

Landscape Grammar

by

Kevin Miles Mayall

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Planning

Waterloo, Ontario, Canada, 2002

©Kevin Mayall, 2002

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Acknowledgements

First, I would like to thank both of my advisors, Dr. Brent Hall and Dr. Thomas Seebohm, for their guidance through this thesis. Brent has shown extraordinary commitment to the project at all stages of its evolution and has shown his interest in every detail. Thomas has always given his utmost confidence in my abilities and sought to attain the highest quality for this project. Both have fulfilled at various times, the roles of directors, quality control and cheering section. Their dedication to the thesis and to myself are reflected in the quality of the final product.

I would also like to acknowledge my employers of the past few years, the Bermuda Civil Service and specifically the Department of Planning. My experience in the Department has given me new perspectives on this thesis that I might not have had in the university arena. I thank my co-workers who have covered my duties while I was on leave, my past and current superiors (Brian Rowlinson, Rudolph Hollis, and Peter Adwick) for their encouragement and grant of such leave, and the Head of the Civil Service for making the further education of civil servants a Government priority.

I must also express heartfelt thanks to my family for their unwavering moral support. Their pride in every step of this thesis has helped me to continually believe that it would eventually be completed. I thank my family, as well, for instilling in me a respect for knowledge and education which has undoubtedly carried me through this project.

Finally, and most significantly, my sincerest and undying gratitude goes to my wife, Suzanne, who has sacrificed so much and provided incredible support over these past years, all to selflessly ensure that I would achieve an educational goal of completing this thesis and obtaining a doctorate. This work is as much hers as it is mine and I have been extremely fortunate to have had her with me through this experience.

Abstract

The protection and enhancement of visual resources constitute an on-going challenge to the planning authorities in many communities. The crux of this challenge is to guide development towards built and natural landscape forms that will not cause detriment to an existing landscape character. To understand and cope with this problem, there is the need for a means to define and model a landscape's character, to identify methods for constructing that character definition, to create tools for storing and using such a definition to visualize its spatial manifestations, and to incorporate alternative development regulatory parameters in order to assess their impact on landscape character.

Current spatial data technologies are able to portray inventories of specific, real-world objects. While well established in the planning profession, these technologies and their attendant data manipulation tools do not easily facilitate the creation of generalized, non-specific statements that are applicable across a region. Such generalized statements regarding visual and spatial features are at the heart of descriptions of landscape character and implicit within most planning regulations intended to produce a desirable landscape character. Current spatial data tools therefore do not satisfy the stated needs of planning for landscape character.

In satisfying these conceptual, methodological and technological deficiencies, the research presented in this dissertation defines and demonstrates a theory of *landscape grammar* which formally draws parallels between the structures of linguistics and the character of landscapes. A landscape grammar defines a landscape character using a spatial vocabulary and syntax rules and can be applied to a site to generate landscape forms that embody the defined character. In this dissertation, the spatial counterparts of the linguistic concepts of vocabulary and grammar rules are formalized and implemented for use in a custom-developed geographic information system. Methods that enable the use of landscape grammars in a planning environment are presented and subsequently applied through the formal expression of planning regulations into the grammar-based model. The theory, methods and software implementation are demonstrated using a residential area of the island of Bermuda. The iterative grammatical generation of an example two-dimensional landscape scene is demonstrated with further three-dimensional representations of the results for visualization purposes. Alternative planning regulations are also incorporated into the case study grammar and resultant three-dimensional landscapes are shown. Several suggestions for future research on landscape grammars are offered in the conclusions of the dissertation.

