loop (l), the assertions about *no-torque-points* model where a perpendicular force acting on a given surface of the loop must act so that its line of action will cross the center of gravity of the loop. Nielsen uses this information to determine when a given force acting on an object can result in a torque on the object.

```
******************************************************************
***                    Parsing The Figure                      ***
******************************************************************

Choosing the new frame of reference.

The old coordinate system is being rotated by 45 degrees.

Determining new vertex coordinates for all objects.

Extracting qualitative information about the figure.

Interpreting text sentence:
 (THE LOOP IS MOVING DOWN THE TRACK)

The ATN parse of the sentence:
 `` (THE LOOP IS MOVING DOWN THE TRACK) '' is:

(S (NP (DET THE) (NOUN LOOP))
 (V (AUX IS) (VERB MOVING)
  (CLAUSE (AP (ADV DOWN) (NP (DET THE) (NOUN TRACK))))))

The text description accompanying the figure has been processed.

 The following Algernon path shall be asserted:

 Algy path =

((:THE ?V1 (MEMBER CONDUCTING-LOOPS ?V1) (MOBILITY-OF ?V1 MOBILE))
 (:THE ?V2 (MEMBER SUPPORT-OBJECTS ?V2) (TOUCHING ?V1 ?V2 P-INIT))
 (:THE ?V3 (MEMBER VELOCITY-VECTORS ?V3)
          (VECTOR-DIRECTION-OF ?V3 P ?VEL-VECT))
 (:THE ?V4 (MEMBER FORCE-VECTORS ?V4)
          (VECTOR-DIRECTION-OF ?V4 P ?FORCE-VECT))
 (VELOCITY-VECTOR ?V1 P-INIT ?VEL-VECT)
 (PERM-FORCE-VECTOR ?V1 P ?FORCE-VECT))
```

```
Creating a postscript file for the figure after the change
in the frame of reference.

The processing of the input figure has been completed.

> (algy)
Algernon 1.2.4 of 16 December 1992.
algy> vf l
L:
  name:       ((l) nil)
  isa:        (circular-objects nil)
              (mobile-objects nil) (solid-objects nil)
-              (conducting-loops nil)
              (convex-2-d-objects nil) (field-sources nil)
              (objects nil) (world-objects nil) (things nil)
  is-rectangular: (false nil)
-  is-conducting-object: (true nil)
-  is-solid:   (true nil)
-  mobility-of: (mobile nil)
  current-flow-source:
     (non-value) (permanent nil)
  no-torque-point: ((f-e-4 edge-center) nil) ((f-e-3 edge-center) nil)
                 ((f-e-2 edge-center) nil) ((f-e-1 edge-center) nil)
*  touching:   ((o p-init) nil)
*  velocity-vector: ((p-init (1 0 0)) nil)
*  perm-force-vector: ((p (1 0 0)) nil)

algy> vf f    (magnetic field)
F:
  name:       ((f) nil)
  isa:        (convex-polygons nil) (convex-2-d-objects nil)
              (permeable-objects nil) (immobile-objects nil)
-              (magnetic-fields nil) (steady-uniform-magnetic-fields nil)
              (objects nil) (world-objects nil) (things nil)
-  mobility-of: (immobile nil)
-  is-conducting-object: (false nil)
-  field-type-of: (uniform nil) (steady nil)
-  is-solid:   (false nil)
-  field-direction-of: ((0 0 -1) nil)
  edge-of:    (f-e-4 nil) (f-e-3 nil) (f-e-2 nil) (f-e-1 nil)
  vertex-of:  (f-v-4 nil) (f-v-3 nil) (f-v-2 nil) (f-v-1 nil)
  perimeter-of: ((f-v-1 f-e-1 f-v-2 f-e-2 f-v-3 f-e-3 f-v-4 f-e-4) nil)
  connected-edges: ((f-e-3 f-v-3 f-e-4) nil) ((f-e-2 f-v-3 f-e-3) nil)
                 ((f-e-1 f-v-2 f-e-2) nil) ((f-e-4 f-v-1 f-e-1) nil)
  connected-vertices: ((f-v-4 f-e-4 f-v-1) nil) ((f-v-3 f-e-3 f-v-4) nil)
                    ((f-v-2 f-e-2 f-v-3) nil) ((f-v-1 f-e-1 f-v-2) nil)
  has-bottommost-edge: ((true p-init) nil)
  bottommost: ((359 408 418 p-init) nil)
  bottommost-edge: ((f-e-4 p-init) nil)
  bottommost-vertex: ((f-v-1 p-init) nil) ((f-v-4 p-init) nil)
  has-leftmost-edge: ((true p-init) nil)
  leftmost:   ((359 418 452 p-init) nil)
  has-topmost-edge: ((true p-init) nil)
  topmost:    ((359 408 452 p-init) nil)
  has-rightmost-edge: ((true p-init) nil)
```

```
  rightmost:  ((408 418 452 p-init) nil)
  box-bounding-shape: (true nil)

algy> vf o    (track)
O:
  name:       ((o) nil)
  isa:        (convex-polygons nil) (convex-2-d-objects nil) (things nil)
              (immobile-objects nil) (solid-objects nil)
-              (support-objects nil)
              (objects nil) (world-objects nil)
  is-rectangular: (true nil)
  mobility-of: (immobile nil)
-  is-conducting-object: (false nil)
-  is-solid:   (true nil)
  box-bounding-shape: (true nil)

algy> vf mv   (velocity vector in the diagram)
Mv:
  name:       ((mv) nil)
  isa:        (line-objects nil) (vectors nil) (velocities nil)
-              (velocity-vectors nil)
              (things nil) (objects nil) (world-objects nil)
  vertex-of:  (mv-v-2 nil) (mv-v-1 nil)
  transformed-figure-slope: (0 nil)
  vector-direction-of: ((p (1 0 0)) nil)

algy> vf fv   (force vector in the diagram)
Fv:
  name:       ((fv) nil)
  isa:        (line-objects nil) (vectors nil)
-              (force-vectors nil)
              (objects nil) (world-objects nil) (things nil)
  vertex-of:  (fv-v-2 nil) (fv-v-1 nil)
  transformed-figure-slope: (0 nil)
  vector-direction-of: ((p (1 0 0)) nil)
```

8.6.4 The Initial Model Produced by QPC

The following shows the QPC model fragments (view descriptions) from our spatial model that are found to be active by QPC in the initial state of this example. Here, no magnetic fields processes are active since the loop and the field are disconnected. We also show the trace output produced by QPC to illustrate how it assembles a model for qualitative simulation after determining which model fragments are to be instantiated. In this example, the qualitative simulation of the initial model by QSIM results in one qualitative behavior. A transition occurs after a qualitatively significant change in the spatial state - when the loop reaches the field.

```
Model P-INIT is the initial model.
```

```
Initializing model P-INIT.
Instantiated views for P-INIT:
    0. (force-on-loop l)
    1. (loop-left-of-perm-field l f)
    2. (left-of-view l f)
    3. (aligned-x-view f l)
    4. (aligned-x-view l f)
    5. (no-rotation l)
    6. (constraints-on-mobile-objects l)
    7. (zero-theta-z l)
Asserting that the world is closed.
Resolving influences.
Inferring that all uninfluenced vars are constant.
Resolving correspondences.
Resolving quantity spaces.
  The quantity spaces are totally ordered.
  Narrowing the qspaces.
  Asserting shared quantity spaces.
Collecting region transitions.
Defining a QDE.
Constructing the QSIM QDE.
Building an initial state.
Run time: 0.060 seconds to initialize S-0.
  Built initial state S-0, which is complete.
Simulating from initial state S-0.
Variables in P-INIT likely to chatter are NONE.
Run time: 0.290 seconds to simulate 4 states.
  Generated 1 behavior.
  Created model #<Model: M-1>, a transition from state S-2.
```

8.6.5 The Second Model Produced by QPC

The next model constructed by QPC is for reasoning about what happens as the loop enters the field. The information about the world state after the qualitative simulation of the previous model (P-INIT) is first copied into the initial world description of the new model (model M-1). However, as shown by the trace below, the values of any piecewise-continuous variables and entities are ignored. In our magnetic fields domain model, the net force on an object is a function of any permanent forces acting on the object and any magnetic forces acting on the object. The net force is modeled as a piecewise-continuous variable since a magnetic force may exist in one qualitative state, but not in another. The following trace shows that the value of the net force on the loop is not inherited.

Without this feature, the value of the net force on the loop would be inherited into model M-1, and since there is non-zero magnetic force in model M-1, there would be a contradiction regarding the value of the net force of the loop. This illustrates the usefulness of our methods for modeling piecewise-continuous variables in QPC.

```
Model M-1 follows P-INIT because of a transition.
Initializing model M-1.
  Model M-1 is a transition from P-INIT via S-2.
  Inheriting model assumptions:
      "[none]"
  Inheriting initial conditions assumptions:
      (region-active (:slot q<) (:quote f.lm-xval) (:quote l.rm-xval))
      (region-active (:slot q>) (:quote l.rm-xval) (:quote f.lm-xval))
      (region-active (:slot q>) (:quote f.rm-xval) (:quote l.lm-xval))
      (region-active (:slot q<) (:quote l.lm-xval) (:quote f.rm-xval))
  Assert the qspace orderings from S-2.
  Handle non-inheritable variables and circular q-spaces.
  Assert the correspondences from S-2.
  Inheriting values from state S-2:
       (l.perm-force-x ((0 inf) nil))
       (l.perm-force-y (0 nil))
       (l.perm-force-z (0 nil))
       (l.net-force-x ((0 inf) nil))
       (l.net-mag-force-x (0 nil))
       (l.net-force-y (0 nil))
       (l.net-mag-force-y (0 nil))
       (l.net-force-z (0 nil))
       (l.net-mag-force-z (0 nil))
ignoring value for piecewise-continuous variable L.NET-FORCE-X
ignoring value for piecewise-continuous variable L.NET-MAG-FORCE-X
ignoring value for piecewise-continuous variable L.NET-FORCE-Y
ignoring value for piecewise-continuous variable L.NET-MAG-FORCE-Y
ignoring value for piecewise-continuous variable L.NET-FORCE-Z
ignoring value for piecewise-continuous variable L.NET-MAG-FORCE-Z
```

At copying the allowed set of values from model P-init, QPC determines the model fragments that are active in the model M-1, which represents the spatial state where the loop is entering the field. In model M-1, there is ambiguity regarding whether the flux through the loop is increasing, decreasing, or steady. Thus, QPC creates three models to explore each of the possibilities. In this example, since the field direction is into the page (negative), the flux through the loop is negative. The net flux through the loop therefore decreases (moves away from zero) as the loop enters the field. This case is covered by the branch covered by model M-3.

As the loop enters the field, a counterclockwise current flow is induced in the loop, and a magnetic force acts on the loop to oppose its entry into the field. The net force (F_n) on the loop is given by $F_n = F_p - F_m$, where F_p is the permanent force on the loop as given in the diagram, and F_m is the magnetic force. Then, since $F_n = m \times a_l$, where a_l is the acceleration of the loop, and since $dv_l/dt = a_l$, where v_l is the velocity of the loop, we have that dv_l/dt is a monotonically increasing function of F_n. However, as shown in greater detail in Appendix A, we also have that F_m is a monotonically increasing function of v_l.

Without quantitative knowledge of the magnitude of the magnetic force, there are three possible initializations for the values of these variables during the qualitative simulation - that they are equal, that F_p is larger, and that F_m is larger. This results in seven qualitative behaviors, as shown by the output trace below and the corresponding branches in the behavior tree shown in Figure 8.8. In the case where $F_p = F_m$, the velocity of the loop remains steady as it moves through the field. In the case where F_p is larger, the loop continues to accelerate, but F_m increases as well. Then, the difference between F_p and F_m moves toward zero. There are three behaviors associated with this branch: (a) that the loop moves completely into the field before F_n reaches zero; (b) that it moves completely into the field just as F_n reaches zero; and (c) F_n reaches zero, making the velocity constant, before the loop is enclosed in the field.

```
Instantiated views for M-3:
  0. (force-on-loop l)
  1. (aligned-x-view l f)
  2. (aligned-x-view f l)
  3. (aligned-y-view l f)
  4. (bb-overlap-view l f)
  5. (aligned-y-view f l)
  6. (no-rotation l)
  7. (constraints-on-mobile-objects l)
  8. (zero-theta-z l)
  9. (overlap-bb-left-view l f)
 10. (flux-through-loop-in-su-perm-neg-field f l)
 11. (decreasing-flux-through-loop-neg-fld f l)
 12. (ccw-induced-emf l)
 13. (create-field-from-current-in-loop l)
 14. (properties-of-field-from-ccw-current-flow l l.m-3.source-of-field)
 15. (setup-magnetic-force-on-loop-in-field-xy-plane l f)
 16. (neg-x-dir-mag-force-on-loop l)
 17. (zero-y-dir-mag-force-on-loop l)
 18. (zero-z-dir-mag-force-on-loop l)
 19. (mag-force-magnitude-2 l f)
 20. (resolve-induced-ccw-current-flow-in-loop-2 f l)
```

```
Asserting that the world is closed.
Resolving influences.
Inferring that all uninfluenced vars are constant.
Resolving correspondences.
Resolving quantity spaces.
Collecting region transitions.
Defining a QDE.
Constructing the QSIM QDE.
Building an initial state.
Run time: 0.150 seconds to initialize (S-5 S-6 S-7).
  Built initial state S-4, which has at most 3 consistent completions.
Simulating from initial state S-4.
Run time: 2.430 seconds to simulate 15 states.
  Generated 7 behaviors.
Display behaviors generated for the model? n
  Created model #<Model: M-5>, a transition from state S-15.
  Created model #<Model: M-6>, a transition from state S-17.
  Created model #<Model: M-7>, a transition from state S-28.
```

The three branches associated with the case where $F_m > F_p$ are similar. Here, since F_n is negative, the velocity of the loop decreases, and therefore F_m also decreases. Thus, the difference term $F_p - F_m$ once again moves towards zero. The three behaviors associated with this branch also consider whether the loop becomes fully enclosed in the field before F_n becomes zero, as it becomes zero, or after it becomes zero.

For all the behaviors where F_n become zero, the qualitative state is the same once the loop becomes fully enclosed in the field. Therefore, QSIM combines the behaviors so that only three transitions result, as shown by the last lines in the output trace above. These transitions cover the cases where F_n is one of positive, zero, or negative once the loop is fully enclosed in the field.

8.6.6 The Remaining Models Created by QPC

The remaining models created the QPC further simulate each of the three branches produced by the qualitative simulation of model M-3. The behaviors in all three branches are essentially the same. Since the values of F_m and F_n are not inherited into any subsequent models, and since after the loop is enclosed in the field, no magnetic fields processes are active, we have $F_n = F_p$ in the subsequent model for all three branches.

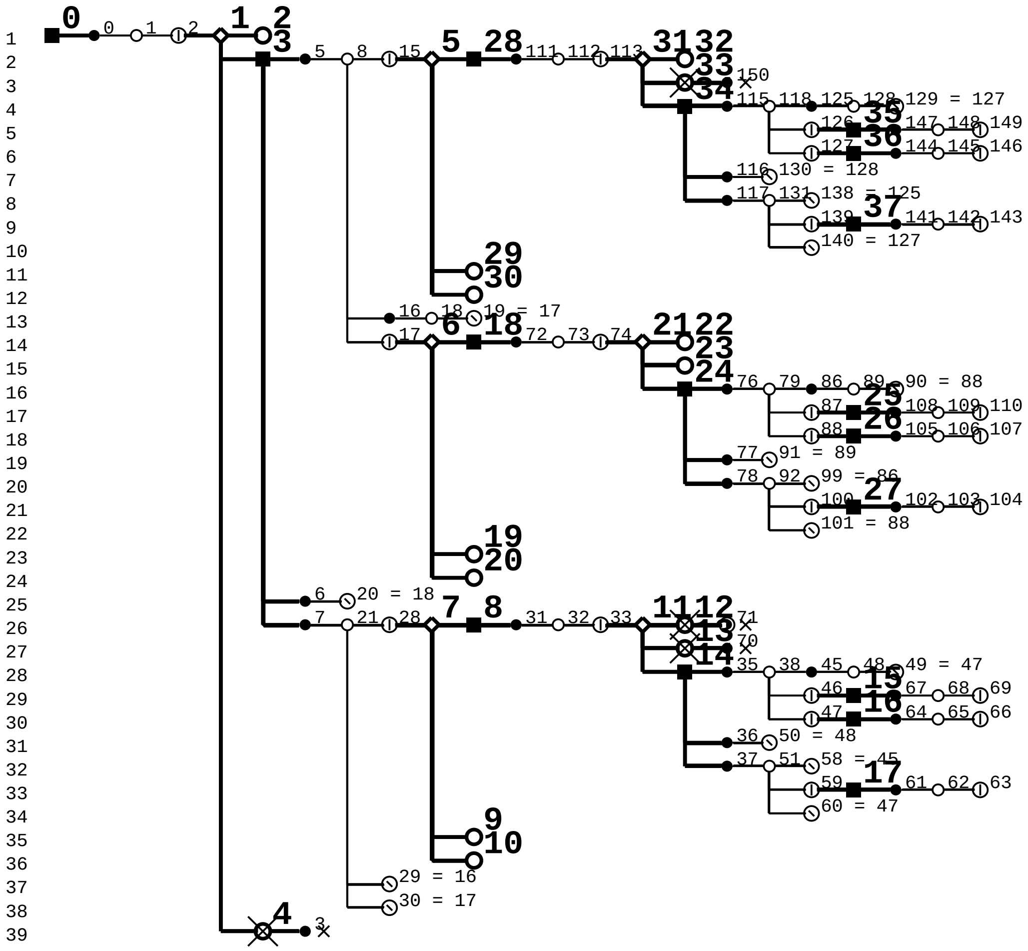


Figure 8.8: Tree of all behaviors produced for the problem given in Figure 8.6.

Then, when the loop exits the field, the situation is similar to the case when the loop entered the field. The only differences are those caused by the fact that the magnitude of the flux passing through the loop now decreases. This, for example, results in a clockwise current flow in the loop. Still, there are once again seven behaviors. Finally, once the loop exits the field, motion in the X-direction continues until the loop reaches the world boundary denoted by $X = Maxval$.

8.6.7 The Qualitative Behaviors For This Example

The tree of behaviors produced for this example is shown in Figure 8.8. A complete simulation of this example was not possible due to memory limitations. However, it has been shown that all branches that are marked as uninvestigated (plain circle at the end of a branch in the tree) in fact result in a contradiction, similar to the other pruned (X) branches in the tree.

The total number of qualitative behaviors produced for this example is too large to display them all. Instead, we show the results of pursuing the assumption that $F_p > F_m$ as the loop enters the field. These behaviors are shown in Figures 8.9, 8.10, and 8.11. The three behaviors result from branching in the qualitative simulation as the loop exits the field. Figure 8.9 shows the case where $F_p - F_m$ remains positive as the loop exits the field. Figure 8.10 shows the case where $F_p - F_m$ remains negative as the loop exits the field. Finally, Figure 8.11 covers all of the other five possible cases where $F_p - F_m$ becomes zero before the loop exits the field.

In the behavior plots, the variable (l.loop-current-flow) shows the direction of current flow induced in the loop. A positive value indicates the current flow is clockwise. Also note the relationships between the permanent net gravitational force on the loop (l.perm-force-x), the net magnetic force on the loop (l.net-mag-force-x), and the net force on the loop (l.net-force-x). The value of the the net force affects whether the velocity of the loop is increasing, decreasing, or constant, as shown by the plots for the variable `l.x-velocity`.

Further, note the discontinuous changes in the derivative of the magnetic

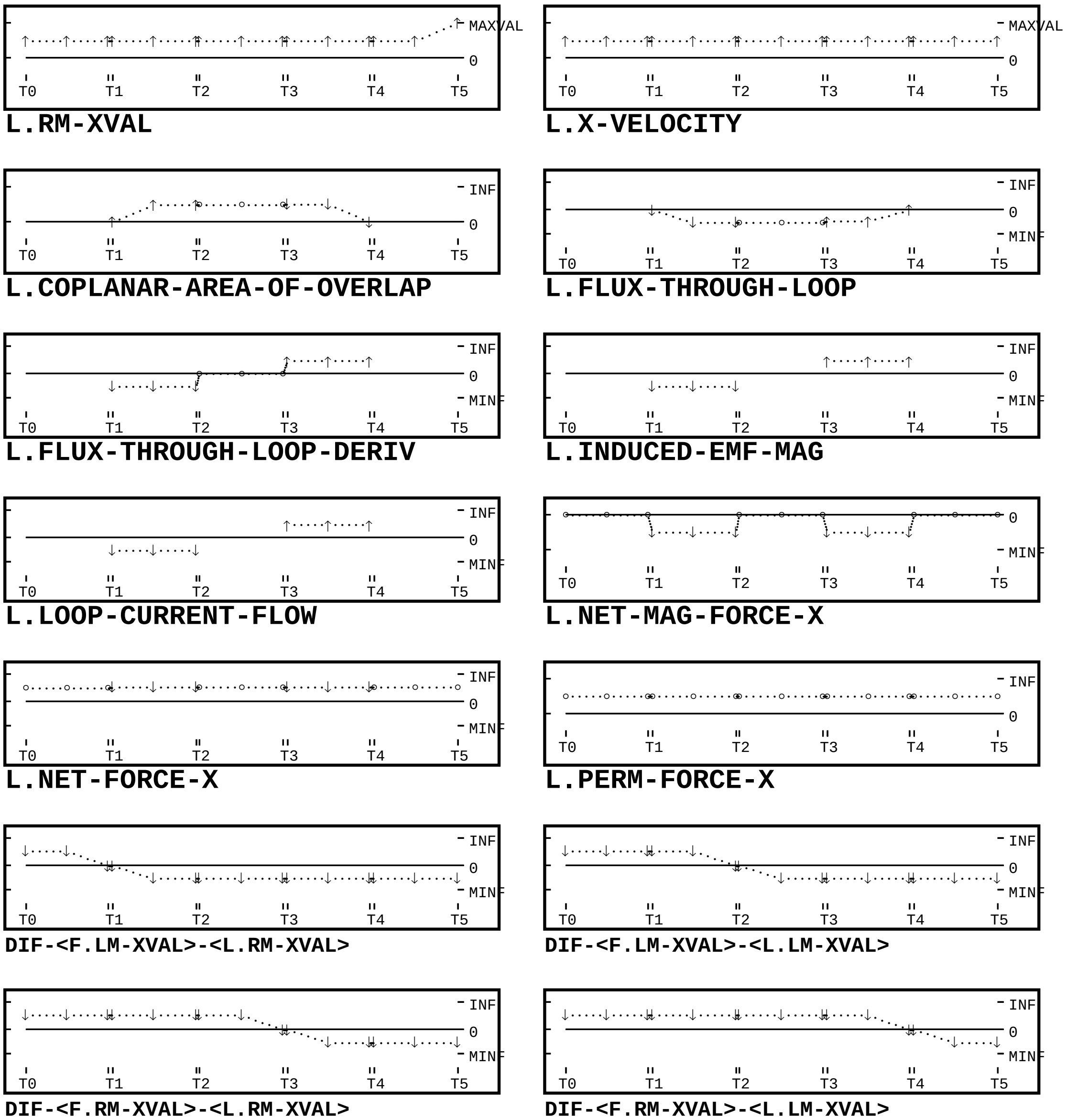


Figure 8.9: A qualitative behavior for the problem given in Figure 8.6 showing the case where $F_p > F_m$ both as the loop enters the field and when the loop exits the field.

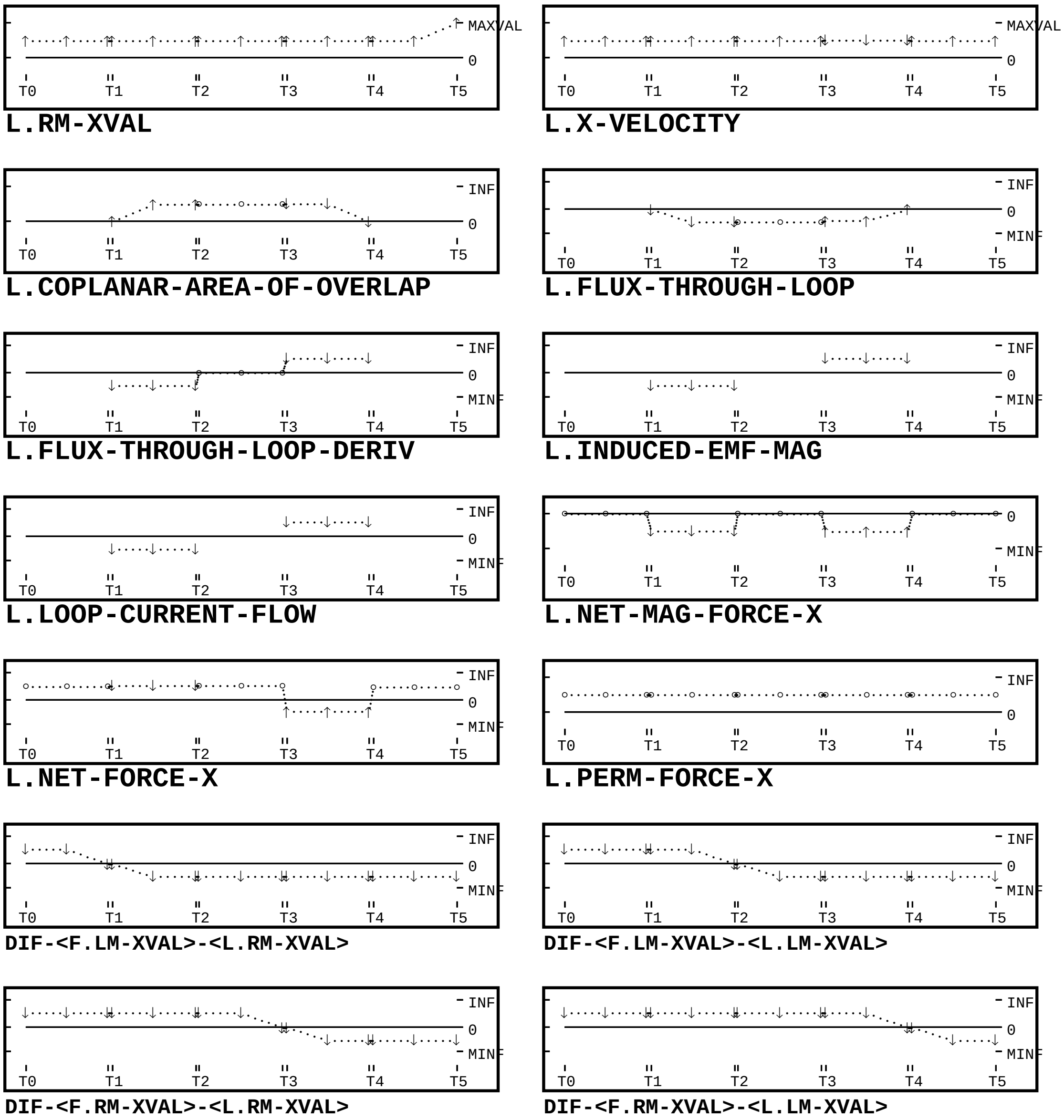


Figure 8.10: A qualitative behavior for the problem given in Figure 8.6 showing the case where $F_p > F_m$ as the loop enters the field but where $F_p < F_m$ as the loop exits the field.