Table of Contents

Chapter 1 Introduction	1
1.1 FOCUS OF THE RESEARCH.....	2
1.2 STRUCTURE OF THE DISSERTATION.....	5
Chapter 2 Concepts and Research for Landscape Grammars	7
2.1 INTRODUCTION.....	7
2.2 THE CHALLENGES OF VISUAL RESOURCE PLANNING	8
2.3 THE NATURE OF VISUAL LANDSCAPE CHARACTER	12
2.4 LANDSCAPE AND LANGUAGE	14
2.5 GRAMMAR STRUCTURES	17
2.6 GRAMMAR MECHANISMS	19
2.6.1 <i>Generative Grammars</i>	20
2.6.2 <i>Analytical Grammars</i>	24
2.7 SPATIAL GRAMMARS	26
2.7.1 <i>Types of Spatial Grammar Structures</i>	26
2.7.2 <i>Spatial Grammar Mechanisms</i>	32
2.7.3 <i>Architectural grammars</i>	38
2.8 IMPLEMENTATION OF SPATIAL GRAMMARS.....	40
2.8.1 <i>Computer-based Spatial Grammars</i>	40
2.8.2 <i>Models of Landscape Representation</i>	45
2.8.3 <i>Models of Knowledge Representation</i>	47
2.9 SUMMARY.....	50
Chapter 3 Landscape Grammar Theory and Definitions	52
3.1 LANDSCAPE GRAMMAR CONCEPTS	52
3.1.1 <i>Landscape Vocabulary</i>	53
3.1.2 <i>Landscape Objects and Scenes</i>	58
3.1.3 <i>Landscape Rules</i>	60
3.2 LANDSCAPE GRAMMAR PROCESSING.....	65
3.3 VISUALIZATION OF THE LANDSCAPE LANGUAGE	75
3.4 SUMMARY.....	78
Chapter 4 Implementation of Landscape Grammars	79

4.1 IMPLEMENTATION OBJECTIVES & METHODOLOGY	79
4.1.1 <i>Selection of a Development Environment</i>	80
4.1.2 <i>Benefits of Common Lisp and the Common Lisp Object System</i>	84
4.1.3 <i>The Selected Development Approach</i>	86
4.2 SYSTEM OVERVIEW.....	88
4.3 THE LANDSCAPE CLASS HIERARCHY	90
4.4 CREATING INSTANCES OF LANDSCAPE CLASSES.....	97
4.5 RULES AND RULESETS	102
4.6 THE RULE INTERPRETER	106
4.7 A SCENE GENERATION EXAMPLE	109
4.8 EXPORT OF THE SCENE FOR 3D VISUALIZATION	115
4.9 SUMMARY.....	121
Chapter 5 Landscape Grammars and Planning.....	122
5.1 LANDSCAPE GRAMMARS IN PLANNING	122
5.2 METHODS FOR LANDSCAPE GRAMMAR CONSTRUCTION IN PLANNING	127
5.2.1 <i>Scoping</i>	129
5.2.2 <i>Grammar Construction</i>	136
5.2.3 <i>Grammar Execution</i>	137
5.2.4 <i>Grammar Evaluation and Refinement</i>	138
5.3 USE OF GRAMMAR VISUALIZATIONS IN PLANNING	141
5.4 IMPLICATIONS FOR PLANNING	143
5.5 SUMMARY.....	145
Chapter 6 The Bermuda Visual Landscape	146
6.1 THE LANDSCAPE OF BERMUDA	146
6.1.1 <i>Natural Features</i>	147
6.1.2 <i>Cultural Features</i>	149
6.2 THE PERTINENCE OF LANDSCAPE PLANNING TO BERMUDA.....	154
6.3 THE PLANNING ENVIRONMENT OF BERMUDA.....	155
6.3.1 <i>Bermuda's Planning History</i>	155
6.3.2 <i>The Current Planning Environment</i>	157
6.3.3 <i>Landscape Planning in Bermuda</i>	158
6.3.4 <i>Adoption of Landscape Planning Technologies</i>	161

6.4 SUITABILITY OF THE BERMUDA RESIDENTIAL LANDSCAPE FOR LANDSCAPE GRAMMAR USE	162
6.5 SELECTION OF A CASE STUDY NEIGHBOURHOOD	165
6.6 SUMMARY.....	168
Chapter 7 Construction of a Bermuda Grammar.....	169
7.1 KNOWLEDGE ACQUISITION.....	169
7.2 GRAMMAR CONSTRUCTION AND EXECUTION	172
7.3 VOCABULARY CLASSES.....	179
7.4 GRAMMAR RULES.....	182
7.4.1 Roads	184
7.4.2 Land Parcels.....	188
7.4.3 Houses.....	193
7.4.4 Driveways.....	197
7.4.5 Walls and Hedges.....	200
7.4.6 Trees.....	204
7.5 3D VISUALIZATIONS.....	208
7.6 PLANNING SCENARIOS.....	215
7.7 DISCUSSION.....	219
7.8 SUMMARY.....	227
Chapter 8 Evaluation and Conclusions.....	228
8.1 EVALUATION OF RESEARCH OBJECTIVES	228
8.2 CONTRIBUTIONS OF THE RESEARCH.....	230
8.3 FUTURE RESEARCH.....	232
8.4 CONCLUSION	236
Appendix A Functions of the Landscape Grammar System	238
Appendix B Application of the LGS System to Other Spatial Grammars	257
Appendix C Sections relevant to the “Details of Planning” from the 1992 Bermuda Planning Statement.....	260
Appendix D Rules of the Southcourt Avenue Grammar	262