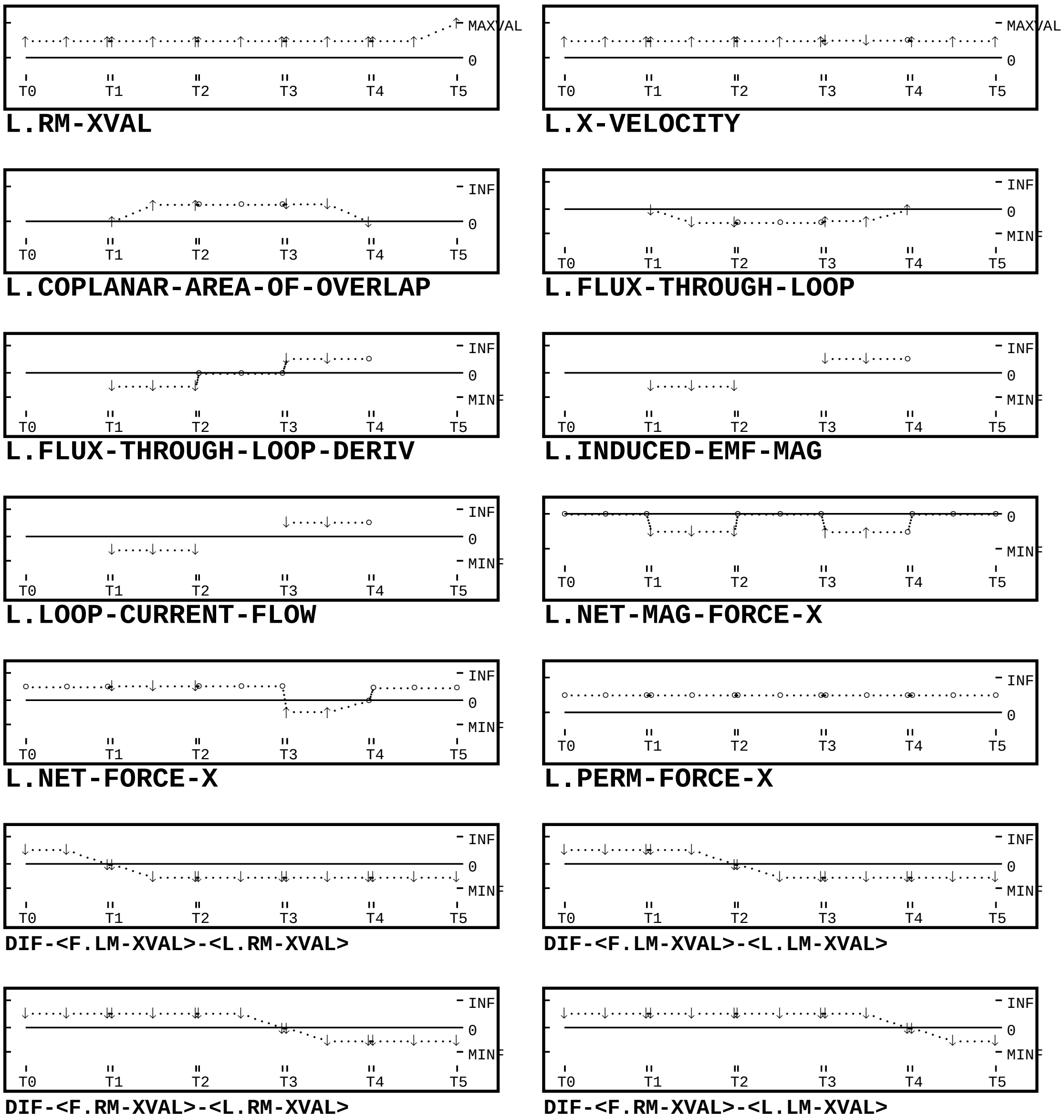


Figure 8.11: A qualitative behavior for the problem given in Figure 8.6 showing a single combined behavior for all case where $F_p > F_m$ as the loop enters the field but where $F_n = F_p - F_m$ becomes zero before or as the loop exits the field.

flux through the loop (l.flux-through-loop-derivative) and the net magnetic force (l.net-force-x). This illustrates the usefulness of our ability to model piecewise-continuous variables. If the values of the above variables were always inherited into the new model after a transition, a contradiction would result at the points where discontinuous changes occur in the behavior plots.

Finally, note the four variables shown at the bottom of each behavior plot. These model the inequality relations between the X-coordinate values of the rightmost and leftmost points of the bounding boxes of the loop and the magnetic field. These variable illustrate how our model of spatial relations between objects, and our model of translation motion, is implemented and used by QPC.

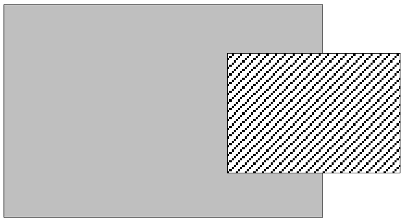
QPC automatically creates a variable to represent the difference between variables that are involved in an inequality constraint. A transition occurs whenever the value of any of these 'DIFference' variables becomes zero. For example, the bottom right box models the value of the difference between $rm(f)$ and $lm(l)$. These two values are equal precisely when the leftmost edge of the loop is touching the rightmost edge of the field.

8.6.8 Final Knowledge Base Descriptions for the Objects

The following shows selected assertions in the knowledge base about the conducting loop after the qualitative simulation of the problem is complete. All the models where the magnetic force vector is shown as (1 0 0) represent models created by QPC based on incorrect assumptions about the direction of change in the magnetic flux passing through the loop. These models are proven to be inconsistent after qualitative simulation.

```
algy> vf l
L: [slot]
 name:       ((l) nil)
 exists-current-flow: ((true m-33) nil) ((true m-34) nil) ((true m-24) nil)
                      ((true m-13) nil) ((true m-14) nil) ((true m-3) nil)
                      ((true m-4) nil)
 loop-current-flow-direction: ((ccw m-33) nil) ((cw m-34) nil)
                              ((cw m-24) nil)  ((ccw m-13) nil)
                              ((cw m-14) nil) ((ccw m-3) nil)
```

- **Diagram:**



- **Text:** None Required!

Figure 8.12: A magnetic fields problem which illustrates the basic principles of operation of an oscillator.

```
                              ((cw m-4) nil)
 induced-emf-dir: ((ccw m-33) nil) ((cw m-34) nil) ((cw m-24) nil)
                  ((ccw m-13) nil) ((cw m-14) nil) ((ccw m-3) nil)
                  ((cw m-4) nil)
 net-mag-force-vector: ((m-33 (1 0 0)) nil) ((m-34 (-1 0 0)) nil)
                       ((m-24 (-1 0 0)) nil) ((m-13 (1 0 0)) nil)
                       ((m-14 (-1 0 0)) nil) ((m-3 (-1 0 0)) nil)
                       ((m-4 (1 0 0)) nil)
```

8.7 An Oscillator Example

This section presents an example which illustrates the basic principles of operation of an oscillator. The input we gave to the Figure Understander for this problem is shown in Figure 8.12. These is no additional text for this problem. There is enough domain-specific information in the diagram itself to completely describe the initial problem state.

8.7.1 Problem Description

The diagram shows a conducting loop with a permanent counterclockwise current flow that is partially overlapped with a magnetic field that is pointed out of the page. With this configuration, a net magnetic force given by $\mathbf{F_m} = i\mathbf{l} \times \mathbf{B}$ will act at the center of the left edge of the rectangular conducting loop. Here, the direction of $\mathbf{l}$ is the direction of current flow in the left edge of the loop. The vector $\mathbf{B}$ represents the magnetic field, and therefore, its direction is that of the magnetic field, which is (0 0 1), or +Z. Since the direction of current flow in the left edge of the loop is -Y, or (0 -1 0), the direction of $\mathbf{F_m}$ is (-Y $\times$ +Z) which is -X. Since there are no other forces acting on the loop, it will begin to move into the field.

Once it is fully enclosed in the field, the magnetic forces acting on the four edges of the loop will cancel each other, and the loop will move at a constant velocity until it reaches the left edge of the field. As the loop begins to exit the field, a net magnetic force will act on the right edge of the loop. This time, F_m is positive, and will act to slow the motion of the loop. In this problem, we do not model the total energy in the loop. Therefore, there are three possible qualitative behaviors: (a) the loop exits the field and moves to the left edge of the world (X = 0); (b) the loop stops just as it exits the field; and (c) the loop stops before it exits the field. If conservation of energy is taken into account, only behavior (c) is possible for this example.

In any event, for case (c), once the loop comes to a complete stop, it begins to move back into the field since the net force is still in the positive X direction. The loop then becomes enclosed in the field, and moves to the right edge of the field with constant velocity. Again, without a model of the energy in the loop, three qualitative behaviors are possible: (a) the loop exits the field and moves to the right edge of the world (X = Maxval); (b) the loop comes to a stop just as it exits the field; and (c) the loop stops before it exits the field. For case (c), the qualitative description of the world state is identical to the initial state, and the process repeats, resulting in an oscillatory behavior.

As the loop moves into and out of the loop, the net magnetic flux flowing through the loop changes. This results in an induced emf in the loop that opposes the change in flux. In all cases where the loop enters the field, the flux through the loop increases, and a clockwise current flow will be induced in the loop to oppose this change. Similarly, whenever the loop exits the field, the flux through the loop decreases, and a counterclockwise current flow will be induced in the loop.

8.7.2 What Does This Example Illustrate?

The oscillator is a device where domain processes are activated based on the spatial configuration of its parts. The domain processes in turn change the spatial configuration. Thus, this example illustrates the need for reasoning about simultaneous changes in both the spatial and non-spatial properties of a physical system.

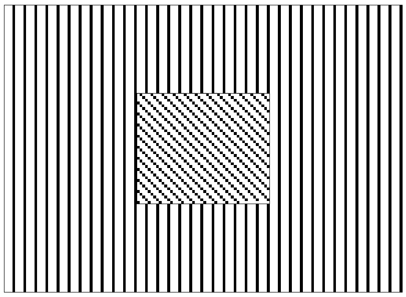
With respect to the features of our spatial model, this example further illustrates our ability to reason about the orientation of the *actual* surfaces of an object, and our ability to perform qualitative vector operations, such as the cross product operation. This example also exercises our ability to reason about the relative positions of objects and to reason about the effects of translational motion.

The fact that the net magnetic force acts on the left edge of the loop in the initial state is determined by a Lisp function. This function considers the shape and orientation of the loop, along with knowledge of the relative positions of the loop and the field, to determine where the net magnetic force acts on the loop. In this case, we note that the loop is rectangular is shape and that it is aligned with the coordinate axes. The relative position of the loop and the field is that the loop overlaps the right boundary of the field.

In this situation, the magnetic force vectors that act on the top and bottom edges of the loop cancel, and the net force on the loop is given by the magnetic force acting on the left edge of the loop. These conclusions are illustrated by the assertions in the final knowledge base description of the loop, as shown in Section 8.7.6.

Incidentally, in our motor examples, as illustrated by the situation in Figure 8.13, we consider cases where the magnetic forces acting on a conducting loop placed within a magnetic field cancel, but where there is a net torque. This results in the rotational motion of the loop.

- **Diagram:**



- **Text:** None Required!

Figure 8.13: A magnetic fields problem which illustrates the basic principles of operation of a motor. Here, the magnetic field is in the +Y direction, and the permanent current flow in the loop is in the clockwise direction. This configuration results in a zero net magnetic force on the loop, but there is a net torque which results in a counterclockwise rotation.

8.7.3 Input Processing For This Example

The following output shows the results of processing the diagram shown in Figure 8.12. In the resulting description of conducting loop, `l`, the fact that there is a permanent counterclockwise current flow in the loop is extracted directly from the Picture Semantics description for the conducting loop. Similarly, the direction of the magnetic field, and the fact that it is a steady, uniform field, is extracted from the Picture Semantics description for the magnetic field. Assertions based on Picture Semantics descriptions are marked with a dash (-). The remaining assertions are made by the Figure Understander.

```
*****************************************************************
***                    Parsing The Figure                     ***
*****************************************************************

Choosing the new frame of reference.

The old coordinate system is being rotated by 0 degrees.

Determining new vertex coordinates for all objects.

Extracting qualitative information about the figure.
```

```
The text description accompanying the figure has been processed.

Creating a postscript file for the figure after the change in the
frame of reference.

The processing of the input figure has been completed.
NIL

> (algy)
Algernon 1.2.4 of 16 December 1992.

algy> vf l
L: [slot]

  name:       ((l) nil)
  isa:        (convex-polygons nil) (mobile-objects nil)
              (convex-2-d-objects nil) (solid-objects nil)
              (field-sources nil)
-              (conducting-loops nil)
              (objects nil) (world-objects nil) (things nil)
  is-rectangular: (true nil)
-  is-conducting-object: (true nil)
-  is-solid:   (true nil)
-  mobility-of: (mobile nil)
-  permanent-steady-current-flow: (true nil)
-  loop-current-flow-direction: ((ccw p-init) nil)
-  current-flow-source: (permanent nil)
-  exists-current-flow: ((true p-init) nil) ((true p) nil)
  has-bottommost-edge: ((true p-init) nil)
  has-leftmost-edge: ((true p-init) nil)
  has-topmost-edge: ((true p-init) nil)
  has-rightmost-edge: ((true p-init) nil)
  no-torque-point: ((l-e-4 edge-center) nil) ((l-e-3 edge-center) nil)
                   ((l-e-2 edge-center) nil) ((l-e-1 edge-center) nil)
  box-bounding-shape: (true nil)

algy> vf f
F: [slot]

  name:       ((f) nil)
  isa:        (convex-2-d-objects nil) (permeable-objects nil)
              (immobile-objects nil) (magnetic-fields nil)
-              (steady-uniform-magnetic-fields nil) (objects nil)
              (world-objects nil) (things nil)
-  is-rectangular: (true nil)
-  mobility-of: (immobile nil)
-  is-conducting-object: (false nil)
-  field-type-of: (uniform nil) (steady nil)
-  is-solid:   (false nil)
-  field-direction-of: ((0 0 1) nil)
  box-bounding-shape: (true nil)
```

8.7.4 The Models Produced by QPC

Since the direction of motion of the loop is only determined during qualitative simulation, QPC executes a three-way branch during its initial model construction step on the assumption that the flux through the loop could be increasing, decreasing, or constant. The correct model is the one that assumes that the flux through the loop is increasing in the initial model. This is given by model M-2. The following trace shows the QPC model fragments from our spatial and domain models that are found to be active by QPC in model M-2.

```
Instantiated views for M-2:
  0. (properties-of-field-from-ccw-current-flow l l.p-init.source-of-field)
  1. (aligned-x-view l f)
  2. (aligned-x-view f l)
  3. (aligned-y-view l f)
  4. (bb-overlap-view l f)
  5. (aligned-y-view f l)
  6. (no-rotation l)
  7. (constraints-on-mobile-objects l)
  8. (zero-theta-z l)
  9. (overlap-bb-right-view l f)
 10. (flux-through-loop-in-su-perm-field f l)
 11. (increasing-flux-through-loop-pos-fld f l)
 12. (cw-induced-emf-in-perm-current-flow-loop l)
 13. (create-field-from-current-in-loop l)
 14. (setup-magnetic-force-on-loop-in-field-xy-plane l f)
 15. (neg-x-dir-mag-force-on-loop l)
 16. (zero-y-dir-mag-force-on-loop l)
 17. (zero-z-dir-mag-force-on-loop l)
 18. (mag-force-magnitude-4 l f)
 19. (force-on-loop l)
 20. (resolve-induced-cw-current-flow-in-loop-1 f l)
 21. (permanent-current-flow-non-reversed-loop-initial-ccw l)
```

The last view (21) defines the properties of the permanent counterclockwise current flow in the loop. In the transformer example, it was known that the loop was immobile. In this example, the loop is mobile, and therefore, in our domain model, we have to consider the fact that the initial counterclockwise current flow could appear to be clockwise if the loop becomes inverted due to rotation. In fact, that case arises in the motor example shown in Figure 8.13.

The QPC views which model the properties of a permanent counterclockwise current flow in a mobile conducting loop are shown in Figure 8.14. The cases where the loop is inverted and not inverted are both illustrated. Note that the current flow variable is modeled as piecewise-continuous, since its sign can change discontinuously between models if the loop were to become inverted.

The qualitative simulation of model M-2 results in one behavior, a transition to the spatial state where the loop is completely within the magnetic field. Then, after the loop reaches the left boundary of the field, it begins to exit the field. The trace below shows the model fragments that are instantiated when the loop exits the field to the left, resulting in a counterclockwise induced emf in the loop and a net magnetic force in the positive X direction.

```
Instantiated views for M-9:
  0. (properties-of-field-from-ccw-current-flow l l.m-8.source-of-field)
  1. (aligned-x-view l f)
  2. (aligned-x-view f l)
  3. (aligned-y-view l f)
  4. (bb-overlap-view l f)
  5. (aligned-y-view f l)
  6. (no-rotation l)
  7. (constraints-on-mobile-objects l)
  8. (zero-theta-z l)
  9. (overlap-bb-left-view l f)
 10. (flux-through-loop-in-su-perm-field f l)
 11. (decreasing-flux-through-loop-pos-fld f l)
 12. (ccw-induced-emf-in-perm-current-flow-loop l)
 13. (create-field-from-current-in-loop l)
 14. (setup-magnetic-force-on-loop-in-field-xy-plane l f)
 15. (pos-x-dir-mag-force-on-loop l)
 16. (zero-y-dir-mag-force-on-loop l)
 17. (zero-z-dir-mag-force-on-loop l)
 18. (mag-force-magnitude-4 l f)
 19. (force-on-loop l)
 20. (resolve-induced-ccw-current-flow-in-loop-2 f l)
 21. (permanent-current-flow-non-reversed-loop-initial-ccw l)
```

8.7.5 The Qualitative Behaviors For This Example

The tree of behaviors produced for this example is shown in Figure 8.15. The qualitative behavior plot that shows the oscillatory behavior is shown in Figure 8.16. Since the oscillatory behavior is infinite, the simulation was forcibly terminated after one cycle, as shown in Figure 8.16. The point at which the simulation was terminated is shown with the label of 26 in Figure 8.15.

Once again, the last four variables in the behavior plot show the inequality

```
(defview permanent-current-flow-non-reversed-loop-initial-CCW ()
  :individuals ((loop :type conducting-loops
                      :conditions
                      ((current-flow-source loop permanent)
                       (permanent-steady-current-flow loop true)
                       (scenario ?model ?sce)
                       (initial-model ?sce ?init)
                       (loop-current-flow-direction loop CCW ?init)
                       (orientation-type loop not-reversed ?model))))
  :relations ((q< (loop-current-flow loop) zero)
              (q< (perm-emf loop) zero)
              (constant (perm-emf loop))
              (q> (loop-current-flow loop) minval)
              (exists-current-flow loop TRUE ?model)
              (loop-current-flow-direction loop CCW ?model)
              (piecewise-continuous-variable
               ?model (loop-current-flow-deriv loop))
              (piecewise-continuous-variable
               ?model (loop-current-flow loop))))

; initial CCW current in loop now appears CW due to reversed orientation.
(defview permanent-current-flow-reversed-loop-initial-CCW ()
  :individuals ((loop :type conducting-loops
                      :conditions
                      ((current-flow-source loop permanent)
                       (permanent-steady-current-flow loop true)
                       (scenario ?model ?sce)
                       (initial-model ?sce ?init)
                       (loop-current-flow-direction loop CCW ?init)
                       (orientation-type loop reversed ?model))))
  :relations ((q> (loop-current-flow loop) zero)
              (q< (loop-current-flow loop) maxval)
              (q> (perm-emf loop) zero)
              (constant (perm-emf loop))
              (exists-current-flow loop TRUE ?model)
              (loop-current-flow-direction loop CW ?model)
              (piecewise-continuous-variable
               ?model (loop-current-flow-deriv loop))
              (piecewise-continuous-variable
               ?model (loop-current-flow loop))))
```

Figure 8.14: Views in the magnetic fields domain model which model the properties of permanent current flow in a mobile conducting loop. These views take into the account that the direction of current flow could appear to be reversed if the orientation of the loop becomes reversed.

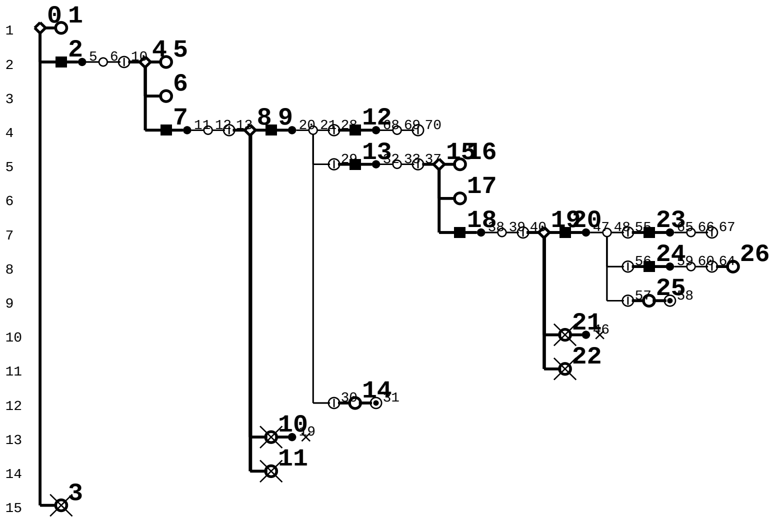


Figure 8.15: Tree of all behaviors produced for the problem given in Figure 8.12.

relations between the X-coordinate values of the rightmost and leftmost edges of the bounding boxes of the loop and the field. The plots show that the inequality relations $(lm(f) < rm(l))$ and $(rm(f) > lm(l))$ are always true. Therefore, it can be inferred that the loop and field are always overlapped in this behavior plot.

Note that this example requires the ability to perform complex, integrated reasoning about simultaneous dynamic change in both spatial and non-spatial properties. For example, the value for the variable `l.net-mag-force-x` shows that the net force on the loop is in the -X direction whenever the loop overlaps the right boundary of the field, and that it is in the +X direction whenever the loop overlaps the left boundary of the field. The magnitude of the magnetic force is changing whenever it is non-zero because as the force affects the velocity of the loop (l.x-velocity), the derivative of the magnetic flux passing through the loop (l.flux-through-loop-deriv) changes, and this causes an induced emf (l.induced-emf-mag) to appear in the loop. This in turn affects the net current flow (l.loop-current-flow) in the loop, which in turn affects the strength of the magnetic force.

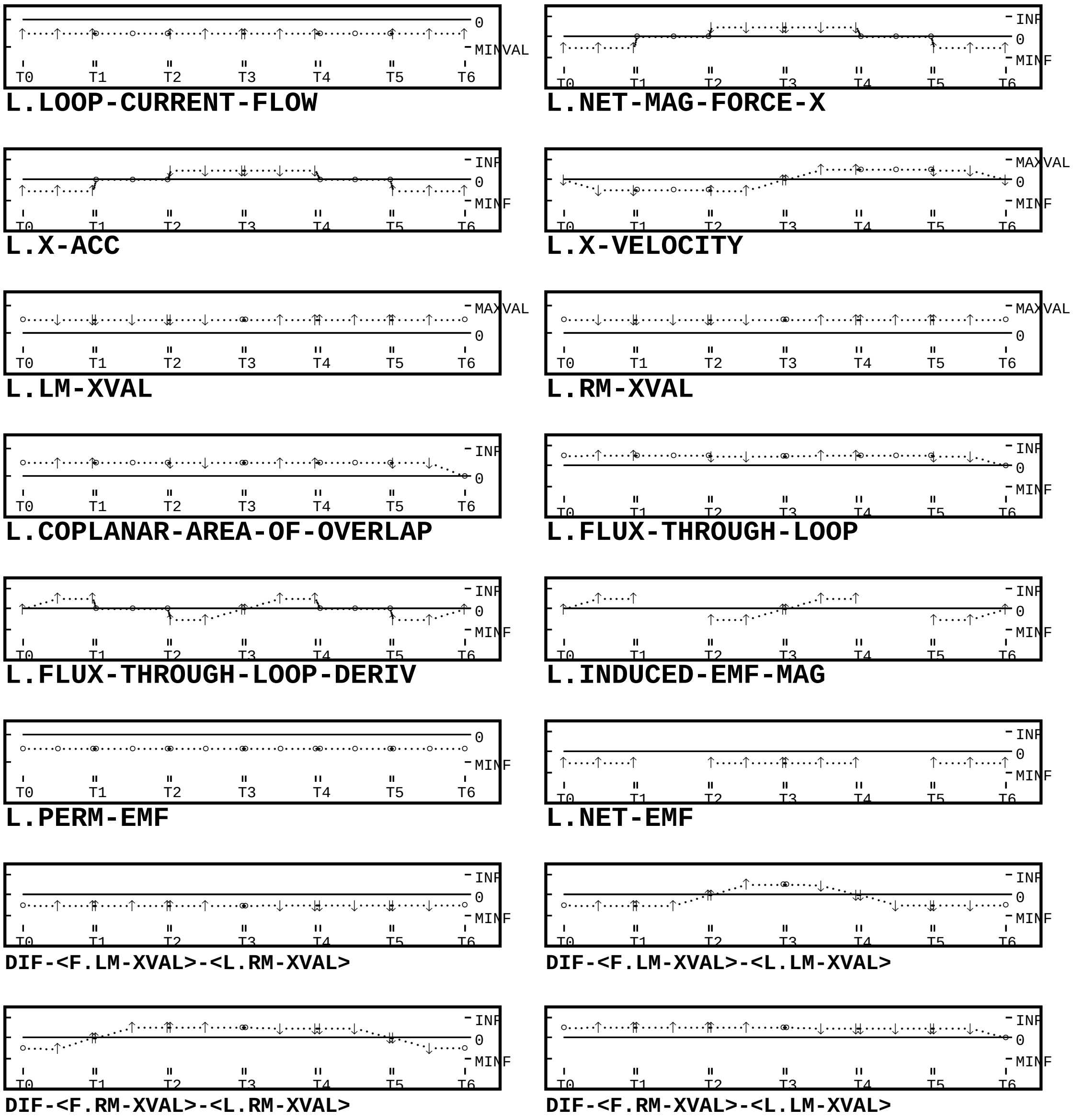


Figure 8.16: Simulation output showing the oscillatory behavior for the problem given in Figure 8.12.

8.7.6 Final Knowledge Base Descriptions for the Objects

The following shows the relevant assertions made to the knowledge base description of the conducting loop during the simulation of models M-2 and M-9. These show the differences in the directions of the net magnetic force on the loop, and in the direction of the induced current flow, in the cases where the loop enters (M-2) and exits (M-9) the magnetic field.

```
algy> vf l
L: [slot]
  name:       ((l) nil)
  in-xy-plane: ((true m-9) nil) ((true m-2) nil)
  orientation-type: ((not-reversed m-9) nil) ((not-reversed m-2) nil)
  net-mag-force-vector: ((m-9 (1 0 0)) nil) ((m-2 (-1 0 0)) nil)
  induced-emf-dir: ((ccw m-9) nil) ((cw m-2) nil)
```

8.8 A Generator Example

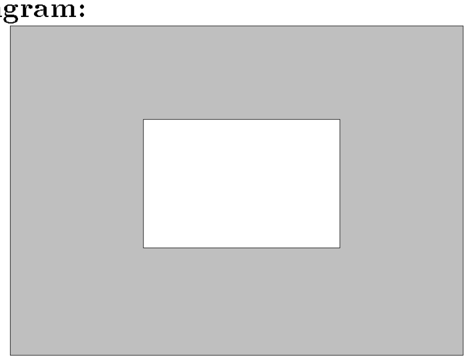
This example illustrates the basic operation of a generator, as described by the diagram and text shown in Figure 8.17. Here, a loop is rotated with a constant velocity within a magnetic field. As the loop rotates, its projected area of overlap with respect to the magnetic field changes, resulting in a change in the net magnetic flux that flows through the loop. This causes an induced emf, and therefore a current flow, to appear in the loop.

In this example, we do not model the torque on the loop due to any magnetic force that may act on the loop as it rotates. Instead, we will assume that external work is being done on the loop to compensate for any loss in energy due to any torque on the loop.

8.8.1 Problem Description

The diagram shows a conducting loop that is rotating in a counterclockwise direction about an axis parallel to the X-axis that passes through its center of gravity. Henceforth, we will simply use the phrase ‘rotating about the X-axis’ to describe the motion of the loop. As discussed in Chapter 4, the angle of rotation of the loop

- **Diagram:**



- **Text:** The loop is rotating in a counterclockwise direction about the X-axis.

Figure 8.17: A magnetic fields problem which requires reasoning about the basic principles of operation of a generator.

with respect to the X-axis is given by θ_Y. Initially, when θ_Y is zero, the magnetic flux through the loop is at a maximum value. As the loop rotates, its projected area onto the field decreases, and the number of lines of flux passing through the loop also decreases. This results in an induced emf in the loop which in turn causes a counterclockwise current flow in the loop.