List of Figures

Figure 2.1	The Generative Grammar Interpretation Process.....	21
Figure 2.2	The Analytical Grammar Interpretation Process.....	25
Figure 2.3	A String Grammar Rule from an L-system	27
Figure 2.4	A Graph Grammar and Example Application.....	28
Figure 2.5	Shape Grammar Rules and Example Application	29
Figure 2.6	Use of Cells to Represent Discrete Objects and Continuous Spatial Variables	31
Figure 2.7	Transformation in Matching Spatial Grammar Rules.....	33
Figure 2.8	Context-Free and Context-Sensitive Spatial Grammar Rules.....	34
Figure 2.9	Cell Neighbourhoods in Cellular Automata	34
Figure 2.10	Labelling in Spatial Grammar Rules	35
Figure 2.11	The Problem of Shape Emergence in Spatial Grammars	35
Figure 2.12	Processing Cellular Automata.....	36
Figure 3.1	An Object-Type Hierarchy for Built Shelters.....	54
Figure 3.2	Inheritance of Attribute Definitions.....	56
Figure 3.3	An Alternate Class Hierarchy for Built Shelters	57
Figure 3.4	The Landscape Grammar Interpretation Process	66
Figure 3.5	Identification of the Set of Matching Rules.....	67
Figure 3.6	Identification of the Matching Sets of Objects for a Rule	68
Figure 3.7	Serial Selection of Rules and Objects for Firing.....	70
Figure 3.8	Firing of Selected Rules on Selected Objects.....	74
Figure 3.9	The Landscape Grammar Interpretation Process with Formulae	75
Figure 3.10	Nested Grammar Interpretation	76
Figure 4.1	The Components of Object-Oriented Landscapes.....	81
Figure 4.2	Components of the Landscape Grammar System.....	88
Figure 4.3	Functional Architecture of the Landscape Grammar System	89
Figure 4.4	Main Interface of the Landscape Grammar System.....	90
Figure 4.5	Menu Structure of the Landscape Grammar System.....	91
Figure 4.6	Landscape Grammar Class Hierarchy	92
Figure 4.7	Details of the Spatial Level Classes.....	93
Figure 4.8	The Class Builder Tool	96
Figure 4.9	Output from the Class Grapher Tool for Point, Polyline and Polygon Classes.....	97

Figure 4.10	Examples of Instances of Landscape Classes	98
Figure 4.11	Details of the Grid Class	101
Figure 4.12	The Incorporation of Raster Data into a Scene.....	102
Figure 4.13	The Rule Editor Tool.....	105
Figure 4.14	Interpretation Module of the LGS.....	107
Figure 4.15	The LGS Rule Stepper Dialog.....	109
Figure 4.16	Iterative Modification of a Scene	113
Figure 4.17	Export of Landscape Grammar and Scene	116
Figure 4.18	Use of Elevation Data.....	118
Figure 4.19	Roof Cutting Objects	119
Figure 5.1	Landscape Evolution and Landscape Grammar Elements.....	125
Figure 5.2	The Landscape Grammar Development Process.....	128
Figure 5.3	Scoping in Landscape Grammar Development.....	131
Figure 5.4	Identification of Grammar Components.....	132
Figure 6.1	Main Commercial Centres and Road Network of Bermuda	147
Figure 6.2	Aerial photomosaic of Bermuda.....	149
Figure 6.3	Development Zones from the Bermuda Plan 1992.....	150
Figure 6.4	“Miscellaneous Lines” from Bermuda Topographic Mapping	152
Figure 6.5	Residents’ Identification of Changes in Bermuda’s Appearance	155
Figure 6.6	Significant Recent Events in Bermuda’s Planning History	156
Figure 6.7	The Evolution of the Definition of the “Bermuda Image”	160
Figure 6.8	Aerial Photograph of Southcourt Avenue	167
Figure 6.9	Topographic and Zoning Details of Southcourt Avenue	167
Figure 7.1	Aerial Views of Southcourt Avenue Looking Northwest in (i) 1990 & (ii) 2002.....	173
Figure 7.2	Aerial View Looking up Southcourt Avenue from Ocean (2002).....	174
Figure 7.3	Roadside View of Southcourt Avenue Looking North.....	174
Figure 7.4	Roadside View of Southcourt Avenue Looking South.....	174
Figure 7.5	Photographic Survey of Southcourt Avenue Properties.....	175
Figure 7.6	Landscape Data for Southcourt Avenue.....	177
Figure 7.7	Landscape Vocabulary for Southcourt Avenue	180
Figure 7.8	Initial Scene for Generation of Southcourt Avenue	183
Figure 7.9	Ruleset Organization for Southcourt Avenue Landscape Grammar	183
Figure 7.10	Generation of Road Features for Southcourt Avenue	186
Figure 7.11	Generation of Parcel Features for Southcourt Avenue.....	190