The projected area of the loop onto the field is given by $A \times cos(\theta_Y)$, where A is the area of the loop. Since A is constant, the derivative the projected area of overlap of the loop onto the field is a monotonically decreasing function of $sin(\theta_Y)$. Then, the derivative of the flux passing through the loop is also a monotonically decreasing function of $sin(\theta_Y)$. The magnitude of the emf, V, generated in the loop is given by $V = d\Phi/dt$, where Φ is the net magnetic flux passing through the loop.

The result is that as θ_Y increases towards 90 degrees, the value of $sin(\theta_Y)$ increases, thus increasing the magnitude of dΦ/dt and magnitude of the current flow generated in the loop. When θ_Y becomes 90 degrees, the projection of the loop onto the magnetic field is a line, and the flux through the loop is zero. This marks the point when the orientation of the loop would appear to be inverted when we continue to view it from above. Since the loop is inverted during the interval when θ_Y is between 90 and 180 degrees, the flux through the loop has as a negative value, and continues to decrease (becomes more negative). Thus, the current flow in the loop will continue to appear to be in the counterclockwise direction when viewed from above. Since the value of $sin(\theta_Y)$ decreases towards zero, the magnitude of the current flow generated in the loop also decreases towards zero.

When θ_Y becomes 180 degrees, the loop will appear to be completely inverted, and the flux through the loop will be at a maximum negative value. As the loop continues to rotate counterclockwise, the flux through the loop will begin to increase towards zero, and a clockwise current flow will appear in the loop to oppose this change. After an angle of rotation of 270 degrees, the magnitude of the current flow in the loop will once again be at a maximum value. The projection of the loop onto the field will be a line, and as the loop continues to rotate counterclockwise, it will no longer appear to be inverted. Then, since the qualitative value of the flux through the loop will be positive and increasing, the induced current flow will continue to be in the clockwise direction. The magnitude of this current flow will once again reach zero when θ_Y reaches 360 degrees.

8.8.2 What Does This Example Illustrate?

This example illustrates our ability to reason about the projected area of overlap between two objects and the inverted orientation of objects due to rotation. It also tests our ability to model and reason about variables with circular quantity spaces, such as θ_Y. In this example, after the loop makes a complete revolution, we benefit from the ability to assert that the value of 360 degrees for θ_Y is the same as a value of 0 degrees. Then, the view described in Figure 4.4, which models θ_Y values of 0 to 90 degrees, can be reinvoked.

8.8.3 Input Processing For This Example

The following output shows the results of processing the diagram shown in Figure 8.17. A partial display of the initial knowledge base description of the conducting loop and the magnetic field follows. Inferences made from processing the text in Figure 8.17 are marked with a star (*). The assertions based on the Picture Semantics descriptions of the loop and the field are marked with a dash (-).

```
*****************************************************************
***                    Parsing The Figure                     ***
*****************************************************************

Choosing the new frame of reference.

The old coordinate system is being rotated by 0 degrees.

Determining new vertex coordinates for all objects.

Extracting qualitative information about the figure.

Interpreting text sentence:
(THE LOOP IS ROTATING IN A COUNTERCLOCKWISE DIRECTION ABOUT THE X-AXIS)

The ATN parse of the sentence:
`` (THE LOOP IS ROTATING IN A COUNTERCLOCKWISE
    DIRECTION ABOUT THE X-AXIS) '' is:

(S (NP (DET THE) (NOUN LOOP))
 (VP (AUX IS) (VERB ROTATING)
  (CLAUSE (PP (PREP IN)
              (NP (DET A) (ADJ COUNTERCLOCKWISE) (NOUN DIRECTION))))
  (CLAUSE (AP (ADV ABOUT) (NP (DET THE) (NOUN X-AXIS))))))

The text description accompanying the figure has been processed.

 The following Algernon path shall be asserted:

 Algy path =

((:THE ?V1 (MEMBER CONDUCTING-LOOPS ?V1) (MOBILITY-OF ?V1 MOBILE))
 (ROTATION-DIRECTION ?V1 CCW P) (AXIS-OF-ROTATION ?V1 X P)
 (IS-ROTATING ?V1 TRUE P))

The text description accompanying the figure has been processed.

Creating a postscript file for the figure after the change in the
frame of reference.

The processing of the input figure has been completed.
NIL

> (algy)
```

```
Algernon 1.2.4 of 16 December 1992.
algy> vf l
L:
  name:         ((l) nil)
  isa:          (convex-polygons nil) (mobile-objects nil)
                (convex-2-d-objects nil) (solid-objects nil)
-                (conducting-loops nil)
                (field-sources nil)
                (objects nil) (world-objects nil) (things nil)
-  is-conducting-object: (true nil)
-  is-solid:   (true nil)
-  mobility-of: (mobile nil)
  circular-bounding-shape: (true nil)
*  rotation-direction: ((ccw p) nil)
*  axis-of-rotation: ((x p) nil)
*  is-rotating: ((true p) nil)

algy> vf f
F:
  name:         ((f) nil)
  isa:          (convex-polygons nil) (convex-2-d-objects nil)
                (permeable-objects nil) (immobile-objects nil)
-                (magnetic-fields nil) (steady-uniform-magnetic-fields nil)
                (objects nil) (world-objects nil) (things nil)
-  is-rectangular: (true nil)
-  mobility-of: (immobile nil)
-  is-conducting-object: (false nil)
-  field-type-of: (uniform nil) (steady nil)
-  is-solid:   (false nil)
-  field-direction-of: ((0 0 1) nil)
  box-bounding-shape: (true nil)
```

8.8.4 The Models Produced by QPC

Initially, QPC executes a 6-way branch in its model creation process based on assumptions about whether θ_Y is increasing or decreasing and whether the flux through the loop is increasing or decreasing. The correct model is the one that assumes that θ_Y is increasing and that the flux through the loop is decreasing. This is given by Model M-1. The following trace shows the QPC model fragments from our spatial and domain models that are found to be active by QPC in model M-1.

```
Instantiated views for M-1:
  0. (aligned-x-view l f)
  1. (aligned-x-view f l)
  2. (aligned-y-view l f)
  3. (bb-overlap-view l f)
  4. (aligned-y-view f l)
  5. (rotation-theta-y-0-to-90 l)
  6. (no-rotation-around-y-axis-special-case l)
  7. (no-rotation-around-y-axis l)
```

```
  8. (zero-theta-z l)
  9. (rotation-around-x-axis l)
 10. (constraints-on-mobile-objects l)
 11. (no-trans-motion l)
 12. (proj-area-overlap-no-trans-motion-special-case l f)
 13. (bb-enclosed-view l f)
 14. (flux-through-loop-in-su-perm-field f l)
 15. (decreasing-flux-through-loop-pos-fld f l)
 16. (ccw-induced-emf l)
 17. (create-field-from-current-in-loop l)
 18. (properties-of-field-from-ccw-current-flow l l.m-1.source-of-field)
 19. (resolve-induced-ccw-current-flow-in-loop-2 f l)
```

The qualitative simulation of model M-1 results in one behavior, a transition to the spatial state where the loop has rotated 90 degrees. The traces below show the views that are active in the models that cover rotation between 90 and 180 degrees (M-7), 180 and 270 degrees (M-8), and 270 and 360 degrees (M-9). Then, we show the views active in model M-10, which covers an angle of rotation of 360 to 450 degrees, to show that with the use of circular quantity spaces, this case is treated identically to the case modeled by M-1, where the angle of rotation goes from 0 to 90 degrees.

Note that an example QPC model fragment which covers angles of rotation of 0 to 90 degrees was shown in Figure 4.4. The view descriptions for the other three rotation intervals are similar.

```
Instantiated views for M-7:
  0. (aligned-x-view f l)
  1. (aligned-x-view l f)
  2. (aligned-y-view f l)
  3. (aligned-y-view l f)
  4. (bb-overlap-view l f)
  5. (bb-enclosed-view l f)
  6. (flux-through-loop-in-su-perm-field f l)
  7. (decreasing-flux-through-loop-pos-fld f l)
  8. (ccw-induced-emf l)
  9. (create-field-from-current-in-loop l)
 10. (properties-of-field-from-ccw-current-flow l l.m-7.source-of-field)
 11. (resolve-induced-ccw-current-flow-in-loop-2 f l)
 12. (rotation-theta-y-90-to-180 l)
 13. (no-rotation-around-y-axis l)
 14. (no-rotation-around-y-axis-special-case l)
 15. (zero-theta-z l)
 16. (rotation-around-x-axis l)
 17. (constraints-on-mobile-objects l)
 18. (no-trans-motion l)
 19. (proj-area-overlap-no-trans-motion-special-case l f)
```

```
Instantiated views for M-8:
  0. (aligned-x-view l f)
  1. (aligned-x-view f l)
  2. (aligned-y-view l f)
  3. (bb-overlap-view l f)
  4. (bb-enclosed-view l f)
  5. (flux-through-loop-in-su-perm-field f l)
  6. (increasing-flux-through-loop-pos-fld f l)
  7. (cw-induced-emf l)
  8. (create-field-from-current-in-loop l)
  9. (properties-of-field-from-cw-current-flow l l.m-8.source-of-field)
 10. (resolve-induced-cw-current-flow-in-loop-1 f l)
 11. (aligned-y-view f l)
 12. (rotation-theta-y-180-to-270 l)
 13. (no-rotation-around-y-axis l)
 14. (no-rotation-around-y-axis-special-case l)
 15. (zero-theta-z l)
 16. (rotation-around-x-axis l)
 17. (constraints-on-mobile-objects l)
 18. (no-trans-motion l)
 19. (proj-area-overlap-no-trans-motion-special-case l f)

Instantiated views for M-9:
  0. (aligned-x-view f l)
  1. (aligned-x-view l f)
  2. (aligned-y-view f l)
  3. (aligned-y-view l f)
  4. (bb-overlap-view l f)
  5. (bb-enclosed-view l f)
  6. (flux-through-loop-in-su-perm-field f l)
  7. (increasing-flux-through-loop-pos-fld f l)
  8. (cw-induced-emf l)
  9. (create-field-from-current-in-loop l)
 10. (properties-of-field-from-cw-current-flow l l.m-9.source-of-field)
 11. (resolve-induced-cw-current-flow-in-loop-1 f l)
 12. (rotation-theta-y-270-to-360 l)
 13. (no-rotation-around-y-axis l)
 14. (no-rotation-around-y-axis-special-case l)
 15. (zero-theta-z l)
 16. (rotation-around-x-axis l)
 17. (constraints-on-mobile-objects l)
 18. (no-trans-motion l)
 19. (proj-area-overlap-no-trans-motion-special-case l f)

Instantiated views for M-10:
  0. (aligned-x-view l f)
  1. (aligned-x-view f l)
  2. (aligned-y-view l f)
  3. (bb-overlap-view l f)
  4. (bb-enclosed-view l f)
  5. (flux-through-loop-in-su-perm-field f l)
  6. (decreasing-flux-through-loop-pos-fld f l)
  7. (ccw-induced-emf l)
  8. (create-field-from-current-in-loop l)
  9. (properties-of-field-from-ccw-current-flow l l.m-10.source-of-field)
 10. (resolve-induced-ccw-current-flow-in-loop-2 f l)
```

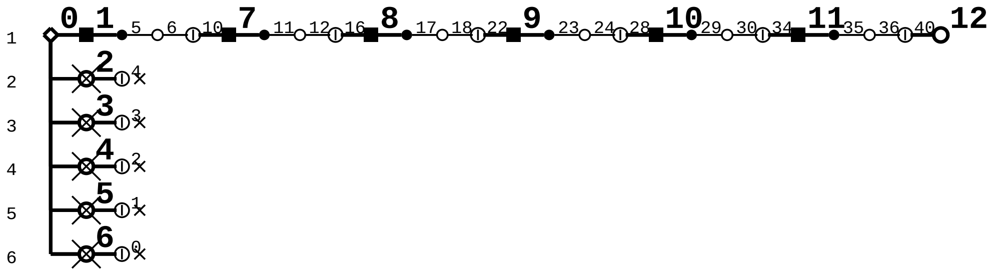


Figure 8.18: Tree of all behaviors produced for the problem given in Figure 8.17.

```
11. (aligned-y-view f l)
12. (rotation-theta-y-0-to-90 l)
13. (no-rotation-around-y-axis l)
14. (no-rotation-around-y-axis-special-case l)
15. (zero-theta-z l)
16. (rotation-around-x-axis l)
17. (constraints-on-mobile-objects l)
18. (no-trans-motion l)
19. (proj-area-overlap-no-trans-motion-special-case l f)
```

8.8.5 The Qualitative Behaviors For This Example

The tree of behaviors produced for this example is shown in Figure 8.18. The only qualitative behavior for this example is shown in Figure 8.19. The simulation was forcibly terminated after 1.5 revolutions of the loop.

In Figure 8.19, the last four variables model the inequality relations between θ_Y and each of the landmark values classes in its circular quantity space. Note the discontinuous changes in these difference terms. That again shown the benefit of our ability to model piecewise-continuous variables.

The plots for the remaining variables show the change in the projected area of overlap (l.proj-area-of-overlap) between the loop and the field as the loop is rotated in the field. This changes the flux through loop (l.flux-through-loop), and results in an induced emf (l.induced-emf-mag) and current flow (l.loop-current-flow) in the loop.

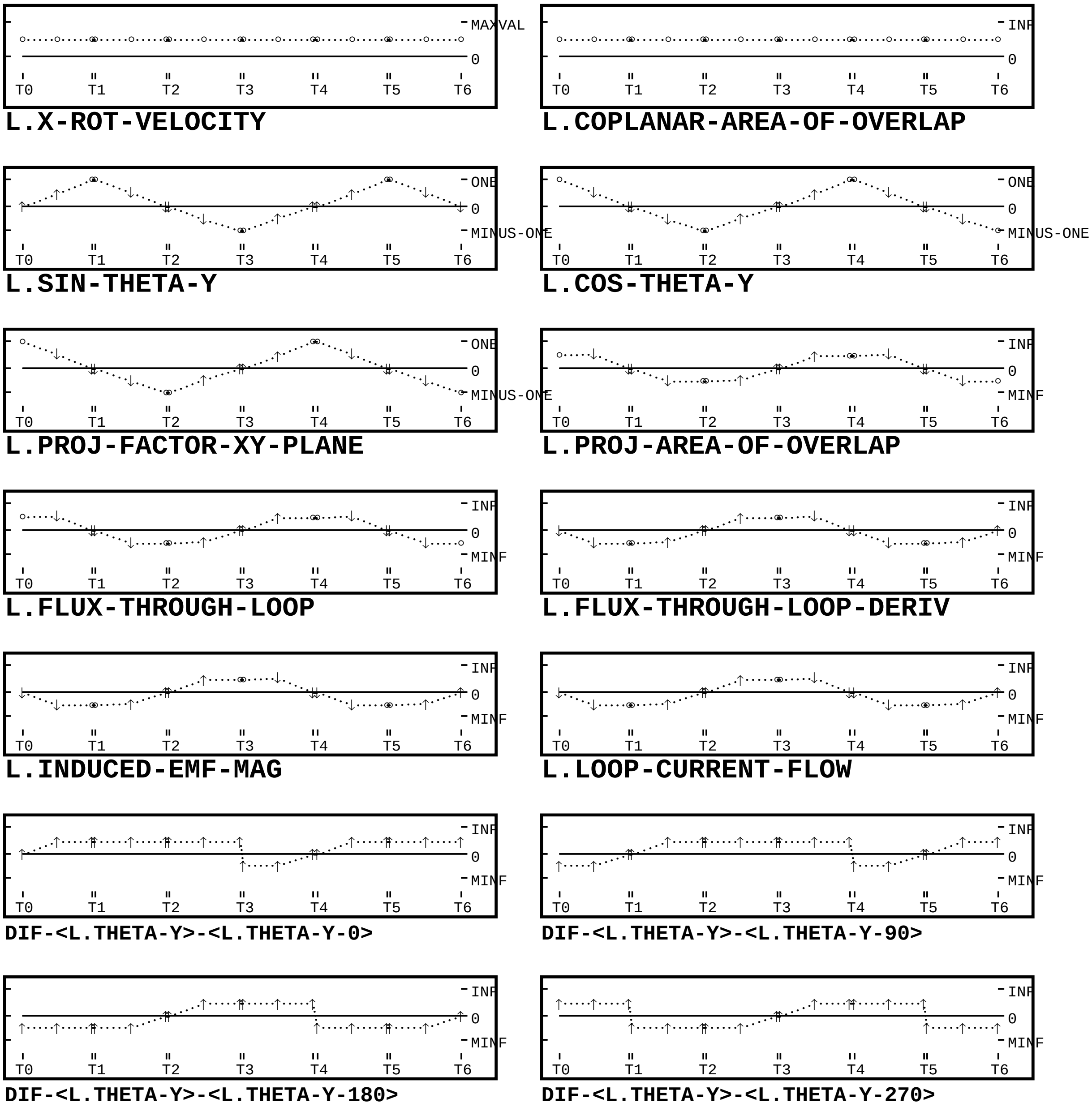


Figure 8.19: Simulation output showing the only qualitative behavior of the simple generator described in Figure 8.17.

8.8.6 Final Knowledge Base Descriptions for the Objects

The following shows the relevant assertions made to the knowledge base description of the conducting loop during the simulation of models M-1, M-7, M-8, and M-9. These models cover one complete revolution of the loop within the magnetic field. The output shows whether an object would appear to be inverted, and the direction of the current flow generated in the loop, for each of these models. The output trace shows that the loop will appear to be inverted when θ_Y has a value between 90 and 270 degrees (models M-7 and M-8). The direction of current flow reflects how it would appear in each model as we continuously view the loop from above the page.

```
algy> vf l
L: [slot]
  name:       ((l) nil)
  top-bottom-reversed: ((false m-9) nil) ((true m-8) nil)
                       ((false m-1) nil) ((true m-7) nil)

  loop-current-flow-direction: ((cw m-9) nil) ((cw m-8) nil)
                               ((ccw m-7) nil) ((ccw m-1) nil)
  exists-current-flow: ((true m-9) nil) ((true m-8) nil)
                       ((true m-1) nil) ((true m-7) nil)
  induced-emf-dir: ((cw m-9) nil) ((cw m-8) nil)
                   ((ccw m-1) nil) ((ccw m-7) nil)
```

8.9 Chapter Summary

This chapter has described the experiments that were conducted to evaluate the capabilities of our models for spatial reasoning. We presented four specific examples covering the principles of operation of transformers, oscillators, and generators which together illustrate the capability of our implemented systems, the Figure Understander and the Motion Understander, to reason about complex problems involving dynamic change in both the spatial and non-spatial properties of a given system.

These problems required reasoning simultaneously about both the relative positions of objects and the orientation of objects. For example, in determining what happens as a conducting loop moves through a magnetic field, knowledge of the relative positions of the loop and the field was used to conclude whether the area of overlap between the two objects was non-zero, and if so, whether the area of overlap was increasing, decreasing, or steady. At the same time, knowledge of the orientation of the loop was used to determine the direction of current flow that was induced in the loop, and to determine where a net magnetic force acted on the loop.

In solving these problems, we demonstrated the capabilities of the Figure Understander for integrating the information in text and diagram input into a single unified knowledge base description. We also demonstrated the ability of the Figure Understander to choose a frame of reference for a problem, and to choose bounding shape abstractions for objects.

Then, we demonstrated the capabilities of our spatial model for reasoning about the effects of translational and rotational motion, and finally, we demonstrated the effectiveness of our methods for modeling piecewise-continuous variables, for modeling objects whose existence is piecewise-continuous, and for modeling variables with circular quantity spaces.

Chapter 9

Conclusions

This dissertation has addressed the task of developing a qualitative spatial representation which can be used for qualitative reasoning about the relative positions and orientation of objects in both static worlds and worlds where there is dynamic change due to translational or rotational motion. Qualitative representations are important because they allow reasoning about situations where only qualitative descriptions of the world state are available.

In our work, we considered examples from the domain of understanding natural language text descriptions of a spatial scene, and from the domain of physics problem solving. The physics problems that we solve are *qualitative* exercises designed to test the student's knowledge of the fundamental properties of magnetic fields. Since they all ask the student to describe what happens as a given event occurs, we solve the problems through qualitative simulation.

This dissertation has made contributions in the area of qualitative spatial reasoning [68, 75, 71] and the area of integration of natural language and visual processing [74, 69, 73]. Together, the techniques we have developed have allowed us to implement the first spatial reasoning system that starts with a diagram and text description of the initial state of a problem, and automatically generates the qualitative behaviors that are achievable from the initial state.

In the remainder of this chapter, we will present an evaluation of our spatial reasoning methods based on the successes that were achieved and the limitations that were discovered in conducting the experimental evaluation described in Chapter 8. We will conclude our presentation with a discussion of directions for future work.

9.1 Strengths and Limitations of Our Work

Table 8.1, reproduced below as Table 9.1, lists the 8 basic sets of examples we used to evaluate the effectiveness of our spatial reasoning methods. The items in the left column of Table 9.1 represent the features of the Figure Understander and our spatial representation that were exercised by each set of examples.

The advantages and limitations of our Picture Semantics-based approach for solving the coreference problem in integrating diagram and text input were described in detail in Section 5.7. Our goal now is provide an evaluation of our methods for qualitative spatial reasoning.

9.1.1 Our Model of the Relative Positions of Objects

Our approach for reasoning about directional and topological spatial relations between objects was to first define a common representation for the spatial extent of an object. To achieve this, instead of using the actual shape of an object, we use simplifying shape abstractions to approximate its spatial extent. We use the adequacy criterion that the shape abstraction must fully cover the object at all times. This allows us to identify the spatial configurations where it is *at all possible* for two objects to be intersecting. This property is important in reasoning about the effects of magnetic fields since a magnetic field can only affect objects that are within the field.

Regardless of the bounding shape that is chosen for an object, we describe the spatial extent of an object through the rectangular bounding box projected onto the X-Y plane by its bounding shape. The bounding box is drawn to be aligned with the extrinsic Cartesian coordinate system for the problem.

Problem Features, Methods Tested	ScUnd	RdUnd	Auto	Trns	Osc	Rmp	Gen	Mtr
Static World Problem	X	X						
Dynamic Change Problem			X	X	X	X	X	X
Diagram Processing	X	X	X	X	X	X	X	X
Text/Diagram Integration	X			X		X	X	
Int/Deictic Ref Frames	X							
Alt Frame Ref Selection						X		
Relative Positions	X	X	X	X	X	X	X	X
Orientation of Surfaces					X	X	X	X
Orientation of Vectors	X	X	X	X	X	X	X	X
Translational Motion			X		X	X		
Rotational Motion							X	X
Projection onto XY Plane							X	X
Inversion of Objects							X	X
Magnetic Forces					X	X	X	X
Torque							X	X
Vector Cross Products					X	X	X	X
Piecewise-Cont Vars				X	X	X	X	X
Piecewise-Cont Objects				X	X	X		
Circular Quantity Spcs							X	X
Multiple Moving Objects			X					
Motion In Multiple Dirs			X		X	X		

Table 9.1: Table showing the spatial reasoning features that were highlighted by each of the problem classes we have considered in evaluating our work. The problem classes are: ScUnd - urban scene understanding, RdUnd - understanding road configurations, Auto - motion of automobiles, Trns - transformers, Osc - oscillators, Rmp - problems involving translational motion of a conducting object through a magnetic field, Gen - generators, Mot - motors.

Since the bounding boxes of all objects are aligned with the coordinate axes, we describe the relative positions of objects through inequality constraints on the X-coordinate values of the rightmost and leftmost edges of their bounding boxes, and the Y-coordinate values of the topmost and bottommost edges of their bounding boxes. In our examples, we showed how spatial models based on inequality constraints could be straightforwardly implemented for use in qualitative simulation systems such as QPC [9, 17].

Limitations of Our Model of Relative Positions

The advantage of our model of the relative positions of objects is that we employ shape abstractions to assign a uniform shape for all objects. Since this shape is a rectangle aligned with the frame of reference for the problem, we showed that inequality constraints could be used to *qualitatively* model the relative positions of objects.

The weakness of this approach is a loss in precision. Since the actual shape of an object may not be rectangular, it is possible that conclusions based on the relative positions of the bounding boxes of objects may not apply to the actual objects.

For example, in Section 8.6, we considered what would happen as a *circular* conducting loop moved down a track through a magnetic field. However, for reasoning about the area of overlap between the loop and the field, we ignored the circular shape of the loop, and instead approximated its spatial extent through a rectangular bounding box. In the behaviors shown in Figures 8.9, 8.10, and 8.11, the derivative of the magnetic flux through the loop changes discontinuously to zero once the loop in enclosed in the loop. This would be the case if the loop were rectangular in shape. However, for a circular loop, after the loop is half-enclosed in the field, the derivative of its area of overlap with field, and therefore, the derivative of the magnetic flux passing through the loop, would begin to move towards zero. The net effect is that there is no discontinuous change in the derivative of the flux through the loop once it is fully enclosed in the field.

Unfortunately, there can be a loss of precision whenever any shape abstraction is used. However, it can be observed that conclusions about the relative positions of objects will increase in accuracy as the bounding abstractions more closely match the actual spatial extent of the underlying objects.

While it may be important to base the choice of a bounding shape abstraction on the accuracy of the conclusions it provides, the ease of reasoning *qualitatively* about the chosen shape abstraction should also be a factor. For example, Cui, Cohn, and Randell [11] use convex hulls to bound objects, which are likely to provide a better approximation of the actual spatial extents of objects than rectangular bounding boxes. However, Cui, Cohn, and Randell rely on an external *quantitative* function to compute the convex hulls of objects. When only qualitative descriptions of the spatial state are available, it may not be possible to compute the convex hull of an object.

Further, Mukerjee and Joe [57, 58] argue that rectangular bounding boxes drawn around the intrinsic front of an object may better approximate the spatial extent of the object than bounding boxes drawn based on an arbitrary extrinsic frame of reference. The problem with Mukerjee and Joe's approach is that since the bounding boxes of any two objects may not be aligned, the relative positions of two objects cannot be described through inequality relations on the coordinates of the edges of the bounding boxes.

Mukerjee and Joe provide an alternate qualitative method, using collision parallelograms, to reason about the relative positions and orientation of objects. However, if objects are allowed to rotate, the locations of the collision parallelogram could change, and their new locations would have to be numerically computed.

Relative Positions and Motion

In our work, we reason not only with static worlds, but also with worlds where objects may translate and rotate. In our examples, we showed that it was possible to model the change in position of an object due to translational motion by describing its effects on the X-coordinate values of the rightmost and leftmost edges and the Y-coordinate values of the topmost and bottommost edges of the object's bounding box. We also showed how the truth values of the inequality constraints which describe the relative positions of objects will be affected by translational motion.

During rotation, depending on the criteria used to define the bounding boxes of objects, the shapes and positions of the bounding boxes could both change. Computing the changing shapes of bounding boxes in a qualitative manner is a non-trivial problem. To avoid this problem, the shape abstraction that we use for objects that can rotate is a bounding sphere drawn to cover all points that could be touched by the object as it rotates about any axis passing through its center of gravity. Then, if the rectangular bounding box projected onto the X-Y plane by its bounding sphere is used to model the spatial extent of an object, it can be guaranteed that the size and shape of the bounding box will remain constant if the object rotates.

The limitations of this method are (a) that the types of rotation allowed are restricted, and (b) that there is a greater loss in precision since the bounding sphere will often be larger than a minimum bounding rectangle. For example, we cannot reason about situations where the rotation of an object removes its overlap with another object.

9.1.2 Reasoning about Orientation

The problems that we considered in evaluating this work use knowledge of the orientation of objects for several purposes. First, knowledge of the orientation of the individual surfaces of an object is used to determine the intrinsic front direction of an object or the direction of current flow at a given point on the surface of an object. We determine the orientation of any surface of a convex polygon based on its location between the extremal points of the object.

Second, many of the problems required the use of vectors. In our work, we used vectors to describe intrinsic front directions, directions of motion, directions of forces, and the directions of magnetic fields. The solutions to the problems used to evaluate our work made extensive use of our methods for qualitatively performing such vector operations as the cross product operation.

Third, many of the problems required the determination of *where* magnetic forces could act on a conducting loop, and whether the action of these forces would result in a non-zero net force or torque. For these problems, we used our model of the overlap between objects to determine which surfaces of the conducting loop were fully enclosed or partially enclosed in the field. Then, knowledge of the orientation of the loop was used to determine which magnetic forces acting on the loop would cancel, and which would result in a torque or a non-zero net force on the object.

Reasoning about Rotation

The only type of rotation that was covered in the evaluation was the rotation of an object about its center of gravity about an axis parallel to the X or Y axis. Here, the most important effects of the rotation are that the projected area of the object onto the X-Y plane could change, and that the object could become inverted. In the generator example presented in Section 8.8, we demonstrated the ability of our spatial reasoning system to reason about these properties. We also demonstrated its ability to reason about variables with circular quantity spaces.

9.2 Directions for Future Work

To summarize the main contribution of this work, we have demonstrated, based on the examples presented in Chapter 8, in Appendix A, and the many other implemented examples not described in this dissertation, that we have developed a spatial representation that can be used to reason about both the relative positions and orientation of objects. Our spatial reasoning methods have been successfully used to reason about problems involving static worlds and worlds where there is dynamic change due to translational or rotational motion.

Although we have met our initial goals in performing this research, our results only provide a starting point, and there are many directions for future work.

Modeling the Relative Sizes of Objects

First, we do not provide a means to compare the relative sizes of objects. Our theory of the relative positions of objects effectively divides the world into nine regions around each object, where the rectangular bounding box for the object lies at the center. Then, if another object is in translational motion, changes in the relative positions of the two objects will occur whenever the moving object crosses any spatial region defined by the first object.

One possible source of branching during qualitative simulation is determining whether the moving object will be fully enclosed within a given region before it moves into the next region. This requires knowledge of the relative sizes of the objects. In our work, when a diagram is processed by the Figure Understander, it computes a bounding sphere for all objects and orders them by the size of the diameter of the bounding sphere. Our implemented model of enclosure assumes that an object can become enclosed in another object if the diameter of its bounding sphere is smaller. There are numerous spatial configurations where this model can be shown to be wrong. In future work, a formal model of the relative sizes of objects will need to be added.

Race Conditions

When more than one object is in motion, we encounter the problem of which object will first cause a qualitatively significant change in the spatial state. To resolve these *race conditions*, more precise information will be required, such as the distances between objects. Without any additional information, the only recourse is to create branches in the qualitative simulation for each possible case.

In an experiment involving the qualitative simulation of the effects of translational motion on the relative positions of two automobiles, we specified neither their relative sizes nor their relative speeds and distance from an intersection where a collision could occur. A large number of behaviors resulted, with branching based on the relative sizes of the automobiles, and race conditions regarding which automobile would enter the intersection first, and where a collision would occur. The eventual result was that all available memory was exhausted before the simulation of the problem completed!

Rotation in the X-Y Plane

We presented arguments in Chapter 4 that reasoning qualitatively about the change in the orientation of an object as it rotates in the X-Y plane appears to be an inherently ambiguous problem. Since our example sets didn't require reasoning about rotation in the X-Y plane, we did not implement any solutions to this problem. However, it would be an interesting exercise for future work to determine if any qualitative solutions exist. A related problem is: Given knowledge of the relative positions of a group of objects with respect to one frame of reference, can the relative positions of the objects be determined if the frame of reference is changed? Freksa [27] is actively pursuing solutions to this problem.

Mechanical Constraint Problems

Recently, there has been much work in solving kinematics problems through qualitative reasoning [23, 36, 40]. The methods that have been developed rely primarily on the use of Configuration Spaces [23] to provide as input a description of all the legal spatial configurations of a kinematic system. The interesting qualitative behaviors are determined by studying the motions that are possible based on the Configuration Space for a given system.

In our work with the magnetic fields domain, we consider both solid and permeable objects. Therefore, the use of a Configuration Space to identify the forbidden spatial configurations (based on the fact that two solid objects cannot overlap in space) may not be helpful. Other properties true of the magnetic fields domain that are not compatible with the use of Configuration Spaces are that objects can have infinite spatial extent (e.g., the magnetic field around a current carrying conductor), and that spatial objects can be dynamically created and destroyed.

Nielsen [61] presents several *qualitative* methods for reasoning about mechanical constraints. The diagram processing capabilities of the Figure Understander include the ability to extract several of the spatial properties of objects used by

Nielsen. It would be an intersecting direction for future work to test whether the mechanical constraint problems addressed by Nielsen can be solved in the framework of our spatial representation.

Appendix A

A Complete Magnetic Fields Example

This appendix shows the output produced for the magnetic fields problem described through the diagram and text given in Figure A.1. The corresponding diagram from Resnick and Halliday's physics text [76] is shown in Figure A.2.

A.1 Description of the Example

The problem in Resnick and Halliday's text is to determine the resulting behavior as a conducting loop is pulled out of the magnetic field at a steady velocity $\mathbf{v}$. As the loop is pulled out of the field, the net magnetic flux passing through the loop changes, resulting in an induced electromotive force (emf) which causes a clockwise current flow in the loop. Then, magnetic forces act on the loop, as shown in Figure A.2, resulting in a net force which opposes the motion of the loop. Since the loop is being pulled out of the magnetic field at a steady velocity, thermal energy appears in the loop at a rate equal to the rate at which mechanical work is done on the loop.

In our magnetic fields domain model, we do not consider thermal energy. Therefore, we chose to solve the alternate problem where the loop has a non-zero initial velocity of $\mathbf{v}$, but where no external work is being done on the loop to keep the velocity steady. Then, as magnetic forces act on the loop, the velocity of the loop will decrease. This results in three possible qualitative behaviors - the loop

- **Diagram:**

- **Text:** The loop is moving out of the field.

Figure A.1: A magnetic fields problem showing a conducting loop moving out of a magnetic field whose direction is pointed into the page.

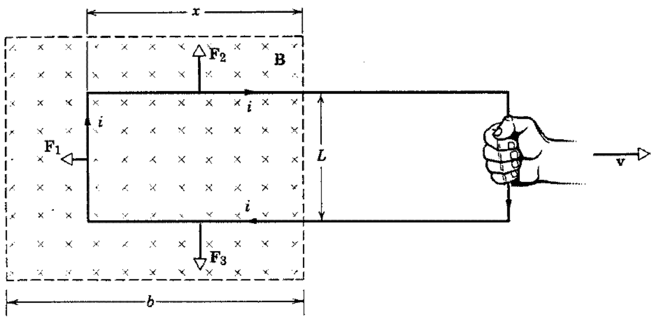


Figure A.2: The corresponding figure in Resnick and Halliday's physics text [76] for the problem described in Figure A.1.

stops moving before it exits the field, the loops stops moving exactly as it exits the field, or the loops keeps moving, with a reduced but steady velocity, after it exits the field. In the latter case, the loop continues to move to the right until it reaches a world boundary.

The equations that govern the behavior in this example are given below. The variables x and L are defined as shown in Figure A.2.

- $\mathbf{F} = i\mathbf{l} \times \mathbf{B}$, where F is the magnetic force acting on a given section $\mathbf{l}$ of the loop. The direction of the vector term $\mathbf{l}$ is the direction of current flow in that section of the loop.

- $i = V/R$, where V is the magnitude of the emf induced in the loop, and R is the resistance in the loop.
- $V = d\Phi_B/dt$, where Φ_B is the total magnetic flux passing through the loop.
- $\Phi_B = BLx$, where B represents the strength of the magnetic field.
- $d\mathbf{v}/dt = \mathbf{a} = \mathbf{F}/m$, where $\mathbf{v}$ is the velocity of the loop, $\mathbf{a}$ is the acceleration of the loop, and m is the mass of the loop.

After manipulating these equations, we can obtain:

- $V = BLv$, where v is the magnitude of the velocity of the loop.
- $F = B^2L^2v/R$, where F is the magnitude of the net magnetic force acting on the loop.
- Then, $dv/dt = -F/m = -cv$, where c is constant.

Then, since B, L, R, and m are all constant, we have that the magnitude of the net magnetic force acting on the loop is a monotonically increasing function of the velocity of the loop. Since the force acts against the motion of the loop, the derivative of the velocity of the loop is a monotonically decreasing function of the velocity of the loop.

A.2 Annotated Demonstration

First, as described in Chapter 5, we provide the problem description through the parse-figure function.

```
(quietly (setq *no-translation* nil) (setq *no-rotation* T)
        (parse-figure
                     ; input diagram
                       "pull-right-neg-fld.ps"
                     ; input text
                       '((the loop is moving out of the field))
                     ; input picture semantics description
                       'pictures-for-mag-fields
                     ; select a frame of reference for the problem
                       :optimize-FR T
                     ; output file for rotated diagram
                       :pict-out-file "figout.ps"))
```

We use the global variables *no-translation* and *no-rotation* to limit the model fragments that are considered during a simulation. For this problem, we have set *no-rotation* to be true, indicating that there is no rotation in this example. Then, the Figure Understander uses the more optimal minimum bounding rectangle as the bounding abstraction for all objects. Further, when a scenario defintion is created for QPC, we directly assert that the rotational velocity of all mobile objects is zero and constant.

Without the direct assertion that the rotational velocity of the conducting loop is constant, QPC will initially create multiple models on the assumption that perhaps rotation could take place. These assumptions will later be refuted, but due to the expense in terms of memory usage and execution time, we set the global variable `*no-rotation*` to True to prevent QPC from unnecessarily constructing models which consider the possibility of rotation.

A.2.1 Input Processing

The output of the Figure Understander is shown below.

```
************************************************************************
***                         Parsing The Figure                       ***
************************************************************************

Choosing the new frame of reference.

The old coordinate system is being rotated by 0 degrees.

Determining new vertex coordinates for all objects.

Extracting qualitative information about the figure.

Interpreting text sentence:
 (THE LOOP IS MOVING OUT OF THE FIELD)

The ATN parse of the sentence:
 `` (THE LOOP IS MOVING OUT OF THE FIELD) '' is:

(S (NP (DET THE) (NOUN LOOP))
 (VP (AUX IS) (VERB MOVING)
  (CLAUSE (AP (ADV OUT) (PP (PREP OF) (NP (DET THE) (NOUN FIELD)))))))

The text description accompanying the figure has been processed.

 The following Algernon path shall be asserted:
```

```
 Algy path =

((:THE ?V1 (MEMBER CONDUCTING-LOOPS ?V1) (MOBILITY-OF ?V1 MOBILE))
 (:THE ?V2 (MEMBER VELOCITY-VECTORS ?V2)
           (VECTOR-DIRECTION-OF ?V2 P ?VEL-VECT))
 (VELOCITY-VECTOR ?V1 P-INIT ?VEL-VECT))

Creating a postscript file for the figure after the change in the
frame of reference.

The processing of the input figure has been completed.
```

Description of the Diagram Objects

The following are the Algernon knowledge base descriptions of the conducting loop, velocity vector, and the magnetic field in the diagram after initial processing by the Figure Understander. Any assertions marked with a star (*) are those added after processing the text.

```
> (algy)
Algernon 1.2.4 of 16 December 1992.

algy> vf l (the conducting loop)
L:
  name:        ((l) nil)
  isa:         (convex-polygons nil) (mobile-objects nil)
               (convex-2-d-objects nil) (solid-objects nil)
               (field-sources nil) (conducting-loops nil)
               (objects nil) (world-objects nil) (things nil)
  number-of-vertices: (4 nil)
  input-object-type: (5 nil)
  input-object-shape: (1 nil)
  is-rectangular: (true nil)
  is-conducting-object: (true nil)
  is-solid:   (true nil)
  mobility-of: (mobile nil)
  current-flow-source:
     (non-value) (permanent nil)
  edge-of:     (l-e-4 nil) (l-e-3 nil) (l-e-2 nil) (l-e-1 nil)
  vertex-of:  (l-v-4 nil) (l-v-3 nil) (l-v-2 nil) (l-v-1 nil)
  perimeter-of: ((l-v-1 l-e-1 l-v-2 l-e-2 l-v-3 l-e-3 l-v-4 l-e-4) nil)
  connected-edges: ((l-e-3 l-v-3 l-e-4) nil) ((l-e-2 l-v-3 l-e-3) nil)
                   ((l-e-1 l-v-2 l-e-2) nil) ((l-e-4 l-v-1 l-e-1) nil)
  connected-vertices: ((l-v-4 l-e-4 l-v-1) nil) ((l-v-3 l-e-3 l-v-4) nil)
                      ((l-v-2 l-e-2 l-v-3) nil) ((l-v-1 l-e-1 l-v-2) nil)
  cog-xval:   ((410 p-init) nil)
  cog-yval:   ((1127/2 p-init) nil)
  has-bottommost-edge: ((true p-init) nil)
  bottommost-edge: ((l-e-4 p-init) nil)
  bottommost-vertex: ((l-v-1 p-init) nil) ((l-v-4 p-init) nil)
  bottommost: ((370 450 536 p-init) nil)
  has-leftmost-edge: ((true p-init) nil)
  leftmost-edge: ((l-e-1 p-init) nil)
```

```
  leftmost-vertex: ((l-v-2 p-init) nil) ((l-v-1 p-init) nil)
  leftmost:   ((370 536 591 p-init) nil)
  has-topmost-edge: ((true p-init) nil)
  topmost-edge: ((l-e-2 p-init) nil)
  topmost-vertex: ((l-v-3 p-init) nil) ((l-v-2 p-init) nil)
  topmost:    ((370 450 591 p-init) nil)
  has-rightmost-edge: ((true p-init) nil)
  rightmost-edge: ((l-e-3 p-init) nil)
  rightmost-vertex: ((l-v-4 p-init) nil) ((l-v-3 p-init) nil)
  rightmost:  ((450 536 591 p-init) nil)
  box-bounding-shape: (true nil)
*  velocity-vector: ((p-init (1 0 0)) nil)

algy> vf mv  (the vector)
Mv:
  name:        ((mv) nil)
  isa:         (velocities nil) (line-objects nil) (vectors nil)
               (velocity-vectors nil) (objects nil)
               (world-objects nil) (things nil)
  number-of-vertices: (2 nil)
  input-object-type: (12 nil)
  input-object-shape: (0 nil)
  is-rectangular: (false nil)
  vertex-of:  (mv-v-2 nil) (mv-v-1 nil)
  transformed-figure-slope: (0 nil)
  vector-direction-of: ((p (1 0 0)) nil)

algy> vf f  (the magnetic field)
F:
  name:        ((f) nil)
  isa:         (convex-polygons nil) (convex-2-d-objects nil)
               (permeable-objects nil) (immobile-objects nil)
               (magnetic-fields nil) (steady-uniform-magnetic-fields nil)
               (objects nil) (world-objects nil) (things nil)
  number-of-vertices: (4 nil)
  input-object-type: (1 nil)
  input-object-shape: (1 nil)
  is-rectangular: (true nil)
  mobility-of: (immobile nil)
  is-conducting-object: (false nil)
  field-type-of: (uniform nil) (steady nil)
  is-solid:   (false nil)
  field-direction-of: ((0 0 -1) nil)
  edge-of:    (f-e-4 nil) (f-e-3 nil) (f-e-2 nil) (f-e-1 nil)
  vertex-of:  (f-v-4 nil) (f-v-3 nil) (f-v-2 nil) (f-v-1 nil)
  perimeter-of: ((f-v-1 f-e-1 f-v-2 f-e-2 f-v-3 f-e-3 f-v-4 f-e-4) nil)
  connected-edges: ((f-e-3 f-v-3 f-e-4) nil) ((f-e-2 f-v-3 f-e-3) nil)
                   ((f-e-1 f-v-2 f-e-2) nil) ((f-e-4 f-v-1 f-e-1) nil)
  connected-vertices: ((f-v-4 f-e-4 f-v-1) nil) ((f-v-3 f-e-3 f-v-4) nil)
                      ((f-v-2 f-e-2 f-v-3) nil) ((f-v-1 f-e-1 f-v-2) nil)
  cog-xval:   ((671/2 p-init) nil)
  cog-yval:   ((564 p-init) nil)
  has-bottommost-edge: ((true p-init) nil)
  bottommost-edge: ((f-e-4 p-init) nil)
  bottommost-vertex: ((f-v-1 p-init) nil) ((f-v-4 p-init) nil)
  bottommost: ((262 409 515 p-init) nil)
  has-leftmost-edge: ((true p-init) nil)
  leftmost-edge: ((f-e-1 p-init) nil)
```

```
  leftmost-vertex: ((f-v-2 p-init) nil) ((f-v-1 p-init) nil)
  leftmost:   ((262 515 613 p-init) nil)
  has-topmost-edge: ((true p-init) nil)
  topmost-edge: ((f-e-2 p-init) nil)
  topmost-vertex: ((f-v-3 p-init) nil) ((f-v-2 p-init) nil)
  topmost:    ((262 409 613 p-init) nil)
  has-rightmost-edge: ((true p-init) nil)
  rightmost-edge: ((f-e-3 p-init) nil)
  rightmost-vertex: ((f-v-4 p-init) nil) ((f-v-3 p-init) nil)
  rightmost:  ((409 515 613 p-init) nil)
  box-bounding-shape: (true nil)
```

Description of the Relative Positions of Objects

The contents of two lists, `*initial-ordinal-relations*` and `*permanent-ordinal-relations*`, which provide the inequality relations between the X-coordinate values of the rightmost and leftmost edges and the Y-coordinate values of the topmost and bottommost edges of the bounding boxes of the conducting loop and magnetic field are given below. These are passed directly to QPC as part of the scenario description, and are used in determining the directional and topological spatial relations that are satisfied in the initial state.

```
> *initial-ordinal-relations*
((Q< (TM-YVAL L) (TM-YVAL F)) (Q< (BM-YVAL F) (BM-YVAL L))
 (Q< (BM-YVAL L) (TM-YVAL F)) (Q< (BM-YVAL F) (TM-YVAL L))
 (Q< (RM-XVAL F) (RM-XVAL L)) (Q< (LM-XVAL F) (LM-XVAL L))
 (Q< (LM-XVAL L) (RM-XVAL F)) (Q< (LM-XVAL F) (RM-XVAL L)))
> *permanent-ordinal-relations*
((Q< (BM-YVAL F) (TM-YVAL F)) (Q< (BM-YVAL L) (TM-YVAL L))
 (Q< (LM-XVAL F) (RM-XVAL F)) (Q< (LM-XVAL L) (RM-XVAL L)))
```

A.2.2 Scenario Description Provided to QPC

After the diagram and text input is processed by the Figure Understander, QPC is invoked through the `(run-qpc)` function. This function first causes a scenario (input) description to be generated for QPC, and then invokes QPC to perform the simulation. The following lists, generated as part of the scenario description for QPC, each provide a number of assertions that are true of the initial state and of the entire simulation, respectively.

```
algy> (run-qpc)
Finished finalizing initial conditions
```

Relations =

```
((Q= (X-ROT-VELOCITY L) ZERO) (Q= (Y-ROT-VELOCITY L) ZERO)
 (Q= (Z-ROT-VELOCITY L) ZERO) (Q= (PROJ-FACTOR-XY-PLANE L) ONE)
 (Q= (COS-THETA-Y L) ONE) (Q= (SIN-THETA-Y L) ZERO) (Q= (THETA-X L) ZERO)
 (Q= (THETA-Y L) (THETA-Y-0 L)) (Q< (THETA-Y L) (THETA-Y-90 L))
 (Q< (THETA-Y-90 L) (THETA-Y-180 L)) (Q> (THETA-Y L) (THETA-Y-270 L))
 (Q> (THETA-Y-0 L) ZERO) (Q> (THETA-Y-90 L) ZERO)
 (Q> (THETA-Y-180 L) ZERO) (Q> (THETA-Y-270 L) ZERO)
 (Q< (THETA-Y-0 L) MAXVAL) (Q< (THETA-Y-90 L) MAXVAL)
 (Q< (THETA-Y-180 L) MAXVAL) (Q< (THETA-Y-270 L) MAXVAL)
 (Q= (COS-THETA-X L) ONE) (Q= (SIN-THETA-X L) ZERO)
 (Q= (COS-THETA-XY L) ONE) (Q= (SIN-THETA-XY L) ZERO)
 (Q= (THETA-Z L) (THETA-Z-0 L)) (Q< (THETA-Z L) (THETA-Z-90 L))
 (Q< (THETA-Z-90 L) (THETA-Z-180 L)) (Q> (THETA-Z L) (THETA-Z-270 L))
 (Q= (Z-VELOCITY L) ZERO) (Q= (Y-VELOCITY L) ZERO) (Q> (X-VELOCITY L) ZERO)
 (Q< (TOTAL-RESISTANCE L) MAXVAL) (Q> (TOTAL-RESISTANCE L) ZERO)
 (Q> (TM-YVAL L) ZERO) (Q> (BM-YVAL L) ZERO) (Q> (RM-XVAL L) ZERO)
 (Q> (LM-XVAL L) ZERO) (Q< (TM-YVAL L) MAXVAL) (Q< (BM-YVAL L) MAXVAL)
 (Q< (RM-XVAL L) MAXVAL) (Q< (LM-XVAL L) MAXVAL) (Q> (BS-DIAMETER L) ZERO)
 (Q< (BS-DIAMETER L) MAXVAL) (Q> (TM-YVAL F) ZERO) (Q> (BM-YVAL F) ZERO)
 (Q> (RM-XVAL F) ZERO) (Q> (LM-XVAL F) ZERO) (Q< (TM-YVAL F) MAXVAL)
 (Q< (BM-YVAL F) MAXVAL) (Q< (RM-XVAL F) MAXVAL) (Q< (LM-XVAL F) MAXVAL)
 (Q> (BS-DIAMETER F) ZERO) (Q< (BS-DIAMETER F) MAXVAL)
 (Q< (TM-YVAL L) (TM-YVAL F)) (Q< (BM-YVAL F) (BM-YVAL L))
 (Q< (BM-YVAL L) (TM-YVAL F)) (Q< (BM-YVAL F) (TM-YVAL L))
 (Q< (RM-XVAL F) (RM-XVAL L)) (Q< (LM-XVAL F) (LM-XVAL L))
 (Q< (LM-XVAL L) (RM-XVAL F)) (Q< (LM-XVAL F) (RM-XVAL L)))
```

finished finalizing permanent conditions

relations =

```
((Q>= (X-VELOCITY L) ZERO) (DERIVATIVE (X-VELOCITY L) (X-ACC L))
 (DERIVATIVE (Y-VELOCITY L) (Y-ACC L))
 (DERIVATIVE (Z-VELOCITY L) (Z-ACC L))
 (DERIVATIVE (RM-XVAL L) (X-VELOCITY L))
 (DERIVATIVE (LM-XVAL L) (X-VELOCITY L))
 (DERIVATIVE (TM-YVAL L) (Y-VELOCITY L))
 (DERIVATIVE (BM-YVAL L) (Y-VELOCITY L)) (Q< (X-ROT-VELOCITY L) MAXVAL)
 (Q< (Y-ROT-VELOCITY L) MAXVAL) (Q< (Y-VELOCITY L) MAXVAL)
 (Q< (X-VELOCITY L) MAXVAL) (Q> (THETA-Y-0 L) ZERO)
 (Q> (THETA-Y-90 L) ZERO) (Q> (THETA-Y-180 L) ZERO)
 (Q> (THETA-Y-270 L) ZERO) (Q< (THETA-Y-0 L) MAXVAL)
 (Q< (THETA-Y-90 L) MAXVAL) (Q< (THETA-Y-180 L) MAXVAL)
 (Q< (THETA-Y-270 L) MAXVAL) (Q> (THETA-Y L) ZERO)
 (Q< (THETA-Y L) MAXVAL) (Q< (THETA-X L) MAXVAL)
 (Q> (THETA-Z-0 L) ZERO) (Q> (THETA-Z-90 L) ZERO)
 (Q> (THETA-Z-180 L) ZERO) (Q> (THETA-Z-270 L) ZERO)
 (Q< (THETA-Z-0 L) MAXVAL) (Q< (THETA-Z-90 L) MAXVAL)
 (Q< (THETA-Z-180 L) MAXVAL) (Q< (THETA-Z-270 L) MAXVAL)
 (Q> (THETA-Z L) ZERO) (Q< (THETA-Z L) MAXVAL)
 (DERIVATIVE (THETA-Y L) (X-ROT-VELOCITY L))
 (DERIVATIVE (THETA-X L) (Y-ROT-VELOCITY L))
 (DERIVATIVE (X-ROT-VELOCITY L) (X-ROT-ACC L))
 (DERIVATIVE (Y-ROT-VELOCITY L) (Y-ROT-ACC L))
 (DERIVATIVE (Z-ROT-VELOCITY L) (Z-ROT-ACC L))
 (Q= (THETA-Y L) (THETA-Y-0 L))
 (Q= (THETA-Z L) (THETA-Z-0 L)) (Q= (X-ROT-VELOCITY L) ZERO)
 (Q= (Y-ROT-VELOCITY L) ZERO) (Q= (Z-ROT-VELOCITY L) ZERO)
 (Q< (TOTAL-RESISTANCE L) MAXVAL) (CONSTANT (TOTAL-RESISTANCE L))
 (Q>= (TOTAL-RESISTANCE L) ZERO) (Q<= (TOTAL-RESISTANCE L) MAXVAL)
 (DERIVATIVE (FLUX-THROUGH-LOOP L) (FLUX-THROUGH-LOOP-DERIV L))
 (DERIVATIVE (LOOP-CURRENT-FLOW L) (LOOP-CURRENT-FLOW-DERIV L))
```

```
(Q>= (TM-YVAL L) ZERO) (Q>= (BM-YVAL L) ZERO) (Q>= (RM-XVAL L) ZERO)
(Q>= (LM-XVAL L) ZERO) (Q<= (TM-YVAL L) MAXVAL)
(Q<= (BM-YVAL L) MAXVAL)
(Q<= (RM-XVAL L) MAXVAL) (Q<= (LM-XVAL L) MAXVAL)
(Q>= (COS-THETA-X L) MINUS-ONE) (Q<= (COS-THETA-X L) ONE)
(Q>= (SIN-THETA-X L) MINUS-ONE) (Q<= (SIN-THETA-X L) ONE)
(Q>= (COS-THETA-Y L) MINUS-ONE) (Q<= (COS-THETA-Y L) ONE)
(Q>= (SIN-THETA-Y L) MINUS-ONE) (Q<= (SIN-THETA-Y L) ONE)
(Q>= (COS-THETA-XY L) MINUS-ONE) (Q<= (COS-THETA-XY L) ONE)
(Q>= (SIN-THETA-XY L) MINUS-ONE) (Q<= (SIN-THETA-XY L) ONE)
(Q>= (PROJ-FACTOR-XY-PLANE L) MINUS-ONE)
(Q<= (PROJ-FACTOR-XY-PLANE L) ONE)
(Q> (BS-DIAMETER L) ZERO) (Q< (BS-DIAMETER L) MAXVAL)
(CONSTANT (BS-DIAMETER L)) (CONSTANT (RM-XVAL F)) (CONSTANT (TM-YVAL F))
(CONSTANT (LM-XVAL F)) (CONSTANT (BM-YVAL F)) (CONSTANT (THETA-Y F))
(CONSTANT (THETA-X F)) (CONSTANT (COS-THETA-Y F))
(CONSTANT (SIN-THETA-Y F))
(CONSTANT (COS-THETA-X F)) (CONSTANT (SIN-THETA-X F))
(CONSTANT (COS-THETA-XY F)) (CONSTANT (SIN-THETA-XY F))
(Q= (THETA-Y F) (THETA-Y-0 F)) (Q= (COS-THETA-X F) ONE)
(Q= (SIN-THETA-X F) ZERO) (Q= (COS-THETA-Y F) ONE)
(Q= (SIN-THETA-Y F) ZERO)
(Q= (COS-THETA-XY F) ONE) (Q= (SIN-THETA-XY F) ZERO)
(Q= (PROJ-FACTOR-XY-PLANE F) ONE) (CONSTANT (PROJ-FACTOR-XY-PLANE F))
(Q< (FLD-STRENGTH F) ZERO) (CONSTANT (FLD-STRENGTH F))
(Q>= (TM-YVAL F) ZERO)
(Q>= (BM-YVAL F) ZERO) (Q>= (RM-XVAL F) ZERO) (Q>= (LM-XVAL F) ZERO)
(Q<= (TM-YVAL F) MAXVAL) (Q<= (BM-YVAL F) MAXVAL)
(Q<= (RM-XVAL F) MAXVAL)
(Q<= (LM-XVAL F) MAXVAL) (Q>= (COS-THETA-X F) MINUS-ONE)
(Q<= (COS-THETA-X F) ONE) (Q>= (SIN-THETA-X F) MINUS-ONE)
(Q<= (SIN-THETA-X F) ONE) (Q>= (COS-THETA-Y F) MINUS-ONE)
(Q<= (COS-THETA-Y F) ONE) (Q>= (SIN-THETA-Y F) MINUS-ONE)
(Q<= (SIN-THETA-Y F) ONE) (Q>= (COS-THETA-XY F) MINUS-ONE)
(Q<= (COS-THETA-XY F) ONE) (Q>= (SIN-THETA-XY F) MINUS-ONE)
(Q<= (SIN-THETA-XY F) ONE) (Q>= (PROJ-FACTOR-XY-PLANE F) MINUS-ONE)
(Q<= (PROJ-FACTOR-XY-PLANE F) ONE) (Q> (BS-DIAMETER F) ZERO)
(Q< (BS-DIAMETER F) MAXVAL) (CONSTANT (BS-DIAMETER F))
(Q> (BS-DIAMETER F) (BS-DIAMETER L))
(Q< (BM-YVAL F) (TM-YVAL F)) (Q< (BM-YVAL L) (TM-YVAL L))
(Q< (LM-XVAL F) (RM-XVAL F)) (Q< (LM-XVAL L) (RM-XVAL L)))
```

A.2.3 The First Model Proposed by QPC

The following is the output as QPC determines the model fragments that are to be included in its initial model. Because there is no direct assertion in the knowledge base that the magnetic flux through the conducting loop is increasing, decreasing, or constant during the time interval over which the initial model is active, there is a view ambiguity, and QPC executes a three-way branch to consider each of these possibilities. Thus, the initial model splits into three models.