Figure 7.12	Generation of House Features for Southcourt Avenue.....	195
Figure 7.13	Distribution of Hue, Saturation and Lightness in House Colours	197
Figure 7.14	Generation of Driveway Features for Southcourt Avenue.....	199
Figure 7.15	Generation of Wall and Hedge Features for Southcourt Avenue	202
Figure 7.16	Generation of Tree Features for Southcourt Avenue.....	205
Figure 7.17	Final Working Scene for the Described Grammar Interpretation	207
Figure 7.18	Alternative Scenes Generated by the Southcourt Avenue Grammar.....	209
Figure 7.19	Aerial View of Southcourt Avenue and Generated 3D Scene	212
Figure 7.20	Ocean View of Southcourt Avenue and Generated 3D Scene	213
Figure 7.21	Roadside View of Southcourt Avenue and Generated 3D Scene	213
Figure 7.22	3D Visualizations of Grammar-Generated Scenes	214
Figure 7.23	Changes in Landscape Scenes based on Regulatory Grammar Modifications	216
Figure 7.24	Bermudian Neighbourhoods Similar in Structure to Southcourt Avenue	220
Figure B.8.1	Application of LGS to Froebel Block Designs	258
Figure B.8.2	Application of LGS to Other Spatial Grammars.....	259

List of Tables

Table 2.1 Typology of Spatial Grammar Interpretation Mechanisms.....	37
Table 3.1 Example Preconditions for Landscape Rules	63
Table 3.2 Example Consequents for Landscape Rules.....	63
Table 4.1 Relation of Object-Oriented to Landscape Grammar Theory.....	81
Table 4.2 Relation of 2D to 3D Objects.....	117
Table 5.1 Typology of Groups for Input on Landscape Grammars.....	134
Table 6.1 Zoning Composition of Bermuda under the Bermuda Plan 1992.....	151
Table 6.2 Composition of Dwelling Units by Type	153
Table 6.3 Regulations for Development in Residential 1 Development Zone.....	161
Table 6.4 Regulations for Development in Residential 2 Development Zone.....	161
Table 6.5 Spatial Data Availability for Bermuda.....	163
Table 7.1 Vegetative Species Observed at Southcourt Avenue.....	170

Chapter 1

Introduction

The high value attached to the visual landscape character of a region by its residents renders that character a resource. The need for the careful planning and management of this resource is underscored by the perceived threat of the homogenization of landscapes by globalization. Planning agencies are increasingly required to plan for the preservation of existing or creation of distinctive landscape character. This is especially true of jurisdictions that have small and limited land areas that embody a distinctive and valued character. Such places value highly the visual appearance of the landscape, not only for reasons of cultural identity but also for the economic purposes of a marketable and appealing image for the attraction of tourism and other business.

In order to manage the preservation or creation of distinctive and pervasive landscape character, local planning authorities must be able to define those features that contribute to the definition of character. In addition, they must write policies that will influence development in such a way as to produce the desired character of landscape without dictating the specific form of each feature. While there has been much study of perceptions of the aesthetic quality of particular landscapes, the development of methods to define a generalized landscape character and ultimately to test its consequences in the landscape have not been forthcoming.