In the output below, note that QPC first determines the relative positions of objects in the current spatial state. Views such as `Aligned-X-View` and `BB-Over-`

`lap-View` are defined in the general-purpose QPC model library we have developed to support qualitative spatial reasoning. The `Flux-Through-Loop-In-SU-Perm-NegField` view uses the fact that there is overlap between a conducting loop and a permanent magnetic field to infer that there is magnetic flux passing through the loop. This is turn triggers the possibility that the flux through the loop could be changing, hence the three-way branch.

We show two QPC views in Figure A.3 to illustrate how conclusions made by a view in the spatial model can be used by a view in the domain model. Note that the flux-through-loop view uses the fact that the overlap view must be active in its structural conditions. The flux-through-loop view also references the variable `proj-area-of-overlap`, whose value is determined by the overlap view.

```
Instantiating view ALIGNED-X-VIEW.
  L -- ABSTRACT-OBJ22
  F -- ABSTRACT-OBJ12
Instantiating view ALIGNED-X-VIEW.
  F -- ABSTRACT-OBJ22
  L -- ABSTRACT-OBJ12
Instantiating view ALIGNED-Y-VIEW.
  L -- ABSTRACT-OBJ21
  F -- ABSTRACT-OBJ11
Instantiating view ALIGNED-Y-VIEW.
  F -- ABSTRACT-OBJ21
  L -- ABSTRACT-OBJ11
Instantiating view BB-OVERLAP-VIEW.
  F -- ABSTRACT-PERM-OBJ1
  L -- ABSTRACT-SOLID-OBJ1
Instantiating view OVERLAP-BB-RIGHT-VIEW.
  F -- ABSTRACT-PERM-OBJ4
  L -- ABSTRACT-SOLID-OBJ4
Instantiating view FLUX-THROUGH-LOOP-IN-SU-PERM-NEG-FIELD.
  L -- ABSTRACT-LOOP9
  F -- ABSTRACT-MAG-FIELD1
Instantiating view NO-ROTATION.
  L -- ABSTRACT-OBJ2
Instantiating view CONSTRAINTS-ON-MOBILE-OBJECTS.
  L -- ABSTRACT-OBJ1
Instantiating view ZERO-THETA-Z.
  L -- ABSTRACT-OBJ

VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVV                                                              VVVV
VVV                        #<Model: P-INIT>                        VVV
VV                                                                  VV
V                                                                    V
Model Queue: NIL.
Model P-INIT is the initial model.
Initializing model P-INIT.
>> 1 Seconds to execute: (initialize-model model)...
QPC has been unable to determine whether 2 views should be instantiated.
  0. (increasing-flux-through-loop-neg-fld f l)
  Undetermined conditions:
    0. (q> (:quote l.flux-through-loop-deriv) (:quote zero))
```

```
(defview bb-overlap-view ()  ; from spatial reasoning model library
 :individuals
  ((solid-obj :type convex-2-d-objects
              :conditions ((mobility-of solid-obj mobile)
                           (isa solid-obj solid-objects)))
   (perm-obj :type convex-2-d-objects
             :conditions ((isa perm-obj permeable-objects)
                          (in-XY-plane perm-obj TRUE ?model)
                          (is-active-two-role-view
                           ?model 'v-aligned-X solid-obj perm-obj)
                          (is-active-two-role-view
                           ?model 'v-aligned-Y solid-obj perm-obj))))
 :relations
  ((not (q< (coplanar-area-of-overlap solid-obj perm-obj) ZERO))
   (piecewise-continuous-variable
    ?model (coplanar-area-of-overlap-deriv solid-obj perm-obj))
   (mult (cos-theta-X solid-obj) (cos-theta-Y solid-obj)
         (proj-factor-XY-plane solid-obj))
   (mult (coplanar-area-of-overlap solid-obj perm-obj)
         (proj-factor-XY-plane solid-obj)
         (proj-area-of-overlap solid-obj perm-obj))
   (is-active-two-role-view ?model 'v-bb-overlap solid-obj perm-obj)))

(defview flux-through-loop-in-SU-perm-neg-field () ; from domain model
 :individuals
  ((mag-field :type steady-uniform-magnetic-fields
              :conditions ((source-of-field mag-field permanent)))
   (loop :type conducting-loops
         :conditions ((is-active-two-role-view
                       ?model 'v-bb-overlap loop mag-field))))
 :operating-conditions ((q< (fld-strength mag-field) zero))
 :relations
  ((exists-external-flux-through-loop ?model loop mag-field)
   (mult (proj-area-of-overlap loop mag-field)
         (fld-strength mag-field)
         (flux-through-loop loop))
   (constant (fld-strength mag-field))
   (minus (proj-area-of-overlap-deriv loop mag-field)
          (flux-through-loop-deriv loop))
   (piecewise-continuous-variable ?model
                                  (flux-through-loop-deriv loop))
   (piecewise-continuous-variable
    ?model (proj-area-of-overlap-deriv loop mag-field))
   (is-active-two-role-view
    ?model 'v-flux-through-loop-in-SU-perm-field loop mag-field)))
```

Figure A.3: Examples of how views in the magnetic fields domain model can make use of assertions made by views in the spatial model.

```
  1. (decreasing-flux-through-loop-neg-fld f l)
  Undetermined conditions:
    0. (q< (:quote l.flux-through-loop-deriv) (:quote zero))
Break to examine ambiguous views? n
Branching to create 3 new models due to op-conditions-unknown.
  Created M-1 under initial assumption:
        (region-active (:slot q<) (:quote l.flux-through-loop-deriv)
                       (:quote zero))
  Created M-2 under initial assumption:
        (region-active (:slot q>) (:quote l.flux-through-loop-deriv)
                       (:quote zero))
  Created M-3 under initial assumption:
        (region-active (:slot q=) (:quote l.flux-through-loop-deriv)
                       (:quote zero))
```

A.2.4 First Branch of the First Model - Constant Flux In Loop

The first branch explores the possibility that the flux through the loop remains constant. This model is later refuted by QSIM as inconsistent. Since the loop is moving out of the field, and the strength of the field is uniform and constant, the flux through the loop cannot be constant.

```
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVV                                                              VVVV
VVV                          #<Model: M-3>                         VVV
VV                                                                  VV
V                                                                    V
Model Queue: (#<Model: M-2> #<Model: M-1>).
Model M-3 follows P-INIT because of a view ambiguity.
Initializing model M-3.
  Model M-3 is a refinement of P-INIT.
  Inheriting model assumptions:
      "[none]"
  Inheriting initial conditions assumptions:
      (region-active (:slot q=) (:quote l.flux-through-loop-deriv)
                     (:quote zero))
>> 67 Seconds to execute: (initialize-model model)...
>> 0 Seconds to execute: (perhaps-branch-on-view-ambiguity model)...
Instantiated views for M-3:
    0. (aligned-x-view l f)
    1. (aligned-x-view f l)
    2. (aligned-y-view l f)
    3. (bb-overlap-view l f)
    4. (aligned-y-view f l)
    5. (no-rotation l)
    6. (constraints-on-mobile-objects l)
    7. (zero-theta-z l)
    8. (overlap-bb-right-view l f)
    9. (flux-through-loop-in-su-perm-neg-field f l)
Asserting that the world is closed.
Resolving influences.
Inferring that all uninfluenced vars are constant.
Resolving correspondences.
Resolving quantity spaces.
  The quantity spaces are totally ordered.
  Narrowing the qspaces.
```

```
  Asserting shared quantity spaces.
Collecting region transitions.
>> 11 Seconds to execute: (resolve-model model)...
Defining a QDE.
  Cleaning up QDE for QSIM:
    every qspace must have a variable in a constraint,
    minf and inf are unreachable,
    unreachable values must be in a qspace,
    corresponding values must be in qspaces.
Building an initial state.
  Initial values:
          (dif-<f.bm-yval>-<f.tm-yval> ((minf 0) nil))
          (dif-<f.lm-xval>-<f.rm-xval> ((minf 0) nil))
          (dif-<f.bs-diameter>-<l.bs-diameter> ((0 inf) nil))
          (dif-<l.bm-yval>-<l.tm-yval> ((minf 0) nil))
          (dif-<f.bm-yval>-<l.tm-yval> ((minf 0) nil))
          (dif-<f.tm-yval>-<l.tm-yval> ((0 inf) nil))
          (dif-<f.bm-yval>-<l.bm-yval> ((minf 0) nil))
          (dif-<f.tm-yval>-<l.bm-yval> ((0 inf) nil))
          (dif-<l.lm-xval>-<l.rm-xval> ((minf 0) nil))
          (dif-<f.lm-xval>-<l.rm-xval> ((minf 0) nil))
          (dif-<f.rm-xval>-<l.rm-xval> ((minf 0) nil))
          (dif-<f.lm-xval>-<l.lm-xval> ((minf 0) nil))
          (dif-<f.rm-xval>-<l.lm-xval> ((0 inf) nil))
          (f.bm-yval ((0 maxval) nil))
          (l.tm-yval ((0 maxval) nil))
          (f.tm-yval ((0 maxval) nil))
          (l.bm-yval ((0 maxval) nil))
          (f.lm-xval ((0 maxval) nil))
          (l.rm-xval ((0 maxval) nil))
          (f.rm-xval ((0 maxval) nil))
          (l.lm-xval ((0 maxval) nil))
          (l.x-velocity ((0 maxval) nil))
          (overlap-bb-right-view.instantiation.overlap-length
           ((0 nil) nil))
          (f.fld-strength ((nil 0) nil))
          (l.y-velocity (0 nil))
          (l.theta-z-0 ((0 maxval) nil))
          (l.theta-z ((0 maxval) nil))
          (f.bs-diameter ((0 maxval) nil))
          (f.proj-factor-xy-plane (one nil))
          (f.sin-theta-xy (0 nil))
          (f.cos-theta-xy (one nil))
          (f.sin-theta-x (0 nil))
          (f.cos-theta-x (one nil))
          (f.sin-theta-y (0 nil))
          (f.cos-theta-y (one nil))
          (l.bs-diameter ((0 maxval) nil))
          (l.total-resistance ((0 maxval) nil))

Run time: 0.080 seconds to initialize S-0.
  Built initial state S-0, which is complete.
Simulating from initial state S-0.

Run time: 0.030 seconds to simulate 1 states.
  The (initial) state S-0 does not enter the region for this model.
    Fails to enter: (L.FLUX-THROUGH-LOOP-DERIV (0 NIL)).
  The initial state, S-0, is now known to be inconsistent.
  Discarding model M-3.
```

A.2.5 Second Branch of the First Model - Increasing Flux In Loop

The second branch explores the possibility that flux through the loop is increasing. At first glance, it would appear that as the loop moves out of the field, the flux through the loop should decrease. However, because the field direction is pointed into the page (negative), the total magnetic flux passing through the loop is modeled as a negative value. Then, as the loop moves out of the field, the magnitude of the flux through the loop in fact increases, since it becomes a smaller negative value.

The domain model has been carefully designed to take the signs of quantities into account. In the output below, views whose names end in a number, such as `Mag-Force-Magnitude-3`, handle a particular case. The `Mag-Force-Magnitude-3` view, for example, considers the case for the equation $F = iLB$ where current flow is positive and field strength is negative.

In the trace below, several magnetic fields phenomena are considered. First, upon determining (by assumption) that the flux through the loop is increasing, an induced emf is created in the loop, resulting in a clockwise current flow. The direction of the induced emf is determined by a Lisp function which examines the direction of the field and the direction of change in the flux through the loop, and determines the direction of current flow in the loop which would create a field within the loop that opposes the change. In the output trace, the views `CW-Induced-EMF` and `Resolve-Induced-CW-Current-Flow-In-Loop-1` model the induced current in the loop.

Once a current flow is generated in the loop, there are two effects. First, a piecewise-continuous magnetic field is dynamically created. This field (automatically named L.M-2.SOURCE-OF-FIELD, unfortunately!) has a direction pointing into the page within the loop, and a direction pointing out of the page outside the loop. We ignore self-induction in our domain model. So, in this example, the newly created magnetic field has no further effect. In the trace output, the views `Create-Field-From-Current-In-Loop` and `Properties-Of-FieldFrom-CW-`

`-Current-Flow` create the new magnetic field and model its properties.

The second effect of the current flow in the loop is that a magnetic force now acts on the loop. In the relations section of the view, `Setup-Magnetic-Force-On-Loop-In-Field-XY-Plane`, there is a call to a Lisp function which determines where the net magnetic force acts on the loop. In this example, based on the fact that the loop is rectangular in shape, is aligned with the coordinate system, and is crossing only the right boundary of a rectangular field, the Lisp function determines that the net magnetic force will act on the left edge of the loop. The force vector itself is determined by applying the qualitative cross product operation, $\mathbf{F} = i\mathbf{l} \times \mathbf{B}$, using the direction of current flow in the left edge of the loop and the direction of the magnetic field.

The net effect of the magnetic force on the acceleration of the loop is modeled by the view `Force-On-Loop`. Here, the net force on the loop is modeled as the sum of any magnetic forces acting on the loop and any permanent external forces, such as gravity, that may be acting on the loop. The external force on an object is considered to be zero unless a force vector is explicitly provided in the diagram for the problem.

QSIM produces three qualitative behaviors for this model, as expected from the discussion in the introductory paragraphs above. Of the three behaviors, only one, where the loop continues to have a non-zero velocity after exiting the magnetic field, results in a transition to a new qualitatively significant world state.

```
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVV                                                                VVVV
VVV                        #<Model: M-2>                             VVV
VV                                                                    VV
V                                                                      V
Model Queue: (#<Model: M-1>).
Model M-2 follows P-INIT because of a view ambiguity.
Initializing model M-2.
  Model M-2 is a refinement of P-INIT.
Concluded zero Q< OVERLAP-BB-RIGHT-VIEW.INSTANTIATION.OVERLAP-LENGTH.
  Inheriting model assumptions:
      "[none]"
  Inheriting initial conditions assumptions:
      (region-active (:slot q>) (:quote l.flux-through-loop-deriv)
                     (:quote zero))
Instantiating view INCREASING-FLUX-THROUGH-LOOP-NEG-FLD.
```

```
  L -- ABSTRACT-LOOP13
  F -- ABSTRACT-MAG-FIELD4
Instantiating view CW-INDUCED-EMF.
  L -- ABSTRACT-LOOP15
Instantiating view RESOLVE-INDUCED-CW-CURRENT-FLOW-IN-LOOP-1.
  L -- ABSTRACT-LOOP19
  F -- ABSTRACT-MAG-FIELD6
Instantiating view CREATE-FIELD-FROM-CURRENT-IN-LOOP.
  L -- ABSTRACT-LOOP23
Instantiating view PROPERTIES-OF-FIELD-FROM-CW-CURRENT-FLOW.
  L.M-2.SOURCE-OF-FIELD -- ABSTRACT-CF-FLD
  L -- ABSTRACT-LOOP6
Instantiating view SETUP-MAGNETIC-FORCE-ON-LOOP-IN-FIELD-XY-PLANE.
  F -- ABSTRACT-MAG-FLD
  L -- ABSTRACT-LOOP24
net-mag-force-vector on L is (-1 0 0)
Instantiating view NEG-X-DIR-MAG-FORCE-ON-LOOP.
  L -- ABSTRACT-LOOP30
Instantiating view ZERO-Y-DIR-MAG-FORCE-ON-LOOP.
  L -- ABSTRACT-LOOP34
Instantiating view ZERO-Z-DIR-MAG-FORCE-ON-LOOP.
  L -- ABSTRACT-LOOP37
Instantiating view MAG-FORCE-MAGNITUDE-3.
  F -- ABSTRACT-MAG-FLD3
  L -- ABSTRACT-LOOP27
Instantiating view FORCE-ON-LOOP.
  L -- ABSTRACT-LOOP38
>> 87 Seconds to execute: (initialize-model model)...
>> 0 Seconds to execute: (perhaps-branch-on-view-ambiguity model)...
Instantiated views for M-2:
    0. (aligned-x-view l f)
    1. (aligned-x-view f l)
    2. (aligned-y-view l f)
    3. (bb-overlap-view l f)
    4. (aligned-y-view f l)
    5. (no-rotation l)
    6. (constraints-on-mobile-objects l)
    7. (zero-theta-z l)
    8. (overlap-bb-right-view l f)
    9. (flux-through-loop-in-su-perm-neg-field f l)
   10. (increasing-flux-through-loop-neg-fld f l)
   11. (cw-induced-emf l)
   12. (create-field-from-current-in-loop l)
   13. (properties-of-field-from-cw-current-flow l l.m-2.source-of-field)
   14. (setup-magnetic-force-on-loop-in-field-xy-plane l f)
   15. (neg-x-dir-mag-force-on-loop l)
   16. (zero-y-dir-mag-force-on-loop l)
   17. (zero-z-dir-mag-force-on-loop l)
   18. (mag-force-magnitude-3 l f)
   19. (force-on-loop l)
   20. (resolve-induced-cw-current-flow-in-loop-1 f l)
Asserting that the world is closed.
Resolving influences.
Inferring that all uninfluenced vars are constant.
Resolving correspondences.
Resolving quantity spaces.
Collecting region transitions.
>> 15 Seconds to execute: (resolve-model model)...
Defining a QDE.
>> 0 Seconds to execute: (frules:run-rules 'cannonical)...
Constructing the QSIM QDE.
Building an initial state.
  Initial values:
          (dif-<f.bm-yval>-<f.tm-yval> ((minf 0) nil))
```

```
          (dif-<f.lm-xval>-<f.rm-xval> ((minf 0) nil))
          (dif-<f.bs-diameter>-<l.bs-diameter> ((0 inf) nil))
          (dif-<l.bm-yval>-<l.tm-yval> ((minf 0) nil))
          (dif-<f.bm-yval>-<l.tm-yval> ((minf 0) nil))
          (dif-<f.tm-yval>-<l.tm-yval> ((0 inf) nil))
          (dif-<f.bm-yval>-<l.bm-yval> ((minf 0) nil))
          (dif-<f.tm-yval>-<l.bm-yval> ((0 inf) nil))
          (dif-<l.lm-xval>-<l.rm-xval> ((minf 0) nil))
          (dif-<f.lm-xval>-<l.rm-xval> ((minf 0) nil))
          (dif-<f.rm-xval>-<l.rm-xval> ((minf 0) nil))
          (dif-<f.lm-xval>-<l.lm-xval> ((minf 0) nil))
          (dif-<f.rm-xval>-<l.lm-xval> ((0 inf) nil))
          (f.bm-yval ((0 maxval) nil))
          (l.tm-yval ((0 maxval) nil))
          (f.tm-yval ((0 maxval) nil))
          (l.bm-yval ((0 maxval) nil))
          (f.lm-xval ((0 maxval) nil))
          (l.rm-xval ((0 maxval) nil))
          (f.rm-xval ((0 maxval) nil))
          (l.lm-xval ((0 maxval) nil))
          (l.x-velocity ((0 maxval) nil))
          (overlap-bb-right-view.instantiation.overlap-length
           ((0 nil) nil))
          (f.fld-strength ((nil 0) nil))
          (l.y-velocity (0 nil))
          (l.theta-z-0 ((0 maxval) nil))
          (l.theta-z ((0 maxval) nil))
          (f.bs-diameter ((0 maxval) nil))
          (f.proj-factor-xy-plane (one nil))
          (f.sin-theta-xy (0 nil))
          (f.cos-theta-xy (one nil))
          (f.sin-theta-x (0 nil))
          (f.cos-theta-x (one nil))
          (f.sin-theta-y (0 nil))
          (f.cos-theta-y (one nil))
          (l.bs-diameter ((0 maxval) nil))
          (l.total-resistance ((0 maxval) nil))
          (l.net-mag-force-y (0 nil))
          (l.net-mag-force-z (0 nil))

Run time: 0.100 seconds to initialize S-1.
  Built initial state S-1, which is complete.
Simulating from initial state S-1.
Run time: 1.170 seconds to simulate 4 states.
  Generated 3 behaviors.
Display behaviors generated for the model? n
  Created model #<Model: M-4>, a transition from state S-9.
```

A.2.6 The Model After the Transition

Model M-4 is the first model in this example that is created after a transition to a new world state. Note that as QPC copies the results from the previous model into the new model, the values of the piecewise-continuous variables are not inherited.

In the new model, the only source of dynamic change is the velocity of the loop. The simulation ends once the loop reaches a world boundary ($rm(loop) =$

Maxval).

```
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVV                                                                VVVV
VVV                          #<Model: M-4>                           VVV
VV                                                                    VV
V                                                                      V
Model Queue: (#<Model: M-1>).
Model M-4 follows M-2 because of a transition.
Initializing model M-4.
  Model M-4 is a transition from M-2 via S-9.
  Inheriting model assumptions:
      "[none]"
  Inheriting initial conditions assumptions:
      (region-active (:slot q<=) (:quote f.rm-xval) (:quote l.lm-xval))
      (region-active (:slot q>=) (:quote l.lm-xval) (:quote f.rm-xval))
  Assert the qspace orderings from S-9.
  Handle non-inheritable variables and circular q-spaces.
  Assert the correspondences from S-9.
  Inheriting values from state S-9:
        (f.bm-yval ((0 maxval) nil))
        (l.tm-yval ((0 maxval) nil))
        (f.tm-yval ((0 maxval) nil))
        (l.bm-yval ((0 maxval) nil))
        (f.lm-xval ((0 maxval) nil))
        (l.rm-xval ((0 maxval) nil))
        (f.rm-xval ((0 maxval) nil))
        (l.lm-xval ((0 maxval) nil))
        (l.x-velocity ((0 maxval) nil))
        (l.coplanar-area-of-overlap (0 nil))
        (l.right-overlap-length ((minf inf) nil))
        (l.coplanar-area-of-overlap-deriv ((minf 0) nil))
        (overlap-bb-right-view.instantiation.overlap-length (0 nil))
        (f.fld-strength ((minf 0) nil))
        (l.proj-area-of-overlap-deriv ((minf 0) nil))
        (l.flux-through-loop-deriv ((0 inf) nil))
        (l.proj-area-of-overlap (0 nil))
        (l.flux-through-loop (0 nil))
        (l.y-velocity (0 nil))
        (l.theta-z-0 ((0 maxval) nil))
        (l.theta-z ((0 maxval) nil))
        (f.bs-diameter ((0 maxval) nil))
        (f.proj-factor-xy-plane (one nil))
        (f.sin-theta-xy (0 nil))
        (f.cos-theta-xy (one nil))
        (f.sin-theta-x (0 nil))
        (f.cos-theta-x (one nil))
        (f.sin-theta-y (0 nil))
        (f.cos-theta-y (one nil))
        (f.theta-x ((minf inf) nil))
        (f.theta-y ((minf inf) nil))
        (l.bs-diameter ((0 maxval) nil))
        (l.total-resistance ((0 maxval) nil))
        (l.loop-current-flow ((0 inf) nil))
        (l.induced-emf-mag ((0 inf) nil))
        (l.m-2.source-of-field.fld-strength ((0 inf) nil))
        (l.net-mag-force-x ((minf 0) nil))
        (l.mag-force-magnitude ((0 inf) nil))
        (l.net-mag-force-y (0 nil))
        (l.net-mag-force-z (0 nil))
        (mag-force-magnitude-3.instantiation.temp-var ((minf 0) nil))
```

```
        (l.net-force-x ((minf 0) nil))
        (l.net-force-y (0 nil))
        (l.net-force-z (0 nil))
        (l.x-acc ((minf 0) nil))
        (l.y-acc (0 nil))
        (l.z-acc (0 nil))
ignoring value for piecewise-continuous variable
 L.M-2.SOURCE-OF-FIELD.FLD-STRENGTH
ignoring value for piecewise-continuous variable
 L.COPLANAR-AREA-OF-OVERLAP-DERIV
ignoring value for piecewise-continuous variable
 L.PROJ-AREA-OF-OVERLAP-DERIV
ignoring value for piecewise-continuous variable L.FLUX-THROUGH-LOOP-DERIV
ignoring value for piecewise-continuous variable L.LOOP-CURRENT-FLOW
ignoring value for piecewise-continuous variable L.INDUCED-EMF-MAG
ignoring value for piecewise-continuous variable L.NET-MAG-FORCE-X
ignoring value for piecewise-continuous variable L.NET-MAG-FORCE-Y
ignoring value for piecewise-continuous variable L.NET-MAG-FORCE-Z
ignoring value for piecewise-continuous variable L.NET-FORCE-X
ignoring value for piecewise-continuous variable L.NET-FORCE-Y
ignoring value for piecewise-continuous variable L.NET-FORCE-Z
ignoring value for piecewise-continuous variable L.X-ACC
ignoring value for piecewise-continuous variable L.Y-ACC
ignoring value for piecewise-continuous variable L.Z-ACC
Instantiating view LOOP-RIGHT-OF-PERM-FIELD.
  F -- ABSTRACT-LEFT-OBJ2
  L -- ABSTRACT-RIGHT-OBJ2
active-view M-4 V-LOOP-RIGHT-OF-PERM-FIELD L F
Instantiating view LEFT-OF-VIEW.
  L -- ABSTRACT-RIGHT-OBJ
  F -- ABSTRACT-LEFT-OBJ
>> 63 Seconds to execute: (initialize-model model)...
>> 0 Seconds to execute: (perhaps-branch-on-view-ambiguity model)...
Instantiated views for M-4:
    0. (loop-right-of-perm-field l f)
    1. (left-of-view f l)
    2. (aligned-x-view f l)
    3. (aligned-x-view l f)
    4. (no-rotation l)
    5. (constraints-on-mobile-objects l)
    6. (zero-theta-z l)
Asserting that the world is closed.
Resolving influences.
Inferring that all uninfluenced vars are constant.
Resolving correspondences.
Resolving quantity spaces.
  The quantity spaces are totally ordered.
  Narrowing the qspaces.
  Asserting shared quantity spaces.
Collecting region transitions.
>> 6 Seconds to execute: (resolve-model model)...
Defining a QDE.
  Cleaning up QDE for QSIM:
    every qspace must have a variable in a constraint,
    minf and inf are unreachable,
    unreachable values must be in a qspace,
    corresponding values must be in qspaces.
>> 0 Seconds to execute: (frules:run-rules 'cannonical)...
Constructing the QSIM QDE.
Building an initial state.
  Initial values:
          (dif-<f.bm-yval>-<f.tm-yval> ((minf 0) nil))
          (dif-<f.lm-xval>-<f.rm-xval> ((minf 0) nil))
          (dif-<f.bs-diameter>-<l.bs-diameter> ((0 inf) nil))
          (dif-<l.lm-xval>-<l.rm-xval> ((minf 0) nil))
          (dif-<f.lm-xval>-<l.rm-xval> ((minf 0) nil))
```

```
          (dif-<f.rm-xval>-<l.rm-xval> ((minf 0) nil))
          (dif-<l.bm-yval>-<l.tm-yval> ((minf 0) nil))
          (dif-<f.bm-yval>-<l.tm-yval> ((minf 0) nil))
          (dif-<f.tm-yval>-<l.tm-yval> ((0 inf) nil))
          (dif-<f.bm-yval>-<l.bm-yval> ((minf 0) nil))
          (dif-<f.tm-yval>-<l.bm-yval> ((0 inf) nil))
          (dif-<f.lm-xval>-<l.lm-xval> ((minf 0) nil))
          (dif-<f.rm-xval>-<l.lm-xval> (0 nil))
          (f.bs-diameter ((0 maxval) nil))
          (f.fld-strength ((nil 0) nil))
          (f.proj-factor-xy-plane (one nil))
          (f.sin-theta-xy (0 nil))
          (f.cos-theta-xy (one nil))
          (f.sin-theta-x (0 nil))
          (f.cos-theta-x (one nil))
          (f.sin-theta-y (0 nil))
          (f.cos-theta-y (one nil))
          (f.bm-yval ((0 maxval) nil))
          (f.lm-xval ((0 maxval) nil))
          (f.tm-yval ((0 maxval) nil))
          (f.rm-xval ((0 maxval) nil))
          (l.bs-diameter ((0 maxval) nil))
          (l.total-resistance ((0 maxval) nil))
          (l.tm-yval ((0 maxval) nil))
          (l.bm-yval ((0 maxval) nil))
          (l.lm-xval ((0 maxval) nil))
          (l.rm-xval ((0 maxval) nil))
          (l.y-velocity (0 nil))
          (l.x-velocity ((0 maxval) nil))
          (l.theta-z-0 ((0 maxval) nil))
          (l.theta-z ((0 maxval) nil))

Run time: 0.060 seconds to initialize S-12.
  Built initial state S-12, which is complete.
Simulating from initial state S-12.

Variables in M-4 likely to chatter are NONE.
Run time: 0.210 seconds to simulate 3 states.
  Generated 1 behavior.
Display behaviors generated for the model? n
[16:32:31]QPC: Transition state S-14 satisfies no test.
```