This dissertation examines landscape character by articulating the metaphor of landscape as language. In the same sense that sentences are not random arrangements of words, so too are landscapes more than random collections of physical objects. The letters and words of a body of text may be associated with a particular language if they belong to that language's alphabet and vocabulary, and if they are arranged in ways that are consistent with the language's grammar rules. It is because of this conformity that the text of a given language can be read and understood. Similarly, if a landscape is to be associated with a particular visual character, its constituent objects should be representative of those typically found in the region and arranged consistently with other configurations. Thus, in linguistic terms, landscape character contains a vocabulary of physical objects and rules of spatial syntax. Without elements of syntactic order, landscapes lose functional and interpretive value for residents and visitors and hence they lose their associative character. Through a recognizable spatial order, landscapes can convey their meaning and convey the region's distinctive identity.

The vocabulary and syntax rules associated with a landscape's character are presented in this dissertation through the concept of a *landscape grammar*. Scenes constructed using a vocabulary and syntax rules exhibit the character that is embodied in the associated grammar. The vocabulary and rules have therefore not only a definitional value for describing landscape character but also a generative value for directing the construction of new landscapes, whether they are actual, physical scenes or visualized landscape simulations for use in visual landscape planning.

Just as the understanding of a spoken dialect's patterns may allow a person to express their own meaningful statements in that dialect, landscape planners must similarly pursue an understanding of a region's character in order to effect new landscapes that are consistent with, and thus meaningful in relation to, the existing visual syntax. The definition of a region's visual landscape character comprises general statements about the visual and spatial features that typically occur in the area. Local planners must develop an appreciation of these generalities in order to encourage future development that maintains desirable landscape characteristics. They must become familiar with the local landscape 'vernacular', and learn to read and speak its 'language'. For planners, the task of understanding landscape character entails a process of analysis, while the achievement of new, conforming landscapes is undertaken through the writing and enforcement of land-related policy, regulations and codes. Planners have few tools with which to formulate planning regulations from their ideas about visual landscape character and to test the efficacy of these regulations in producing desirable landscapes. Currently, the leap from visual landscape analysis to the formulation of landscape planning regulations is facilitated by intuition about the regulatory elements that might create the desired character.

The concept of a landscape grammar can play a role in regional planning as a framework for landscape character definition, with a rule-based structure similar to planning regulations and codes. The natural and cultural syntax rules of a region's landscapes may be placed alongside regulatory rules derived from planning authorities. Thus, the articulation of a regional landscape grammar can help to smooth the transition from landscape knowledge to planning action. Current landscape planning technologies, while suitable for landscape representation and analysis, are not equipped for the representation of such generalized 'knowledge' statements about a region. Investigation of the validity of a landscape grammar, and thus the efficacy of planners' ideas about character, also calls for new capabilities in these technologies.

1.1 Focus of the Research

The central challenge of visual landscape planning is to guide development in such a way that it will not cause detriment to existing landscape character. It is clear that in order to design measures that

preserve a landscape's character, the components of that character must be understood. Further, it is clear that in order to address this challenge, there is a need for (i) a means by which landscape character can be defined and modelled, (ii) methods for constructing character definitions in a planning context, (iii) tools for storing and making use of a landscape character definition, and (iv) methods for the incorporation of proposed regulations in order to assess their impact on landscape character. None of these needs have, however, been systematically examined in planning literature. The landscape character of a place is less subject to definition than description, in which the value of the landscape deconstruction is attributable to the quality of the observer's verbal articulation of design. The technologies used typically for landscape planning, namely geographic information systems (GIS) and computer-aided design (CAD), are also deficient in addressing the definition of landscape character. GIS and CAD are designed to record the locations and attributes of specific objects, while character is expressed in terms of generalizations.

This dissertation is specifically concerned with the development and implementation of this concept for the purposes of the planning of visual resources. The use of grammar formalisms is proposed first as a theoretical framework for defining landscape character. A philosophical context is also provided for the appropriate incorporation of a landscape grammar into planning activities. An implementation of a landscape grammar is then presented in the form of a computer-based production system, that generates example landscape scenes in two and three dimensions for examining the visual efficacy of the grammar.

The main goal of this dissertation is to develop a means of defining landscape character to allow the investigation of possible landscapes that conform to a prescribed or desired definition. To this end, the dissertation objectives are:

1. to advance the capacity of conventional geographic information technologies for landscape character planning;
2. to develop a theoretical framework that allows landscape character definition;
3. to construct a computer-based implementation that integrates the landscape character theory into conventional geographic information technologies and planning practice; and
4. to allow, in that implementation, the visualization of the potential consequences of a landscape character definition.