A.2.7 Third Branch of the First Model - Decreasing Flux In Loop

QPC performs the qualitative simulation in a depth first manner. Therefore, the third branch of the first model is considered last. Since this model was created with the incorrect assumption that the flux through the loop is decreasing, it is refuted as inconsistent by QSIM.

```
VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV
VVVV                                                               VVVV
VVV                           #<Model: M-1>                         VVV
VV                                                                   VV
V                                                                     V
Model Queue: NIL.
Model M-1 follows P-INIT because of a view ambiguity.
Initializing model M-1.
```

```
  Model M-1 is a refinement of P-INIT.
  Inheriting model assumptions:
      "[none]"
  Inheriting initial conditions assumptions:
      (region-active (:slot q<) (:quote l.flux-through-loop-deriv)
                     (:quote zero))
Instantiating view DECREASING-FLUX-THROUGH-LOOP-NEG-FLD.
  L -- ABSTRACT-LOOP14
  F -- ABSTRACT-MAG-FIELD5
retrieved value for field dir of perm fld is (0 0 -1)
Instantiating view CCW-INDUCED-EMF.
  L -- ABSTRACT-LOOP16
Instantiating view PROPERTIES-OF-FIELD-FROM-CCW-CURRENT-FLOW.
  L.M-1.SOURCE-OF-FIELD -- ABSTRACT-CF-FLD1
  L -- ABSTRACT-LOOP7
loop-XY-aligned is T
Instantiating view POS-X-DIR-MAG-FORCE-ON-LOOP.
  L -- ABSTRACT-LOOP29
net-mag-force-vector on L is (1 0 0)
Instantiating view MAG-FORCE-MAGNITUDE-2.
  F -- ABSTRACT-MAG-FLD2
  L -- ABSTRACT-LOOP26
Instantiating view RESOLVE-INDUCED-CCW-CURRENT-FLOW-IN-LOOP-2.
  L -- ABSTRACT-LOOP22
  F -- ABSTRACT-MAG-FIELD9
>> 86 Seconds to execute: (initialize-model model)...
>> 0 Seconds to execute: (perhaps-branch-on-view-ambiguity model)...
Instantiated views for M-1:
    0. (aligned-x-view l f)
    1. (aligned-x-view f l)
    2. (aligned-y-view l f)
    3. (bb-overlap-view l f)
    4. (aligned-y-view f l)
    5. (no-rotation l)
    6. (constraints-on-mobile-objects l)
    7. (zero-theta-z l)
    8. (overlap-bb-right-view l f)
    9. (flux-through-loop-in-su-perm-neg-field f l)
   10. (decreasing-flux-through-loop-neg-fld f l)
   11. (ccw-induced-emf l)
   12. (create-field-from-current-in-loop l)
   13. (properties-of-field-from-ccw-current-flow l l.m-1.source-of-field)
   14. (setup-magnetic-force-on-loop-in-field-xy-plane l f)
   15. (pos-x-dir-mag-force-on-loop l)
   16. (zero-y-dir-mag-force-on-loop l)
   17. (zero-z-dir-mag-force-on-loop l)
   18. (mag-force-magnitude-2 l f)
   19. (force-on-loop l)
   20. (resolve-induced-ccw-current-flow-in-loop-2 f l)
Asserting that the world is closed.
Resolving influences.
Inferring that all uninfluenced vars are constant.
Resolving correspondences.
Resolving quantity spaces.
Collecting region transitions.
>> 15 Seconds to execute: (resolve-model model)...
Defining a QDE.
  Cleaning up QDE for QSIM:
Constructing the QSIM QDE.
Building an initial state.
  Initial values:
        (dif-<f.bm-yval>-<f.tm-yval> ((minf 0) nil))
        (dif-<f.lm-xval>-<f.rm-xval> ((minf 0) nil))
```

```
        (dif-<f.bs-diameter>-<l.bs-diameter> ((0 inf) nil))
        (dif-<l.bm-yval>-<l.tm-yval> ((minf 0) nil))
        (dif-<f.bm-yval>-<l.tm-yval> ((minf 0) nil))
        (dif-<f.tm-yval>-<l.tm-yval> ((0 inf) nil))
        (dif-<f.bm-yval>-<l.bm-yval> ((minf 0) nil))
        (dif-<f.tm-yval>-<l.bm-yval> ((0 inf) nil))
        (dif-<l.lm-xval>-<l.rm-xval> ((minf 0) nil))
        (dif-<f.lm-xval>-<l.rm-xval> ((minf 0) nil))
        (dif-<f.rm-xval>-<l.rm-xval> ((minf 0) nil))
        (dif-<f.lm-xval>-<l.lm-xval> ((minf 0) nil))
        (dif-<f.rm-xval>-<l.lm-xval> ((0 inf) nil))
        (f.bm-yval ((0 maxval) nil))
        (l.tm-yval ((0 maxval) nil))
        (f.tm-yval ((0 maxval) nil))
        (l.bm-yval ((0 maxval) nil))
        (f.lm-xval ((0 maxval) nil))
        (l.rm-xval ((0 maxval) nil))
        (f.rm-xval ((0 maxval) nil))
        (l.lm-xval ((0 maxval) nil))
        (l.x-velocity ((0 maxval) nil))
        (overlap-bb-right-view.instantiation.overlap-length
         ((0 nil) nil))
        (f.fld-strength ((nil 0) nil))
        (l.y-velocity (0 nil))
        (l.theta-z-0 ((0 maxval) nil))
        (l.theta-z ((0 maxval) nil))
        (f.bs-diameter ((0 maxval) nil))
        (f.proj-factor-xy-plane (one nil))
        (f.sin-theta-xy (0 nil))
        (f.cos-theta-xy (one nil))
        (f.sin-theta-x (0 nil))
        (f.cos-theta-x (one nil))
        (f.sin-theta-y (0 nil))
        (f.cos-theta-y (one nil))
        (l.bs-diameter ((0 maxval) nil))
        (l.total-resistance ((0 maxval) nil))
        (l.net-mag-force-y (0 nil))
        (l.net-mag-force-z (0 nil))

Run time: 0.040 seconds to initialize S-15.
  The initial state, S-15, is inconsistent.
  Discarding model M-1.
  Exiting QPC
  P
```

A.2.8 The Behaviors Produced by Qualitative Simulation

Figures A.5, A.6, and A.7 show the behaviors produced for this example. Figure A.4 shows the tree of all behaviors. Figure A.5 shows the case where the conducting loop exits the magnetic field, and proceeds to travel to the right until it reaches the world boundary. This is shown by the value of `Maxval` for the variable `l.rm-xval`. The velocity of the loop (`l.x-velocity`) is decreasing since the net magnetic force acting on the loop (`l.net-mag-force-x`) is negative. Once the loop exits the field

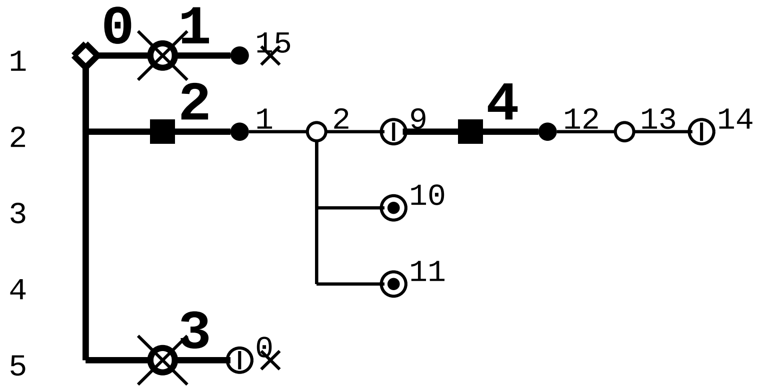


Figure A.4: Tree of all behaviors produced for the problem given in Figure A.1.

(when `l.coplanar-area-of-overlap` becomes zero), the velocity remains steady

The transition condition for detecting that the loop has left the field is when the X-coordinate value of the rightmost edge of the loop equals that of the left-most edge of the field. This inequality constraint is modeled by the QPC variable DIF-<F.RM-XVAL>-<L.LM-XVAL>. The transition occurs when the value of this variable becomes zero.

As the loop moves out of the field, the flux through the loop decreases. This causes an induced emf in the loop which results in a clockwise current flow. This is shown by the positive values for the variables `l.induced-emf-mag` and `l.loop-current-flow`. Since the net magnetic force acting on the loop (`l.net-mag-force-x`) is negative, the velocity of the loop decreases. Therefore, the rate of change in the flux through the loop decreases, and in turn, the magnitude of the induced current flow decreases. Finally, note that the flux through the loop (`l.flux-through-loop`) is negative and increasing toward zero. This is due to the fact that the magnetic field is pointed into the page.

The behaviors in Figures A.6 and A.7 are similar. Figure 2 shows the case where the loop comes to a stop before it exits the field, and Figure 3 shows the case where the loop comes to a stop just as it exits the field.

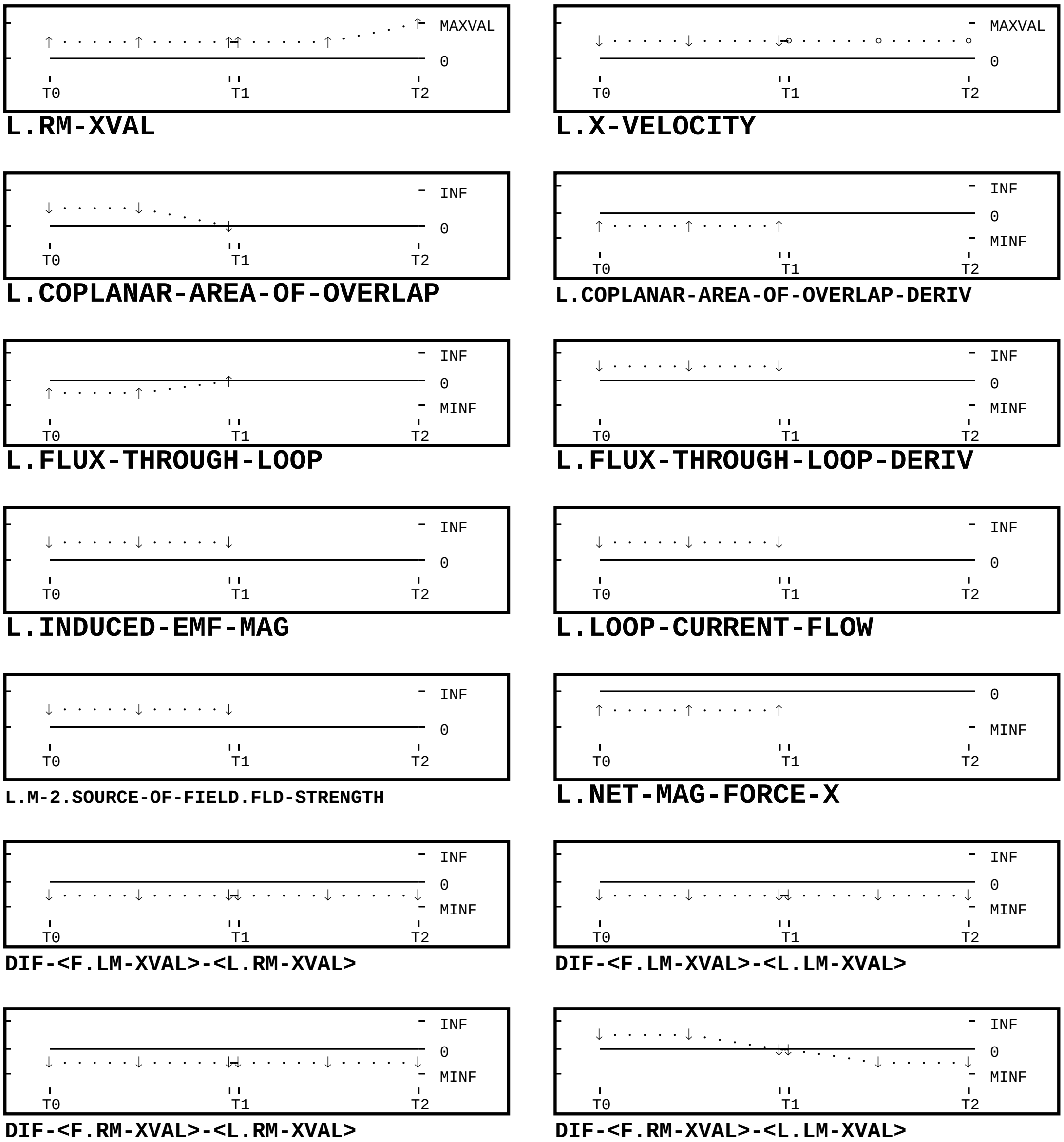


Figure A.5: First behavior for the problem given in Figure A.1.

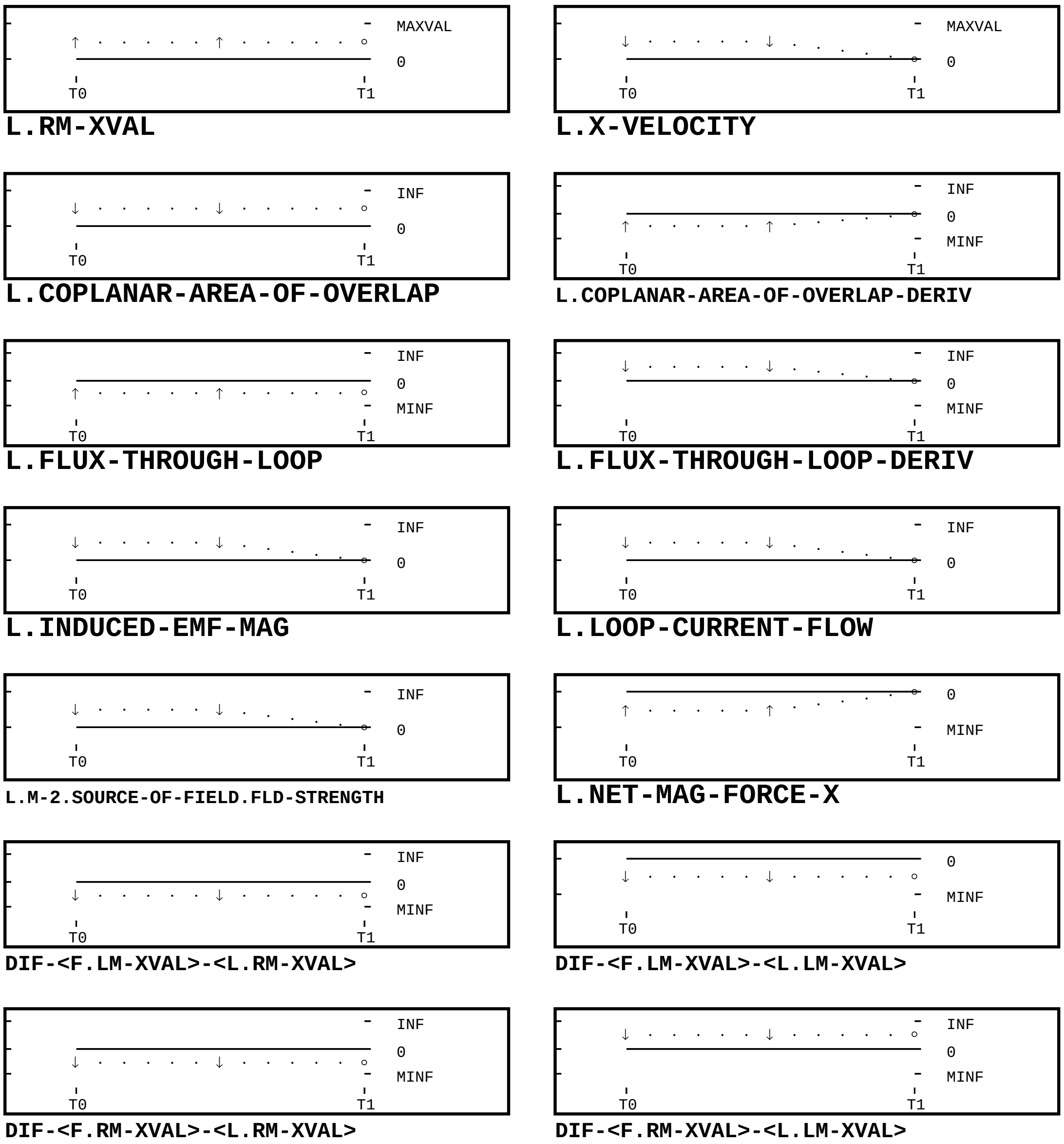


Figure A.6: Second behavior for the problem given in Figure A.1.

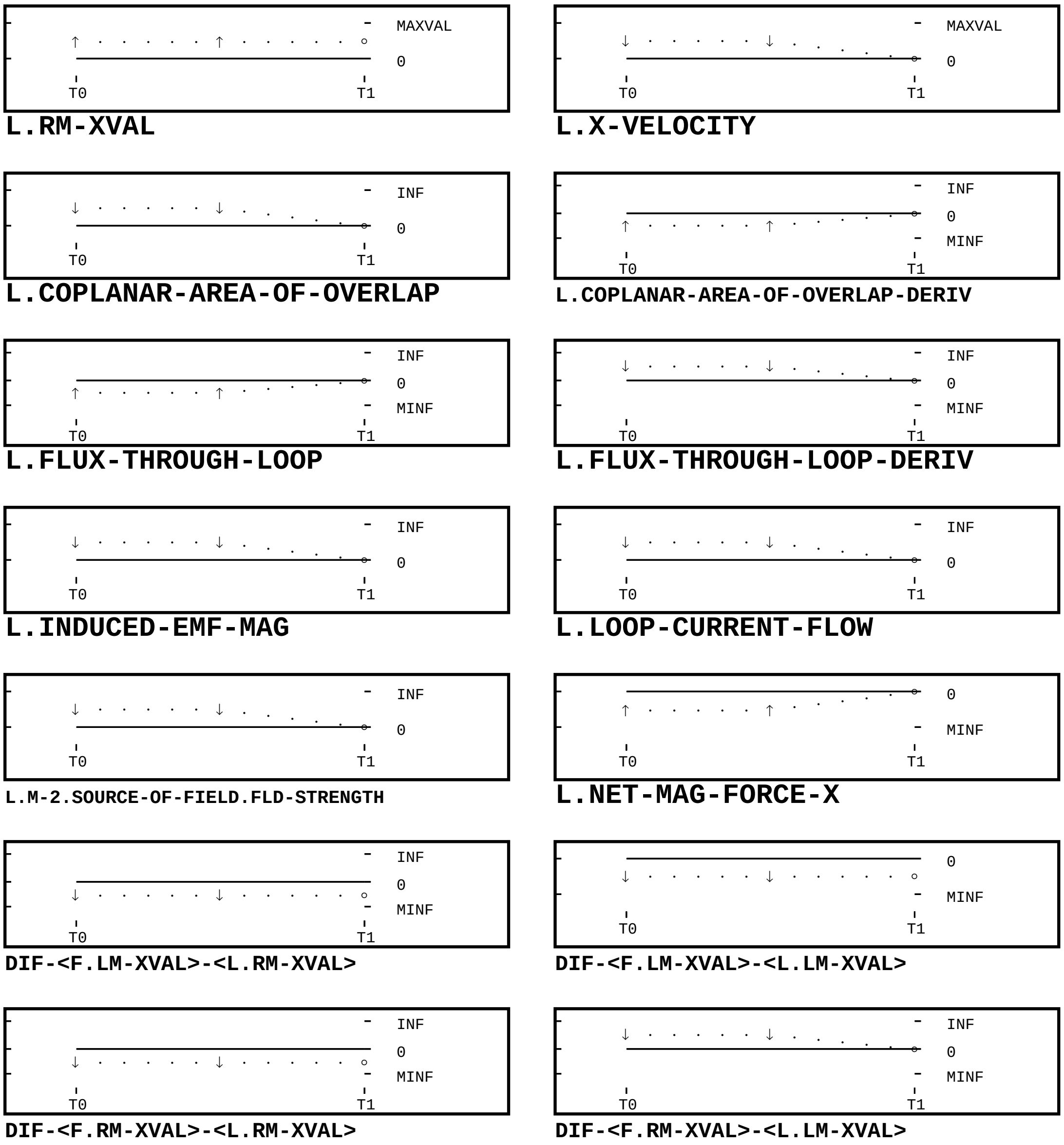


Figure A.7: Third behavior for the problem given in Figure A.1.

A.2.9 The Final KB Descriptions of the Loop and Field

This section presents the final KB descriptions for the conducting loop, permanent magnetic field, and the piecewise-continuous magnetic field created in model M-2. Only new assertions about these objects are shown, and many irrelevant assertions have been omitted. A subset of the quantities (model variables) created by QPC to model the properties of the conducting loop are also shown.

```
algy> vf l
L: [slot]
  name:         ((l) nil)
  in-xy-plane: ((true m-1) nil) ((true m-4) nil) ((true m-2) nil)
               ((true m-3) nil) ((true p-init) nil)
  exists-force-on-mob-obj: ((m-1 true) nil) ((m-2 true) nil)
  loop-current-flow-direction: ((ccw m-1) nil) ((cw m-2) nil)
  exists-current-flow: ((true m-1) nil) ((true m-2) nil)
  induced-emf-dir: ((ccw m-1) nil) ((cw m-2) nil)
  net-mag-force-vector: ((m-1 (1 0 0)) nil) ((m-2 (-1 0 0)) nil)
     (non-values) ((m-1 (0 0 0)) nil) ((m-2 (0 0 0)) nil)
  exists-mag-force-on-mob-obj: ((m-1 true) nil) ((m-2 true) nil)

; subset of the QPC model variables (quantities) involving
; properties of the conducting loop.
  x-velocity: (l.x-velocity nil)
  tm-yval:    (l.tm-yval nil)
  bm-yval:    (l.bm-yval nil)
  rm-xval:    (l.rm-xval nil)
  lm-xval:    (l.lm-xval nil)
  x-acc:      (l.x-acc nil)
  flux-through-loop-deriv: (l.flux-through-loop-deriv nil)
  flux-through-loop: (l.flux-through-loop nil)
  induced-emf-mag: (l.induced-emf-mag nil)
  loop-current-flow-deriv: (l.loop-current-flow-deriv nil)
  loop-current-flow: (l.loop-current-flow nil)
  coplanar-area-of-overlap: ((f l.coplanar-area-of-overlap) nil)
  proj-factor-xy-plane: (l.proj-factor-xy-plane nil)
  proj-area-of-overlap: ((f l.proj-area-of-overlap) nil)
  proj-area-of-overlap-deriv: ((f l.proj-area-of-overlap-deriv) nil)
  coplanar-area-of-overlap-deriv:((f l.coplanar-area-of-overlap-deriv)nil)
  right-overlap-length: ((f l.right-overlap-length) nil)
  left-overlap-length: ((f l.left-overlap-length) nil)
  top-overlap-length: ((f l.top-overlap-length) nil)
  bottom-overlap-length: ((f l.bottom-overlap-length) nil)
  net-force-x: (l.net-force-x nil)
  mag-force-magnitude: (l.mag-force-magnitude nil)
  net-mag-force-x: (l.net-mag-force-x nil)

algy> vf f   ; shows only newly added properties.
F: [slot]

  name:         ((f) nil)
  in-xy-plane: ((true m-1) nil) ((true m-4) nil) ((true m-2) nil)
                ((true m-3) nil) ((true p) nil) ((true p-init) nil)

; dynamically created magnetic field in Model M-2.
```

```
algy> vf l.m-2.source-of-field
L.m-2.source-of-field:
  isa:        (l.m-2.source-of-field-selfset nil)
              (current-flow-magnetic-fields nil) (permeable-objects nil)
              (things nil) (objects nil) (world-objects nil)
              (immobile-objects nil) (magnetic-fields nil)
  in-xy-plane: ((true m-2) nil)
```