It will be shown that the implementation of the landscape grammar concept addresses these objectives and that this concept may be integrated within planning processes. The concept has particular contributions to landscape technology, planning methods and research on spatial grammars.

The research is based on a number of underlying premises. The first is that the visual landscape is important and its inhabitants value an identifiable and meaningful visual environment in their place of

residence. As a consequence, visual landscape planning and management are valuable pursuits that can protect against detrimental changes to visual character and thereby maintain the quality of life for residents. It is also asserted that visual landscape character is definable in terms of its elements. While certain intrinsic qualities of a landscape, such as rhythm and form, are describable in terms of the whole rather than the sum of its parts, it is the collocation and arrangement of a multitude of landscape elements that visually represent the collective landscape character. This atomistic perspective provides the generalizable landscape elements that together comprise the vocabulary with which to describe a landscape's character. Furthermore, by articulating and examining the generalizations that are believed to comprise a landscape's character, a more precise understanding of that character may be gained. Such 'landscape knowledge' is certainly important for planning officials to obtain, if they are to write policies and regulations that preserve their locality's visual character.

The dissertation also rests on the principle that complex data, especially those that involve multiple perspectives, may be better understood through the use of visualization. This is especially applicable for visual landscape information. To understand the visual consequences of certain activities, a visual representation is more informative than a numeric or verbal one. Computer-based modelling and graphics technologies are currently advanced enough to undertake such a task. In addition to using computer graphics technology, the dissertation relies on techniques originating from artificial intelligence, specifically knowledge representation. While computer-based knowledge representation methods do not currently replicate human knowledge abilities, significant and useful approaches have been developed. Hence, this dissertation is based on general knowledge-based approaches, and is thus subject to their limitations.

The impetus for the research came initially from the planning environment of the island of Bermuda. Visual landscape character has been an ongoing area of concern for the Bermuda Department of Planning. Bermuda's traditional landscape character, with both natural and cultural elements, is deeply rooted in its colonial past and present and is highly treasured. Its value is derived from its roles not only as a fundamental icon of cultural identity, but also as a major attraction for the tourism industry. Since 1990, the Bermuda Government has been moving closer to the implementation of an island-wide GIS database. This research was derived originally from a desire to investigate how conventional GIS technology could help the Department plan for the island's landscape character. Previous research (Mayall, 1993) advanced the means to develop efficient three-dimensional landscape models from a two-dimensional GIS database. This dissertation advances that research by developing means to define landscape character in terms of classes of elements and their spatial arrangements.

1.2 Structure of the Dissertation

Following this introductory chapter, in Chapter 2 several domains of knowledge are discussed that are relevant to the dissertation. The importance of the visual landscape and the challenges related to its management are reviewed, identifying the relative lack of methods for the definition of the landscape character of a place. The concept of 'landscape character' is defined explicitly and an analogy drawn between landscape structure and the syntax of verbal language. This analogy leads to the discussion of grammatical structures and mechanisms and their application to spatial objects. The computer-based implementation of spatial grammars is then described and related to existing technological approaches to landscape representation and artificial intelligence.

Chapter 3 focuses specifically on the theoretical definition of landscape grammars as a series of formalisms. The concepts of a landscape vocabulary, objects, scenes and rules are formalized using set theory notation and illustrated with examples. The processing of a landscape grammar is defined formally using flowcharts, detailing how scenes are iteratively constructed and the range of constructions is visualized.

Chapter 4 carries forward the observations from Chapter 2 and the formalisms from Chapter 3 into the implementation of a computer-based landscape grammar interpreter. Following an outline of the specific objectives for the implementation, the relevant development options for landscape grammar interpreters are examined, with further discussion of the selected development environment. The architecture and functionality of the implemented system are then presented. Finally, an illustrated example of a simple landscape grammar and scene construction is provided, as well as demonstrations of the capability of the implementation to operate in the manner of other spatial grammar interpreters.

Following the description of the implemented landscape grammar system, Chapter 5 defines operational methods and considerations for constructing a landscape grammar for a region. A process is detailed for the identification of relevant landscape characteristics, articulation of them as an executable grammar, execution of the grammar using the interpreter, and then the evaluation of the outputs in order to refine the grammar and restart the process of grammar development. Specific uses for a landscape grammar in a planning context are described as well as the integration of the grammar into the process of amending planning regulations. The chapter concludes with suggestions of the possible implications for the planning profession of using landscape grammars.