A.2.10 The QSIM QDE Produced by QPC for model M-2

This section presents the QSIM QDE (qualitative differential equation) produced for model M-2 by QPC. This QDE, and the initial values shown in trace output for model M-2, are used by QSIM to perform the qualitative simulation. The variables whose names begin with 'DIF' are created by QPC to model inequality relations between model variables.

```
algy> (show-qde (car (ask '((model-structure m-2 ?x)) :collect '?x)))
(DEFINE-QDE
 M-2
 (QUANTITY-SPACES
  (DIF-<F.RM-XVAL>-<L.LM-XVAL> (MINF 0 INF))
  (DIF-<F.RM-XVAL>-<L.RM-XVAL> (MINF 0 INF))
  (DIF-<F.LM-XVAL>-<L.RM-XVAL> (MINF 0 INF))
  (DIF-<F.TM-YVAL>-<L.BM-YVAL> (MINF 0 INF))
  (DIF-<F.TM-YVAL>-<L.TM-YVAL> (MINF 0 INF))
  (DIF-<F.BM-YVAL>-<L.TM-YVAL> (MINF 0 INF))
  (DIF-<F.BM-YVAL>-<L.BM-YVAL> (MINF 0 INF))
  (DIF-<L.THETA-Z>-<L.THETA-Z-0> (0))
  (DIF-<L.RIGHT-OVERLAP-LENGTH>-
       <OVERLAP-BB-RIGHT-VIEW.INSTANTIATION.OVERLAP-LENGTH> (MINF 0 INF))
  (DIF-<L.COPLANAR-AREA-OF-OVERLAP>-<L.PROJ-AREA-OF-OVERLAP> (MINF 0 INF))
  (DIF-<F.BS-DIAMETER>-<L.BS-DIAMETER> (0 INF))
  (DIF-<L.LM-XVAL>-<L.RM-XVAL> (MINF 0))
  (DIF-<F.LM-XVAL>-<L.LM-XVAL> (MINF 0))
  (DIF-<F.LM-XVAL>-<F.RM-XVAL> (MINF 0))
  (DIF-<L.BM-YVAL>-<L.TM-YVAL> (MINF 0))
  (DIF-<F.BM-YVAL>-<F.TM-YVAL> (MINF 0)) (F.BM-YVAL (0 MAXVAL))
  (L.TM-YVAL (0 MAXVAL)) (F.TM-YVAL (0 MAXVAL))
  (L.BM-YVAL (0 MAXVAL)) (F.LM-XVAL (0 MAXVAL))
  (L.RM-XVAL (0 MAXVAL)) (F.RM-XVAL (0 MAXVAL))
  (L.LM-XVAL (0 MAXVAL)) (L.X-VELOCITY (0 MAXVAL))
  (L.COPLANAR-AREA-OF-OVERLAP (0 INF))
  (L.RIGHT-OVERLAP-LENGTH (MINF INF))
  (L.COPLANAR-AREA-OF-OVERLAP-DERIV (MINF 0 INF))
  (OVERLAP-BB-RIGHT-VIEW.INSTANTIATION.OVERLAP-LENGTH (0 INF))
  (F.FLD-STRENGTH (MINF 0))
  (L.PROJ-AREA-OF-OVERLAP-DERIV (MINF 0 INF))
  (L.FLUX-THROUGH-LOOP-DERIV (MINF 0 INF))
  (L.PROJ-AREA-OF-OVERLAP (MINF 0 INF))
  (L.FLUX-THROUGH-LOOP (MINF 0 INF)) (L.Y-VELOCITY (MINF 0 MAXVAL))
  (L.THETA-Z-0 (0 MAXVAL)) (L.THETA-Z (0 MAXVAL))
  (F.BS-DIAMETER (0 MAXVAL)) (F.PROJ-FACTOR-XY-PLANE (0 ONE))
  (F.SIN-THETA-XY (MINUS-ONE 0 ONE)) (F.COS-THETA-XY (0 ONE))
  (F.SIN-THETA-X (MINUS-ONE 0 ONE)) (F.COS-THETA-X (0 ONE))
  (F.SIN-THETA-Y (MINUS-ONE 0 ONE)) (F.COS-THETA-Y (0 ONE))
  (F.THETA-X (MINF INF)) (F.THETA-Y (MINF INF))
  (L.BS-DIAMETER (0 MAXVAL)) (L.TOTAL-RESISTANCE (0 MAXVAL))
  (L.LOOP-CURRENT-FLOW (0 INF)) (L.INDUCED-EMF-MAG (0 INF))
  (L.M-2.SOURCE-OF-FIELD.FLD-STRENGTH (0 INF))
```

```
 (L.NET-MAG-FORCE-X (MINF 0)) (L.MAG-FORCE-MAGNITUDE (MINF 0 INF))
 (L.NET-MAG-FORCE-Y (MINF 0 INF)) (L.NET-MAG-FORCE-Z (MINF 0 INF))
 (MAG-FORCE-MAGNITUDE-3.INSTANTIATION.TEMP-VAR (MINF 0 INF))
 (L.NET-FORCE-X (MINF 0 INF)) (L.NET-FORCE-Y (MINF 0 INF))
 (L.NET-FORCE-Z (MINF 0 INF)) (L.X-ACC (MINF 0 INF))
 (L.Y-ACC (MINF 0 INF)) (L.Z-ACC (MINF 0 INF)))
(CONSTRAINTS
 ((ADD L.RIGHT-OVERLAP-LENGTH DIF-<F.LM-XVAL>-<F.RM-XVAL>
   DIF-<F.LM-XVAL>-<L.LM-XVAL>))
 ((ADD DIF-<F.TM-YVAL>-<L.TM-YVAL> DIF-<F.BM-YVAL>-<F.TM-YVAL>
   DIF-<F.BM-YVAL>-<L.TM-YVAL>))
 ((ADD DIF-<L.BM-YVAL>-<L.TM-YVAL> DIF-<F.BM-YVAL>-<L.BM-YVAL>
   DIF-<F.BM-YVAL>-<L.TM-YVAL>))
 ((ADD DIF-<L.BM-YVAL>-<L.TM-YVAL> DIF-<F.TM-YVAL>-<L.BM-YVAL>
   DIF-<F.TM-YVAL>-<L.TM-YVAL>))
 ((ADD DIF-<F.TM-YVAL>-<L.BM-YVAL> DIF-<F.BM-YVAL>-<F.TM-YVAL>
   DIF-<F.BM-YVAL>-<L.BM-YVAL>))
 ((ADD DIF-<F.RM-XVAL>-<L.RM-XVAL> DIF-<F.LM-XVAL>-<F.RM-XVAL>
   DIF-<F.LM-XVAL>-<L.RM-XVAL>))
 ((ADD DIF-<L.LM-XVAL>-<L.RM-XVAL> DIF-<F.LM-XVAL>-<L.LM-XVAL>
   DIF-<F.LM-XVAL>-<L.RM-XVAL>))
 ((ADD DIF-<L.LM-XVAL>-<L.RM-XVAL>
   OVERLAP-BB-RIGHT-VIEW.INSTANTIATION.OVERLAP-LENGTH
   DIF-<F.RM-XVAL>-<L.RM-XVAL>))
 ((ADD DIF-<L.LM-XVAL>-<L.RM-XVAL> DIF-<F.RM-XVAL>-<L.LM-XVAL>
   DIF-<F.RM-XVAL>-<L.RM-XVAL>))
 ((ADD DIF-<F.RM-XVAL>-<L.LM-XVAL> DIF-<F.LM-XVAL>-<F.RM-XVAL>
   DIF-<F.LM-XVAL>-<L.LM-XVAL>))
 ((ADD F.TM-YVAL DIF-<F.BM-YVAL>-<F.TM-YVAL> F.BM-YVAL)
  (MAXVAL 0 MAXVAL))
 ((ADD F.RM-XVAL DIF-<F.LM-XVAL>-<F.RM-XVAL> F.LM-XVAL)
  (MAXVAL 0 MAXVAL))
 ((ADD L.BS-DIAMETER DIF-<F.BS-DIAMETER>-<L.BS-DIAMETER>
   F.BS-DIAMETER)
  (MAXVAL 0 MAXVAL))
 ((ADD L.TM-YVAL DIF-<L.BM-YVAL>-<L.TM-YVAL> L.BM-YVAL)
  (MAXVAL 0 MAXVAL))
 ((ADD L.TM-YVAL DIF-<F.BM-YVAL>-<L.TM-YVAL> F.BM-YVAL)
  (MAXVAL 0 MAXVAL))
 ((ADD L.TM-YVAL DIF-<F.TM-YVAL>-<L.TM-YVAL> F.TM-YVAL)
  (MAXVAL 0 MAXVAL))
 ((ADD L.BM-YVAL DIF-<F.BM-YVAL>-<L.BM-YVAL> F.BM-YVAL)
  (MAXVAL 0 MAXVAL))
 ((ADD L.BM-YVAL DIF-<F.TM-YVAL>-<L.BM-YVAL> F.TM-YVAL)
  (MAXVAL 0 MAXVAL))
 ((ADD L.RM-XVAL DIF-<L.LM-XVAL>-<L.RM-XVAL> L.LM-XVAL)
  (MAXVAL 0 MAXVAL))
 ((ADD L.RM-XVAL DIF-<F.LM-XVAL>-<L.RM-XVAL> F.LM-XVAL)
  (MAXVAL 0 MAXVAL))
 ((ADD L.RM-XVAL DIF-<F.RM-XVAL>-<L.RM-XVAL> F.RM-XVAL)
  (MAXVAL 0 MAXVAL))
 ((ADD L.LM-XVAL OVERLAP-BB-RIGHT-VIEW.INSTANTIATION.OVERLAP-LENGTH
   F.RM-XVAL)
  (MAXVAL 0 MAXVAL))
 ((ADD L.LM-XVAL DIF-<F.LM-XVAL>-<L.LM-XVAL> F.LM-XVAL)
  (MAXVAL 0 MAXVAL))
 ((ADD L.LM-XVAL DIF-<F.RM-XVAL>-<L.LM-XVAL> F.RM-XVAL)
  (MAXVAL 0 MAXVAL))
 ((CONSTANT DIF-<F.BM-YVAL>-<F.TM-YVAL>))
 ((CONSTANT DIF-<F.LM-XVAL>-<F.RM-XVAL>))
 ((CONSTANT DIF-<F.BS-DIAMETER>-<L.BS-DIAMETER>))
 ((CONSTANT DIF-<L.BM-YVAL>-<L.TM-YVAL>))
 ((CONSTANT DIF-<L.LM-XVAL>-<L.RM-XVAL>)) ((CONSTANT F.RM-XVAL))
 ((CONSTANT F.FLD-STRENGTH)) ((CONSTANT F.BS-DIAMETER))
 ((CONSTANT F.THETA-X)) ((CONSTANT F.THETA-Y))
 ((CONSTANT F.BM-YVAL)) ((CONSTANT F.LM-XVAL))
 ((CONSTANT F.TM-YVAL)) ((CONSTANT L.BS-DIAMETER))
 ((CONSTANT L.TOTAL-RESISTANCE)) ((CONSTANT L.THETA-Z-0))
```

```
 ((CONSTANT L.THETA-Z)) ((CONSTANT L.NET-FORCE-Y))
 ((CONSTANT L.NET-FORCE-Z)) ((CONSTANT L.Y-ACC))
 ((CONSTANT L.Z-ACC)) ((CONSTANT L.Y-VELOCITY 0))
 ((CONSTANT F.PROJ-FACTOR-XY-PLANE ONE))
 ((CONSTANT F.SIN-THETA-XY 0)) ((CONSTANT F.COS-THETA-XY ONE))
 ((CONSTANT F.SIN-THETA-X 0)) ((CONSTANT F.COS-THETA-X ONE))
 ((CONSTANT F.SIN-THETA-Y 0)) ((CONSTANT F.COS-THETA-Y ONE))
 ((CONSTANT L.NET-MAG-FORCE-Y 0)) ((CONSTANT L.NET-MAG-FORCE-Z 0))
 ((CONSTANT
   DIF-<L.COPLANAR-AREA-OF-OVERLAP>-<L.PROJ-AREA-OF-OVERLAP> 0))
 ((CONSTANT
   DIF-<L.RIGHT-OVERLAP-LENGTH>-
       <OVERLAP-BB-RIGHT-VIEW.INSTANTIATION.OVERLAP-LENGTH> 0))
 ((CONSTANT DIF-<L.THETA-Z>-<L.THETA-Z-0> 0))
 (((M -) L.MAG-FORCE-MAGNITUDE L.NET-MAG-FORCE-X) (0 0))
 (((M -) L.X-VELOCITY L.COPLANAR-AREA-OF-OVERLAP-DERIV))
 (((M +) L.FLUX-THROUGH-LOOP-DERIV L.INDUCED-EMF-MAG) (0 0))
 (((M +) L.NET-MAG-FORCE-X L.NET-FORCE-X) (0 0))
 (((M +) L.NET-MAG-FORCE-Y L.NET-FORCE-Y) (0 0))
 (((M +) L.NET-MAG-FORCE-Z L.NET-FORCE-Z) (0 0))
 (((M +) L.NET-FORCE-X L.X-ACC) (0 0))
 (((M +) L.NET-FORCE-Y L.Y-ACC) (0 0))
 (((M +) L.NET-FORCE-Z L.Z-ACC) (0 0))
 (((M +) L.LOOP-CURRENT-FLOW L.M-2.SOURCE-OF-FIELD.FLD-STRENGTH)
  (0 0))
 (((M +) OVERLAP-BB-RIGHT-VIEW.INSTANTIATION.OVERLAP-LENGTH
   L.COPLANAR-AREA-OF-OVERLAP)
  (0 0))
 ((D/DT L.BM-YVAL L.Y-VELOCITY)) ((D/DT L.TM-YVAL L.Y-VELOCITY))
 ((D/DT L.LM-XVAL L.X-VELOCITY)) ((D/DT L.RM-XVAL L.X-VELOCITY))
 ((D/DT L.PROJ-AREA-OF-OVERLAP L.PROJ-AREA-OF-OVERLAP-DERIV))
 ((D/DT L.FLUX-THROUGH-LOOP L.FLUX-THROUGH-LOOP-DERIV))
 ((D/DT L.COPLANAR-AREA-OF-OVERLAP
   L.COPLANAR-AREA-OF-OVERLAP-DERIV))
 ((D/DT L.Y-VELOCITY L.Y-ACC)) ((D/DT L.X-VELOCITY L.X-ACC))
 ((MINUS L.PROJ-AREA-OF-OVERLAP-DERIV L.FLUX-THROUGH-LOOP-DERIV))
 ((MINUS MAG-FORCE-MAGNITUDE-3.INSTANTIATION.TEMP-VAR
   L.MAG-FORCE-MAGNITUDE))
 ((MULT L.PROJ-AREA-OF-OVERLAP F.FLD-STRENGTH L.FLUX-THROUGH-LOOP))
 ((MULT L.LOOP-CURRENT-FLOW L.TOTAL-RESISTANCE L.INDUCED-EMF-MAG))
 ((MULT L.LOOP-CURRENT-FLOW F.FLD-STRENGTH
   MAG-FORCE-MAGNITUDE-3.INSTANTIATION.TEMP-VAR))
 ((M+ L.COPLANAR-AREA-OF-OVERLAP-DERIV
   L.PROJ-AREA-OF-OVERLAP-DERIV)
  (0 0))
 ((M+ L.INDUCED-EMF-MAG L.LOOP-CURRENT-FLOW) (0 0))
 ((EQUAL DIF-<F.RM-XVAL>-<L.LM-XVAL>
         OVERLAP-BB-RIGHT-VIEW.INSTANTIATION.OVERLAP-LENGTH))
 ((EQUAL L.COPLANAR-AREA-OF-OVERLAP L.PROJ-AREA-OF-OVERLAP))
 ((EQUAL L.RIGHT-OVERLAP-LENGTH
         OVERLAP-BB-RIGHT-VIEW.INSTANTIATION.OVERLAP-LENGTH))
 ((EQUAL L.THETA-Z L.THETA-Z-0)))
(TRANSITIONS
 ((AND (OR (F.FLD-STRENGTH (0 DEC))
           (F.FLD-STRENGTH ((MINF 0) NIL)))
       (OR (L.FLUX-THROUGH-LOOP-DERIV (0 DEC))
           (L.FLUX-THROUGH-LOOP-DERIV ((MINF 0) NIL)))))
 ((OR (L.FLUX-THROUGH-LOOP-DERIV (0 DEC))
      (L.FLUX-THROUGH-LOOP-DERIV ((MINF 0) NIL))))
 ((OR (L.FLUX-THROUGH-LOOP-DERIV (0 STD))
      (L.FLUX-THROUGH-LOOP-DERIV (0 DEC))
      (L.FLUX-THROUGH-LOOP-DERIV ((MINF 0) NIL))))
 ((AND (OR (DIF-<F.BM-YVAL>-<L.BM-YVAL> (0 STD))
           (DIF-<F.BM-YVAL>-<L.BM-YVAL> (0 DEC))
           (DIF-<F.BM-YVAL>-<L.BM-YVAL> ((MINF 0) NIL)))
       (OR (DIF-<F.TM-YVAL>-<L.TM-YVAL> (0 STD))
           (DIF-<F.TM-YVAL>-<L.TM-YVAL> (0 INC))
           (DIF-<F.TM-YVAL>-<L.TM-YVAL> ((0 INF) NIL)))
```

```
      (OR (DIF-<F.LM-XVAL>-<L.LM-XVAL> (0 STD))
          (DIF-<F.LM-XVAL>-<L.LM-XVAL> (0 DEC))
          (DIF-<F.LM-XVAL>-<L.LM-XVAL> ((MINF 0) NIL)))
      (OR (DIF-<F.RM-XVAL>-<L.RM-XVAL> (0 STD))
          (DIF-<F.RM-XVAL>-<L.RM-XVAL> (0 INC))
          (DIF-<F.RM-XVAL>-<L.RM-XVAL> ((0 INF) NIL)))))
 ((AND (OR (DIF-<F.LM-XVAL>-<L.LM-XVAL> (0 INC))
          (DIF-<F.LM-XVAL>-<L.LM-XVAL> ((0 INF) NIL)))
      (OR (DIF-<F.RM-XVAL>-<L.RM-XVAL> (0 STD))
          (DIF-<F.RM-XVAL>-<L.RM-XVAL> (0 INC))
          (DIF-<F.RM-XVAL>-<L.RM-XVAL> ((0 INF) NIL)))))
 ((AND (OR (DIF-<F.TM-YVAL>-<L.TM-YVAL> (0 DEC))
          (DIF-<F.TM-YVAL>-<L.TM-YVAL> ((MINF 0) NIL)))
      (OR (DIF-<F.BM-YVAL>-<L.BM-YVAL> (0 STD))
          (DIF-<F.BM-YVAL>-<L.BM-YVAL> (0 DEC))
          (DIF-<F.BM-YVAL>-<L.BM-YVAL> ((MINF 0) NIL)))))
 ((AND (OR (DIF-<F.TM-YVAL>-<L.TM-YVAL> (0 STD))
          (DIF-<F.TM-YVAL>-<L.TM-YVAL> (0 INC))
          (DIF-<F.TM-YVAL>-<L.TM-YVAL> ((0 INF) NIL)))
      (OR (DIF-<F.BM-YVAL>-<L.BM-YVAL> (0 INC))
          (DIF-<F.BM-YVAL>-<L.BM-YVAL> ((0 INF) NIL)))))
 ((OR (F.FLD-STRENGTH (0 INC)) (F.FLD-STRENGTH ((0 INF) NIL))))
 ((AND (OR (F.FLD-STRENGTH (0 INC)) (F.FLD-STRENGTH ((0 INF) NIL)))
      (OR (L.FLUX-THROUGH-LOOP-DERIV (0 INC))
          (L.FLUX-THROUGH-LOOP-DERIV ((0 INF) NIL)))))
 ((AND (OR (F.FLD-STRENGTH (0 INC)) (F.FLD-STRENGTH ((0 INF) NIL)))
      (OR (L.FLUX-THROUGH-LOOP-DERIV (0 DEC))
          (L.FLUX-THROUGH-LOOP-DERIV ((MINF 0) NIL)))))
 ((OR (DIF-<L.THETA-Z>-<L.THETA-Z-0> (0 INC))
      (DIF-<L.THETA-Z>-<L.THETA-Z-0> (0 DEC))
      (NOT (DIF-<L.THETA-Z>-<L.THETA-Z-0> (0 NIL)))))
 ((OR (DIF-<F.LM-XVAL>-<L.RM-XVAL> (0 STD))
      (DIF-<F.LM-XVAL>-<L.RM-XVAL> (0 INC))
      (DIF-<F.LM-XVAL>-<L.RM-XVAL> ((0 INF) NIL))))
 ((OR (DIF-<F.RM-XVAL>-<L.LM-XVAL> (0 STD))
      (DIF-<F.RM-XVAL>-<L.LM-XVAL> (0 DEC))
      (DIF-<F.RM-XVAL>-<L.LM-XVAL> ((MINF 0) NIL))))
 ((OR (F.FLD-STRENGTH (0 STD))
      (F.FLD-STRENGTH (0 INC))
      (F.FLD-STRENGTH ((0 INF) NIL))))
 ((OR (DIF-<F.LM-XVAL>-<L.LM-XVAL> (0 INC))
      (DIF-<F.LM-XVAL>-<L.LM-XVAL> ((0 INF) NIL))))
 ((OR (DIF-<F.RM-XVAL>-<L.RM-XVAL> (0 STD))
      (DIF-<F.RM-XVAL>-<L.RM-XVAL> (0 INC))
      (DIF-<F.RM-XVAL>-<L.RM-XVAL> ((0 INF) NIL))))
 ((OR (DIF-<F.RM-XVAL>-<L.LM-XVAL> (0 STD))
      (DIF-<F.RM-XVAL>-<L.LM-XVAL> (0 DEC))
      (DIF-<F.RM-XVAL>-<L.LM-XVAL> ((MINF 0) NIL))))
 ((OR (DIF-<F.LM-XVAL>-<L.RM-XVAL> (0 STD))
      (DIF-<F.LM-XVAL>-<L.RM-XVAL> (0 INC))
      (DIF-<F.LM-XVAL>-<L.RM-XVAL> ((0 INF) NIL))))
 ((OR (DIF-<F.BM-YVAL>-<L.TM-YVAL> (0 STD))
      (DIF-<F.BM-YVAL>-<L.TM-YVAL> (0 INC))
      (DIF-<F.BM-YVAL>-<L.TM-YVAL> ((0 INF) NIL))))
 ((OR (DIF-<F.TM-YVAL>-<L.BM-YVAL> (0 STD))
      (DIF-<F.TM-YVAL>-<L.BM-YVAL> (0 DEC))
      (DIF-<F.TM-YVAL>-<L.BM-YVAL> ((MINF 0) NIL))))
 ((OR (DIF-<F.TM-YVAL>-<L.BM-YVAL> (0 STD))
      (DIF-<F.TM-YVAL>-<L.BM-YVAL> (0 DEC))
      (DIF-<F.TM-YVAL>-<L.BM-YVAL> ((MINF 0) NIL))))
 ((OR (DIF-<F.BM-YVAL>-<L.TM-YVAL> (0 STD))
      (DIF-<F.BM-YVAL>-<L.TM-YVAL> (0 INC))
      (DIF-<F.BM-YVAL>-<L.TM-YVAL> ((0 INF) NIL)))))
(OTHER (QSPACE-HIERARCHY) (QDE-CURVATURE-AT-STEADY NIL)
 (NO-NEW-LANDMARKS DIF-<F.RM-XVAL>-<L.LM-XVAL>
  DIF-<F.RM-XVAL>-<L.RM-XVAL> DIF-<F.LM-XVAL>-<L.RM-XVAL>
  DIF-<F.TM-YVAL>-<L.BM-YVAL> DIF-<F.TM-YVAL>-<L.TM-YVAL>
  DIF-<F.BM-YVAL>-<L.TM-YVAL> DIF-<F.BM-YVAL>-<L.BM-YVAL>
```

```
 DIF-<L.THETA-Z>-<L.THETA-Z-0>
 DIF-<L.RIGHT-OVERLAP-LENGTH>-
     <OVERLAP-BB-RIGHT-VIEW.INSTANTIATION.OVERLAP-LENGTH>
 DIF-<L.COPLANAR-AREA-OF-OVERLAP>-<L.PROJ-AREA-OF-OVERLAP>
 DIF-<F.BS-DIAMETER>-<L.BS-DIAMETER> DIF-<L.LM-XVAL>-<L.RM-XVAL>
 DIF-<F.LM-XVAL>-<L.LM-XVAL> DIF-<F.LM-XVAL>-<F.RM-XVAL>
 DIF-<L.BM-YVAL>-<L.TM-YVAL> DIF-<F.BM-YVAL>-<F.TM-YVAL>)
(REGIONS
 ((OR (L.FLUX-THROUGH-LOOP-DERIV (0 INC))
      (L.FLUX-THROUGH-LOOP-DERIV ((0 INF) NIL)))))
(SCENARIO-CONDITION
 (OR (OR (DIF-<F.BM-YVAL>-<F.TM-YVAL> (0 STD))
         (DIF-<F.BM-YVAL>-<F.TM-YVAL> (0 INC))
         (DIF-<F.BM-YVAL>-<F.TM-YVAL> ((0 INF) NIL)))
     (OR (DIF-<L.BM-YVAL>-<L.TM-YVAL> (0 STD))
         (DIF-<L.BM-YVAL>-<L.TM-YVAL> (0 INC))
         (DIF-<L.BM-YVAL>-<L.TM-YVAL> ((0 INF) NIL)))
     (OR (DIF-<F.BM-YVAL>-<F.TM-YVAL> (0 STD))
         (DIF-<F.BM-YVAL>-<F.TM-YVAL> (0 INC))
         (DIF-<F.BM-YVAL>-<F.TM-YVAL> ((0 INF) NIL)))
     (OR (DIF-<L.BM-YVAL>-<L.TM-YVAL> (0 STD))
         (DIF-<L.BM-YVAL>-<L.TM-YVAL> (0 INC))
         (DIF-<L.BM-YVAL>-<L.TM-YVAL> ((0 INF) NIL)))
     (OR (DIF-<F.LM-XVAL>-<F.RM-XVAL> (0 STD))
         (DIF-<F.LM-XVAL>-<F.RM-XVAL> (0 INC))
         (DIF-<F.LM-XVAL>-<F.RM-XVAL> ((0 INF) NIL)))
     (OR (DIF-<L.LM-XVAL>-<L.RM-XVAL> (0 STD))
         (DIF-<L.LM-XVAL>-<L.RM-XVAL> (0 INC))
         (DIF-<L.LM-XVAL>-<L.RM-XVAL> ((0 INF) NIL)))
     (OR (DIF-<F.LM-XVAL>-<F.RM-XVAL> (0 STD))
         (DIF-<F.LM-XVAL>-<F.RM-XVAL> (0 INC))
         (DIF-<F.LM-XVAL>-<F.RM-XVAL> ((0 INF) NIL)))
     (OR (DIF-<L.LM-XVAL>-<L.RM-XVAL> (0 STD))
         (DIF-<L.LM-XVAL>-<L.RM-XVAL> (0 INC))
         (DIF-<L.LM-XVAL>-<L.RM-XVAL> ((0 INF) NIL)))
     (OR (L.X-VELOCITY (MAXVAL STD))
         (L.X-VELOCITY (MAXVAL INC))
         (L.X-VELOCITY ((MAXVAL INF) NIL)))
     (OR (F.FLD-STRENGTH (0 STD))
         (F.FLD-STRENGTH (0 INC))
         (F.FLD-STRENGTH ((0 INF) NIL)))
     (OR (L.Y-VELOCITY (MAXVAL STD))
         (L.Y-VELOCITY (MAXVAL INC))
         (L.Y-VELOCITY ((MAXVAL INF) NIL)))
     (OR (L.Y-VELOCITY (0 INC))
         (L.Y-VELOCITY (0 DEC))
         (NOT (L.Y-VELOCITY (0 NIL))))
     (OR (L.THETA-Z-0 (MAXVAL STD))
         (L.THETA-Z-0 (MAXVAL INC))
         (L.THETA-Z-0 ((MAXVAL INF) NIL)))
     (OR (L.THETA-Z-0 (0 STD))
         (L.THETA-Z-0 (0 DEC))
         (L.THETA-Z-0 ((MINF 0) NIL)))
     (OR (DIF-<L.THETA-Z>-<L.THETA-Z-0> (0 INC))
         (DIF-<L.THETA-Z>-<L.THETA-Z-0> (0 DEC))
         (NOT (DIF-<L.THETA-Z>-<L.THETA-Z-0> (0 NIL))))
     (OR (L.THETA-Z (MAXVAL STD))
         (L.THETA-Z (MAXVAL INC))
         (L.THETA-Z ((MAXVAL INF) NIL)))
     (OR (L.THETA-Z (0 STD))
         (L.THETA-Z (0 DEC))
         (L.THETA-Z ((MINF 0) NIL)))
     (OR (DIF-<L.THETA-Z>-<L.THETA-Z-0> (0 INC))
         (DIF-<L.THETA-Z>-<L.THETA-Z-0> (0 DEC))
         (NOT (DIF-<L.THETA-Z>-<L.THETA-Z-0> (0 NIL))))
     (OR (F.BS-DIAMETER (MAXVAL STD))
         (F.BS-DIAMETER (MAXVAL INC))
         (F.BS-DIAMETER ((MAXVAL INF) NIL)))
```

```
(OR (F.BS-DIAMETER (0 STD))
    (F.BS-DIAMETER (0 DEC))
    (F.BS-DIAMETER ((MINF 0) NIL)))
(OR (DIF-<F.BS-DIAMETER>-<L.BS-DIAMETER> (0 STD))
    (DIF-<F.BS-DIAMETER>-<L.BS-DIAMETER> (0 DEC))
    (DIF-<F.BS-DIAMETER>-<L.BS-DIAMETER> ((MINF 0) NIL)))
(OR (F.PROJ-FACTOR-XY-PLANE (ONE INC))
    (F.PROJ-FACTOR-XY-PLANE (ONE DEC))
    (NOT (F.PROJ-FACTOR-XY-PLANE (ONE NIL))))
(OR (F.SIN-THETA-XY (0 INC))
    (F.SIN-THETA-XY (0 DEC))
    (NOT (F.SIN-THETA-XY (0 NIL))))
(OR (F.COS-THETA-XY (ONE INC))
    (F.COS-THETA-XY (ONE DEC))
    (NOT (F.COS-THETA-XY (ONE NIL))))
(OR (F.SIN-THETA-X (0 INC))
    (F.SIN-THETA-X (0 DEC))
    (NOT (F.SIN-THETA-X (0 NIL))))
(OR (F.COS-THETA-X (ONE INC))
    (F.COS-THETA-X (ONE DEC))
    (NOT (F.COS-THETA-X (ONE NIL))))
(OR (F.SIN-THETA-Y (0 INC))
    (F.SIN-THETA-Y (0 DEC))
    (NOT (F.SIN-THETA-Y (0 NIL))))
(OR (F.COS-THETA-Y (ONE INC))
    (F.COS-THETA-Y (ONE DEC))
    (NOT (F.COS-THETA-Y (ONE NIL))))
(OR (L.BS-DIAMETER (MAXVAL STD))
    (L.BS-DIAMETER (MAXVAL INC))
    (L.BS-DIAMETER ((MAXVAL INF) NIL)))
(OR (DIF-<F.BS-DIAMETER>-<L.BS-DIAMETER> (0 STD))
    (DIF-<F.BS-DIAMETER>-<L.BS-DIAMETER> (0 DEC))
    (DIF-<F.BS-DIAMETER>-<L.BS-DIAMETER> ((MINF 0) NIL)))
(OR (L.BS-DIAMETER (0 STD))
    (L.BS-DIAMETER (0 DEC))
    (L.BS-DIAMETER ((MINF 0) NIL)))
(OR (L.TOTAL-RESISTANCE (MAXVAL STD))
    (L.TOTAL-RESISTANCE (MAXVAL INC))
    (L.TOTAL-RESISTANCE ((MAXVAL INF) NIL)))
(OR (F.BM-YVAL (0 DEC)) (F.BM-YVAL ((MINF 0) NIL)))
(OR (F.BM-YVAL (MAXVAL INC)) (F.BM-YVAL ((MAXVAL INF) NIL)))
(OR (L.TM-YVAL (0 DEC)) (L.TM-YVAL ((MINF 0) NIL)))
(OR (L.TM-YVAL (MAXVAL INC)) (L.TM-YVAL ((MAXVAL INF) NIL)))
(OR (F.TM-YVAL (0 DEC)) (F.TM-YVAL ((MINF 0) NIL)))
(OR (F.TM-YVAL (MAXVAL INC)) (F.TM-YVAL ((MAXVAL INF) NIL)))
(OR (L.BM-YVAL (0 DEC)) (L.BM-YVAL ((MINF 0) NIL)))
(OR (L.BM-YVAL (MAXVAL INC)) (L.BM-YVAL ((MAXVAL INF) NIL)))
(OR (F.LM-XVAL (0 DEC)) (F.LM-XVAL ((MINF 0) NIL)))
(OR (F.LM-XVAL (MAXVAL INC)) (F.LM-XVAL ((MAXVAL INF) NIL)))
(OR (L.RM-XVAL (0 DEC)) (L.RM-XVAL ((MINF 0) NIL)))
(OR (L.RM-XVAL (MAXVAL INC)) (L.RM-XVAL ((MAXVAL INF) NIL)))
(OR (F.RM-XVAL (0 DEC)) (F.RM-XVAL ((MINF 0) NIL)))
(OR (F.RM-XVAL (MAXVAL INC)) (F.RM-XVAL ((MAXVAL INF) NIL)))
(OR (L.LM-XVAL (0 DEC)) (L.LM-XVAL ((MINF 0) NIL)))
(OR (L.LM-XVAL (MAXVAL INC)) (L.LM-XVAL ((MAXVAL INF) NIL)))
(OR (L.X-VELOCITY (0 DEC)) (L.X-VELOCITY ((MINF 0) NIL)))
(OR (F.FLD-STRENGTH (0 INC)) (F.FLD-STRENGTH ((0 INF) NIL)))
(F.FLD-STRENGTH (0 NIL))
(OR (DIF-<L.THETA-Z>-<L.THETA-Z-0> (0 DEC))
    (DIF-<L.THETA-Z>-<L.THETA-Z-0> ((MINF 0) NIL)))
(OR (DIF-<L.THETA-Z>-<L.THETA-Z-0> (0 INC))
    (DIF-<L.THETA-Z>-<L.THETA-Z-0> ((0 INF) NIL)))
(OR (F.PROJ-FACTOR-XY-PLANE (MINUS-ONE DEC))
    (F.PROJ-FACTOR-XY-PLANE ((MINF MINUS-ONE) NIL)))
(OR (F.PROJ-FACTOR-XY-PLANE (ONE INC))
    (F.PROJ-FACTOR-XY-PLANE ((ONE INF) NIL)))
(OR (F.SIN-THETA-XY (MINUS-ONE DEC))
    (F.SIN-THETA-XY ((MINF MINUS-ONE) NIL)))
```

```
      (OR (F.SIN-THETA-XY (ONE INC))
          (F.SIN-THETA-XY ((ONE INF) NIL)))
      (OR (F.COS-THETA-XY (MINUS-ONE DEC))
          (F.COS-THETA-XY ((MINF MINUS-ONE) NIL)))
      (OR (F.COS-THETA-XY (ONE INC))
          (F.COS-THETA-XY ((ONE INF) NIL)))
      (OR (F.SIN-THETA-X (MINUS-ONE DEC))
          (F.SIN-THETA-X ((MINF MINUS-ONE) NIL)))
      (OR (F.SIN-THETA-X (ONE INC))
          (F.SIN-THETA-X ((ONE INF) NIL)))
      (OR (F.COS-THETA-X (MINUS-ONE DEC))
          (F.COS-THETA-X ((MINF MINUS-ONE) NIL)))
      (OR (F.COS-THETA-X (ONE INC))
          (F.COS-THETA-X ((ONE INF) NIL)))
      (OR (F.SIN-THETA-Y (MINUS-ONE DEC))
          (F.SIN-THETA-Y ((MINF MINUS-ONE) NIL)))
      (OR (F.SIN-THETA-Y (ONE INC))
          (F.SIN-THETA-Y ((ONE INF) NIL)))
      (OR (F.COS-THETA-Y (MINUS-ONE DEC))
          (F.COS-THETA-Y ((MINF MINUS-ONE) NIL)))
      (OR (F.COS-THETA-Y (ONE INC))
          (F.COS-THETA-Y ((ONE INF) NIL)))
      (OR (L.TOTAL-RESISTANCE (0 DEC))
          (L.TOTAL-RESISTANCE ((MINF 0) NIL)))
      (OR (L.TOTAL-RESISTANCE (MAXVAL INC))
          (L.TOTAL-RESISTANCE ((MAXVAL INF) NIL)))))
 (MODEL #<Model: M-2>)
 (UNREACHABLE-VALUES (L.Z-ACC MINF INF) (L.Y-ACC MINF INF)
  (L.X-ACC MINF INF) (L.NET-FORCE-Z MINF INF)
  (L.NET-FORCE-Y MINF INF) (L.NET-FORCE-X MINF INF)
  (MAG-FORCE-MAGNITUDE-3.INSTANTIATION.TEMP-VAR MINF INF)
  (L.NET-MAG-FORCE-Z MINF INF) (L.NET-MAG-FORCE-Y MINF INF)
  (L.MAG-FORCE-MAGNITUDE MINF INF) (L.NET-MAG-FORCE-X MINF)
  (L.M-2.SOURCE-OF-FIELD.FLD-STRENGTH INF) (L.INDUCED-EMF-MAG INF)
  (L.LOOP-CURRENT-FLOW INF) (F.THETA-Y MINF INF)
  (F.THETA-X MINF INF) (L.FLUX-THROUGH-LOOP MINF INF)
  (L.PROJ-AREA-OF-OVERLAP MINF INF)
  (L.FLUX-THROUGH-LOOP-DERIV MINF INF)
  (L.PROJ-AREA-OF-OVERLAP-DERIV MINF INF)
  (OVERLAP-BB-RIGHT-VIEW.INSTANTIATION.OVERLAP-LENGTH INF)
  (L.COPLANAR-AREA-OF-OVERLAP-DERIV MINF INF)
  (L.RIGHT-OVERLAP-LENGTH MINF INF)
  (L.COPLANAR-AREA-OF-OVERLAP INF)
  (DIF-<L.COPLANAR-AREA-OF-OVERLAP>-<L.PROJ-AREA-OF-OVERLAP> MINF
   INF)
  (DIF-<L.RIGHT-OVERLAP-LENGTH>-
      <OVERLAP-BB-RIGHT-VIEW.INSTANTIATION.OVERLAP-LENGTH> MINF INF)
  (DIF-<F.BM-YVAL>-<L.BM-YVAL> MINF INF)
  (DIF-<F.BM-YVAL>-<L.TM-YVAL> MINF INF)
  (DIF-<F.TM-YVAL>-<L.TM-YVAL> MINF INF)
  (DIF-<F.TM-YVAL>-<L.BM-YVAL> MINF INF)
  (DIF-<F.LM-XVAL>-<L.RM-XVAL> MINF INF)
  (DIF-<F.RM-XVAL>-<L.RM-XVAL> MINF INF)
  (DIF-<F.RM-XVAL>-<L.LM-XVAL> MINF INF)
  (DIF-<F.BS-DIAMETER>-<L.BS-DIAMETER> INF 0)
  (DIF-<L.LM-XVAL>-<L.RM-XVAL> MINF 0)
  (DIF-<F.LM-XVAL>-<L.LM-XVAL> MINF 0)
  (DIF-<F.LM-XVAL>-<F.RM-XVAL> MINF 0)
  (DIF-<L.BM-YVAL>-<L.TM-YVAL> MINF 0)
  (DIF-<F.BM-YVAL>-<F.TM-YVAL> MINF 0)
  (L.TOTAL-RESISTANCE 0 MAXVAL) (L.BS-DIAMETER 0 MAXVAL)
  (F.COS-THETA-Y 0) (F.COS-THETA-X 0) (F.COS-THETA-XY 0)
  (F.PROJ-FACTOR-XY-PLANE 0) (F.BS-DIAMETER 0 MAXVAL)
  (L.THETA-Z 0 MAXVAL) (L.THETA-Z-0 0 MAXVAL)
  (L.Y-VELOCITY MINF MAXVAL) (F.FLD-STRENGTH MINF 0)
  (L.X-VELOCITY MAXVAL) (F.RM-XVAL 0 MAXVAL) (L.RM-XVAL MAXVAL)
  (F.LM-XVAL 0 MAXVAL) (F.TM-YVAL 0 MAXVAL) (F.BM-YVAL 0 MAXVAL))))
```