Bermuda, and particularly the Bermudian residential landscape, are used as a case study for the landscape grammar implementation. Chapter 6 describes first the natural and cultural elements of Bermuda's landscape character. The importance of landscape planning on the island and a synopsis of Bermuda's planning environment are then presented to establish the regulatory context. Finally, the

chapter reviews the features of the Bermudian residential landscape that make it suitable as a case study for landscape grammars and identifies specific neighbourhoods that were subjected to detailed study.

Chapter 7 presents the landscape grammar that was developed for a selected study neighbourhood in Bermuda. The data collection methods are described followed by details of the vocabulary and rules of the final grammar. Scenes generated by the landscape grammar interpreter are presented in two and three dimensions. These generated scenes depict the existing landscape character of the study neighbourhood. It is then demonstrated how a landscape grammar can be modified to incorporate planning regulations, generate new landscape scenes, and visualize the possible visual consequences of such regulations. The strengths and weaknesses of the theory, implementation and application of landscape grammars as represented in the dissertation are then reviewed, given the experience of the Bermuda grammar case study.

Chapter 8 concludes the dissertation by first assessing the research in terms of the original research goal and objectives. The contributions of the dissertation to planning, landscape representation technology and grammar research are then identified. Finally, proposals are made for areas of future research on the landscape grammar concept and implementation.

Chapter 2

Concepts and Research for Landscape Grammars

In demonstrating the theoretical and practical foundations of the topic of landscape grammars, this chapter traverses several bodies of literature. First, the value of visual landscape resources is established and the fields of visual resource management and visual resource planning are introduced. This leads to a discussion of the specific challenges of the latter. Place identity and landscape character are then discussed and defined. In addressing the challenges of visual resource planning, a link between landscape and language is introduced as a precursor to grammar theory. Grammars are introduced first as definitive structures and then as generative and analytical mechanisms, with specific reference to linguistics. The adaptation of grammar structures and mechanisms to spatial objects is then presented, with relevant applications that include architectural grammars. Since computer-based visualization is becoming a common tool for landscape planning, the levels of appropriateness of current technologies for establishing a grammar framework are reviewed. It is then argued that a computer-based grammar implementation is essentially a knowledge-based application and thus in the domain of knowledge-bases and their applications to spatial technologies.

This review helps to define problems associated with visual landscape planning and introduces bodies of literature that are used in crafting a problem solution. In Chapter 3, the more specific theoretical underpinnings of landscape grammars and the philosophical context for their use in planning are examined.

2.1 Introduction

The maintenance and development of a desirable visual landscape character is an important task in regional planning. It requires planners not only to understand and describe their regional landscape, but also to translate their knowledge into policies, regulations, and codes. This is an essential problem for the planning of visual character, since few resources exist for planners to test their knowledge of the landscape and demonstrate their ability to produce landscapes of the desired character.

The concept of a landscape grammar holds potential as a solution for this problem, as a landscape grammar may be used to formalize landscape knowledge in a manner that allows it to be applied objectively to landscape planning. Much as a linguistic grammar specifies words and their

arrangement into meaningful sentences, a landscape grammar transforms landscape objects and their configurations into meaningful places. From this, visual landscape character can be defined using a set of definitions of objects that are typically found in a region (for example, various species of trees) and a set of rules that specify how such objects are typically located within a region's scenery (such as on hills or in valleys).

Once a landscape character is described in this manner, it may be used to produce simulated scenes to represent real world landscapes. The scenes can be evaluated visually in terms of the extent to which they conform to the intended character of an area. This is, in essence, the application of planners' landscape knowledge and how well it has been translated into reality through the application of planning regulations and rules. Since a landscape grammar stores planners' knowledge of landscape character, the cycle of scene generation and grammar modification entails a process of knowledge application and refinement.

In this context, the consequences of planning regulations are only apparent many years after their conception and enforcement and are often irreversible. Hence, the ability to evaluate the possible consequences of regulations before they are implemented offers a powerful tool for planners. In short, it allows them and others (such as members of the public) to visualize the future character of the landscape before actual physical change takes place. On the one hand, there is nothing terribly new about this concept as architects and landscape planners have, for many years, used physical models and drawings to allow future landscapes and buildings to be visualized before development. However, on the other hand, the concept of using a grammar of landscape ('syntax' rules and a vocabulary of 'words') with which to construct and then evaluate an unbuilt scene is a new and potentially powerful concept for use in land development research and practical applications.