Bibliography

[1] A. Abella and J. Kender. Qualitatively describing objects using spatial prepositions. In *Proc. 11th National Conf. on Artificial Intelligence*, Cambridge, MA, 1993. AAAI/MIT Press.

[2] *ACM Workshop on Advances in Geographic Information Systems*, Arlington, VA, November 1993.

[3] E. Andre, G. Herzog, and T. Rist. On the simultaneous interpretation of real world image sequences and their natural language description: The system soccer. In *Proc. 8th Eur. Conf. on Artificial Intelligence*, Munich, 1988. John Wiley and Sons.

[4] D. H. Ballard and C. M. Brown. *Computer Vision*. Prentice-Hall, Englewood Cliffs, NJ, 1982.

[5] F. Beer and Jr. E. R. Johnston. *Vector Mechanics for Engineers*. McGraw-Hill, New York, 1984.

[6] G. Brajnik. Introducing boundary conditions in semi–quantitative simulation. In *Proc. 9th Int. Workshop on Qualitative Reasoning About Physical Systems*, Amsterdam, May 1995.

[7] S. K. Chang and E. Jungert. The design of pictorial databases based upon the theory of symbolic projections. *Journal of Visual Languages and Computing*, 2(3):195–215, 1991.

[8] S. K. Chang, Q. Y. Shi, and C. W. Yan. Iconic indexing by 2-d strings. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 9(3):413–428, 1987.

[9] J. Crawford, A. Farquhar, and B. Kuipers. QPC: A compiler from physical models into qualitative differential equations. In *Proc. 8th National Conf. on Artificial Intelligence*, San Mateo, CA, 1990. Morgan Kaufmann.

[10] J. Crawford and B. Kuipers. Algernon - a tractable system for knowledge-representation. In *AAAI Spring Symposium on Implemented Knowledge Representation and Reasoning Systems*, Palo Alto, CA, 1991.

[11] Z. Cui, A. G. Cohn, and D. Randell. Qualitative simulation based on a logical formalism of space and time. In *Proc. 10th National Conf. on Artificial Intelligence*, Cambridge, MA, 1992. AAAI/MIT Press.

[12] E. Davis. *Representing and Acquiring Geographic Knowledge*. Morgan-Kaufmann, Los Altos, CA, 1986.

[13] M. Egenhofer and K. Al-Taha. Reasoning about gradual changes in topological relationships. In A. Frank, I. Campari, and U. Formentini, editors, *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*. Springer-Verlag, Berlin, 1992.

[14] B. Faltings. Qualitative kinematics in mechanisms. *Artificial Intelligence*, 44:89–119, 1990.

[15] B. Faltings. A symbolic approach to qualitative kinematics. *Artificial Intelligence*, 56:139–170, 1992.

[16] A. Farquhar. *Automated Modeling of Physical Systems in the Presence of Incomplete Knowledge*. PhD thesis, Artificial Intelligence Laboratory, Department of Computer Sciences, University of Texas at Austin, 1993.

[17] A. Farquhar. A qualitative physics compiler. In *Proc. 12th National Conf. on Artificial Intelligence*, Cambridge, MA, 1994. AAAI/MIT Press.

[18] S. Feiner and K. McKeown. Coordinating text and graphics in explanation generation. In *Proc. 8th National Conf. on Artificial Intelligence*, Cambridge, MA, 1990. AAAI/MIT Press.

[19] K. Forbus. Spatial and qualitative aspects of reasoning about motion. In *Proc. 1st National Conf. on Artifical Intelligence*, San Mateo, CA, 1980. Morgan Kaufmann.

[20] K. Forbus. Qualitative process theory. *Artificial Intelligence*, 24:85–168, 1984.

[21] K. Forbus. The qualitative process engine. In D. Weld and J. de Kleer, editors, *Readings in Qualitative Reasoning about Physical Systems*. Morgan Kaufmann, San Mateo, CA, 1990.

[22] K. Forbus. Using qualitative physics to build articulate software for thermodynamics education. In *Proc. 12th National Conf. on Artificial Intelligence*, Cambridge, MA, 1994. AAAI/MIT Press.

[23] K. Forbus, P. Nielsen, and B. Faltings. Qualitative spatial reasoning: The clock project. *Artificial Intelligence*, 51:417–471, 1991.

[24] A. Frank, I. Campari, and U. Formentini, editors. *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*. Springer-Verlag, Berlin, 1992.

[25] A. U. Frank. Qualitative spatial reasoning with cardinal directions. In *Proceedings Seventh Austrian Conference on Artificial Intelligence*, Berlin, 1991. Springer-Verlag.

[26] D. Franke and D. Dvorak. CC: Component-connection models for qualitative simulation – a user's guide. Technical Report AI90–126, Artificial Intelligence

Laboratory, Department of Computer Sciences, University of Texas at Austin, January 1990.

[27] C. Freksa. Using orientation information for qualitative spatial reasoning. In A. Frank, I. Campari, and U. Formentini, editors, *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*. Springer-Verlag, Berlin, 1992.

[28] A Galton. Towards an integrated logic of space, time, and motion. In *Proc. 13th Int. Joint Conf. on Artificial Intelligence*, San Mateo, CA, 1993. Morgan Kaufmann.

[29] K. Gapp. Basic meanings of spatial relations: Computation and evaluation in 3d space. In *Proc. 12th National Conf. on Artificial Intelligence*, Cambridge, MA, 1994. AAAI/MIT Press.

[30] J. Glasgow. Spatial reasoning in indeterminate worlds. In *Proc. 12th National Conf. on Artificial Intelligence*, Cambridge, MA, 1994. AAAI/MIT Press.

[31] T. Hadzilacos and N. Tryfona. A model for expressing topological integrity constraints in geographic databases. In A. Frank, I. Campari, and U. Formentini, editors, *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*. Springer-Verlag, Berlin, 1992.

[32] S. He, N. Abe, and T. Kitahashi. Assembly plan generation by integrating pictorial and textual information in an assembly illustration. In *Working Notes of the AAAI-94 Workshop on Integration of Natural Language and Vision Processing*, Seattle, Washington, July 1994.

[33] D. Hernandez. Relative repesentation of spatial knowledge: The 2-d case. In D. Mark and A. Frank, editors, *Cognitive and Linguistic Aspects of Geographic Space*. Kluwer Academic Publishers, Dordrecht, 1991.

[34] J. Herring. The mathematical modeling of spatial and non-spatial information in geographic information systems. In D. Mark and A. Frank, editors, *Cognitive and Linguistic Aspects of Geographic Space*. Kluwer Academic Publishers, Dordrecht, 1991.

[35] A. Herskovits. Semantics and pragmatics of locative expressions. *Cognitive Science*, 9(3):341–378, 1985.

[36] L. Joskowicz and E. Sacks. Computational kinematics. *Artificial Intelligence*, 51:381–416, 1991.

[37] E. Jungert. The observer's point of view: An extension of symbolic projections. In A. Frank, I. Campari, and U. Formentini, editors, *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*. Springer-Verlag, Berlin, 1992.

[38] H. Kay and B. Kuipers. Numerical behavior envelopes for qualitative models. In *Proc. 11th National Conf. on Artificial Intelligence*, pages 606–613, Cambridge, MA, 1993. AAAI/MIT Press.

[39] S. Kerpedjiev. Describing spatial relations in weather reports through prepositions. In *Working Notes of the AAAI-94 Workshop on Integration of Natural Language and Vision Processing*, Seattle, Washington, July 1994.

[40] H. Kim. Qualitative reasoning about the geometry of fluid flow. In *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, Cambridge, MA, 1990. Lawrence Erlbaum and Associates.

[41] T. Kiriyama, T. Tomiyama, and H. Yoshikawa. Qualitative reasoning in conceptual design with physical features. In Boi Faltings and Peter Struss, editors, *Recent Advances in Qualitative Physics*. MIT Press, Cambridge, MA, 1992.

[42] B. Kuipers. Qualitative simulation. *Artificial Intelligence*, 29:289–338, 1986.

[43] B. Kuipers. *Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge*. The MIT Press, Cambridge, MA, 1994.

[44] B. Kuipers and D. Berleant. Using incomplete quantitative knowledge in qualitative reasoning. *Proc. 7th National Conf. on Artificial Intelligence*, 1988.

[45] B. Kuipers and Y. T. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Journal of Robotics and Autonomous Systems*, 8:47–63, 1991.

[46] B. Kuipers and T. Levitt. Navigation and mapping in large scale space. *AI Magazine*, 9(2):25–43, 1988.

[47] B. Landau and R. Jackendoff. “what” and “where” in spatial language and spatial cognition. *Behavioral and Brain Sciences*, 16:217–265, 1993.

[48] J. Larkin and H. Simon. Why a diagram is (sometimes) worth 10,000 words. *Cognitive Science*, 11:65–99, 1987.

[49] L. Latecki and S. Pribbenow. On hybrid reasoning for processing spatial expressions. In *Proc. 10th European Conf. on Artificial Intelligence*, Chichester, England, 1992. John Wiley and Sons.

[50] L. Latecki and R. Rohrig. Orientation and qualitative angle for spatial reasoning. In *Proc. 13th Int. Joint Conf. on Artificial Intelligence*, San Mateo, CA, 1993. Morgan Kaufmann.

[51] R. Laurini and D. Thompson. *Fundamentals of Spatial Information Systems.* Academic Press, London, 1992.

[52] S. Lee and M. Shan. Access methods of image databases. *Intl. Journal of Pattern Recognition and Artificial Intelligence*, 4(1):27–44, 1990.

[53] W. Maass. From visual perception to multimodal communication: Incremental route descriptions. *AI Review Journal*, 8(2-3), 1994.

[54] D. Mark and A. Frank, editors. *Cognitive and Linguistic Aspects of Geographic Space*. Kluwer Academic Publishers, Dordrecht, 1991.

[55] P. McKevitt, editor. *Working Notes of the AAAI Workshop on Integration of Natural Language and Vision Processing*, Seattle, WA, 1994.

[56] J. P. Mohren and J. Muller. A geometric approach to depictional representation of spatial relations. In *Proc. 10th European Conf. on Artificial Intelligence*, Chichester, England, 1992. John Wiley and Sons.

[57] A. Mukerjee and G. Joe. A qualitative model for space. In *Proc. 8th National Conf. on Artificial Intelligence*, Cambridge, MA, 1990. AAAI/MIT Press.

[58] A. Mukerjee and G. Joe. A qualitative spatial representation based on tangency and alignment. Technical Report TAMU 92-003, Department of Computer Sciences, Texas A&M University, College Station, Texas, 1992.

[59] N. Hari Narayanan, editor. *Working Notes of the AAAI Spring Symposium Series, Symposium: Reasoning with Diagrammatic Representations*, Stanford University, 1992.

[60] N. Hari Narayanan, M. Suwa, and H. Motoda. How things appear to work: Predicting behaviors from device diagrams. In *Proc. 12th National Conf. on Artificial Intelligence*, Cambridge, MA, 1994. AAAI/MIT Press.

[61] P. Nielsen. A qualitative approach to mechanical constraint. In *Proc. 7th National Conf. on Artificial Intelligence*, San Mateo, CA, 1988. Morgan Kaufmann.

[62] P. Nielsen. *A Qualitative Approach to Rigid Body Mechanics*. PhD thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 1988.

[63] G. Novak and W. Bulko. Understanding natural language with diagrams. In *Proc. 8th National Conf. on Artificial Intelligence*, Cambridge, MA, 1990. AAAI/MIT Press.

[64] G. Novak and W. Bulko. Diagrams and text as computer input. *Journal of Visual Languages and Computing*, 4:161–175, 1993.

[65] P. Olivier, T. Maeda, and J. Tsujii. Automatic depiction of spatial descriptions. In *Proc. 12th National Conf. on Artificial Intelligence*, Cambridge, MA, 1994. AAAI/MIT Press.

[66] D. Papadias and T. Sellis. Spatial reasoning using symbolic arrays. In A. Frank, I. Campari, and U. Formentini, editors, *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*. Springer-Verlag, Berlin, 1992.

[67] R. Rajagopalan. Manipulating frames of reference for qualitative spatial reasoning. Manuscript, 1992.

[68] R. Rajagopalan. A model of spatial position based on extremal points. In *Proceedings of the ACM Workshop on Advances in Geographic Information Systems*, Arlington, VA, November 1993.

[69] R. Rajagopalan. The figure understander: A tool for the integration of text and graphical input to a knowledge base. In *Proc. 6th IEEE Intl. Conf. on Tools With Artificial Intelligence*, New Orleans, LA, November 1994.

[70] R. Rajagopalan. Integrating text and graphical input to a knowledge base. In *Working Notes of the AAAI-94 Workshop on Integration of Natural Language and Vision Processing*, Seattle, Washington, July 1994.

[71] R. Rajagopalan. A model for integrated spatial and dynamic reasoning about physical systems. In *Proc. 12th National Conf. on Artificial Intelligence*, Cambridge, MA, 1994. AAAI/MIT Press.

[72] R. Rajagopalan. On shape abstractions for qualitative spatial reasoning. In *Working Notes of the AAAI-94 Workshop on Spatial and Temporal Reasoning*, Seattle, Washington, July 1994.

[73] R. Rajagopalan. Picture semantics for integrating text and diagram input. *AI Review Journal*, 10(3-4), 1995.

[74] R. Rajagopalan and B. Kuipers. The figure understander: A system for integrating text and diagram input to a knowledge base. In *Proc. 7th Intl. Conf. on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, Austin, TX, June 1994.

[75] R. Rajagopalan and B. Kuipers. Qualitative spatial reasoning about objects in motion: Application to physics problem solving. In *Proc. 10th IEEE Conf. on Aritificial Intelligence for Applications*, San Antonio, TX, March 1994.

[76] J. Resnick and D. Halliday. *Fundamentals of Physics.* John Wiley and Sons, New York, third edition, 1988.

[77] G. Retz-Schmidt. Various views on spatial prepositions. *AI Magazine*, 9(2):95–105, 1988.

[78] J. Rickel and B. Porter. Automated modeling for answering prediction questions: Selecting the time scale and system boundary. In *Proc. 12th National Conf. on Artificial Intelligence*, Cambridge, MA, 1994. AAAI/MIT Press.

[79] R. Rodriguez, editor. *Working Notes of the AAAI Workshop on Spatial and Temporal Reasoning*, Seattle, WA, 1994.

[80] N. Rowe and E. Guglielmo. Exploiting captions in retrieval of multimedia data. *Information Processing and Management*, 29(4):453–461, 1993.

[81] S. Rubin. *Computer Aids for VLSI Design.* Addison Wesley, Reading, MA, 1987.

[82] G. Skorstad and K. Forbus. Qualitative and quantitative reasoning about thermodynamics. In *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*, Cambridge, MA, 1989. Lawrence Erlbaum and Associates.

[83] R. Srihari. Piction: A system that uses captions to label human faces in newspaper photographs. In *Proc. 9th National Conf. on Artificial Intelligence*, Cambridge, MA, 1991. AAAI/MIT Press.

[84] R. Srihari. Use of captions and other collateral text in understanding photographs. *AI Review Journal*, 8(5-6), 1994.

[85] R. Srihari and D. T. Burhans. Visual semantics: Extracting visual information from text accompanying pictures. In *Proc. 12th National Conf. on Artificial Intelligence*, Cambridge, MA, 1994. AAAI/MIT Press.

[86] S. Tessler, Y. Iwasaki, and K. Law. Qualitative structural analysis using diagrammatic reasoning. In *Proc. 7th Int. Workshop on Qualitative Reasoning About Physical Systems*, Orcas Island, WA, May 1993.

[87] D. Throop. Spatial unification: Qualitative spatial reasoning about steady state mechanisms. Technical Report TR AI89-95, Artificial Intelligence Laboratory, Department of Computer Sciences, University of Texas at Austin, 1989.

[88] J. Weinberg, S. Uckun, G. Biswas, and S. Manganaris. Qualitative vector algebra. In Boi Faltings and Peter Struss, editors, *Recent Advances in Qualitative Physics*. MIT Press, Cambridge, MA, 1992.

[89] D. S. Weld and J. de Kleer. *Readings in Qualitative Reasoning About Physical Systems*. Morgan Kaufmann, San Mateo, CA, 1990.

[90] L. Worrall. *Geographic Information Systems: Developments and Applications*. Belhaven Press, 1990.

Vita

Raman Minakshisundaram Rajagopalan was born in Urbana, Illinois on November 24, 1964, the son of Minakshisundaram and Meenakshi Rajagopalan. After graduating from Messick High School in Memphis, Tennessee, he entered Memphis State University, and received the Bachelor of Science degree in Electrical Engineering in December 1981. He then entered the University of Illinois at Urbana-Champaign, and obtained the Master of Science degree in Electrical Engineering in May 1984. Raman entered the University of Texas at Austin in September 1986, and obtained a Master of Science degree in Computer Science in December 1988.

Raman has been employed with the IBM Corporation (1984-1991), the EDS Corporation (1992), and as a consultant with the Texas Department of Human Services (1994-1995). After completing the doctoral degree, Raman will begin employment with the Intel Corporation in Portland, Oregon.

Permanent Address: 3652 Hendricks Cove
Memphis, TN 38111

This dissertation was typeset with LaTeX 2_ε[1] by the author.

[1] LaTeX 2_ε is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The macros used in formatting this dissertation were written by Dinesh Das, Department of Computer Sciences, The University of Texas at Austin.