2.2 The Challenges of Visual Resource Planning

The *landscape*¹ becomes a resource when its capacity for satisfying human wants and needs is recognized. The visual resource is composed of those landscape features that may be viewed and evoke reactions. The ascription of value to landscapes leads to further questions about the degree of value a particular landscape holds, either absolutely or in relation to other landscapes. In this way, landscapes and their elements are studied, evaluated and rated using subjective notions such as 'beauty', 'scenic quality', and 'degradation'. Thus, the visual resource may be referred to as the '*scenic resource*', which emphasizes a judgment of quality and implies attention to relationships between design elements

¹ Holistic definitions of the term emphasize a full sensory experience, including socio-cultural and biological qualities. However, this dissertation concentrates on the visual aspects of a landscape experience.

(Gussow, 1979). It is here that the concept of the visual landscape intersects the domain of the visual arts.

Although visual landscape quality constitutes an important component of environmental quality (Litton, 1982), the visual quality of a region as a whole is not always considered in planning practice (Lynch, 1976). It is more often those scenic resources of exceptional aesthetic quality (for example, the national, state/provincial and local parks in North America, and the Areas of Outstanding Natural Beauty in the United Kingdom – Parks Canada, 2000; National Parks Service, 2001; DETR, 2000) that are considered worthy of protection and those of poor quality that are earmarked for rehabilitation or change. Notwithstanding this, the value of the general visual landscape should be considered, because of the important roles it plays in the lives of its residents. Such roles involve elements of cultural, psychological, social, aesthetic, functional, ecological and economic dimensions.

The cultural identity of a society can be tied to the visual elements of its surrounding landscape. Such elements may be historically significant in the area (for example, forts), hold some functional significance (stone walls), or may simply be a familiar part of residents' shared everyday experience (marketplaces). Traditional architectural styles, historic buildings or even whole villages provide residents with a visible continuity between the past and the present (Raitz & Van Dommelen, 1990) and, as noted by Bourassa (1999, 109), "once a landscape acquires meaning for a cultural group, that group will seek to perpetuate that symbolic landscape as a means of self-preservation". It has also been suggested that a stimulating visual environment heightens the observer's attention and thus the potential depth and intensity of human experience (Lynch, 1960).

An easily understood visual environment allows us to function effectively within it. If an area has a high level of clarity in its organization, then orientation within the area is more easily facilitated than if the area is chaotic (Lynch 1960). In addition, the visual landscape is a source of copious amounts of information and, consequently, it can influence local communication by clarifying verbal discourse with meaningful symbols and collective memories (Cullen, 1971; Eckbo, 1969). The visual landscape also holds aesthetic value in the perspective of an object of beauty or art. Cultural landscapes viewed as art hold roles for personal or community expression (Bourassa, 1991).

Due to the high aesthetic value placed on natural objects, a scenic landscape often holds heightened ecological value relative to developed environments. It is commonly recognized that mountainous, forested areas, open spaces and large bodies of water are found visually appealing. Visual quality objectives, therefore, often coincide with the preservation of natural features such as trees, soil and natural topography. Conversely, the conservation of nature often enhances visual quality. This also holds for designed ecological landscapes such as urban parks.

Finally, a community can find economic value in the high visual quality of a region. The visual

character of a city at least partly defines its image/identity. As a marketing feature, it can be used to attract industrial and commercial interests. The tourism industry is, in particular, highly affected by the visual beauty of an area. In most vacation activities, tourists are interested in a different aesthetic experience than that of their home environment. The visual quality of the landscape therefore becomes pivotal for promotional campaigns and in cases where the economy depends on tourism, building controls and environmental management are required (Dilley, 1986).

Thus, visual landscapes can hold significant value for residents and visitors alike. However, the visual resource, as other resources, is subject to pressures from development. Place uniqueness has become threatened by suburban development, the growth of international tourism, and, more recently, economic globalization. Development on a large scale can endow places with similar appearances, independent of location and culture. Competition in the tourism industry can coerce a destination to act and look like its competitors, reducing landscape quality to a set of core elements (“sand, sea and sun” for example). Globalized commerce has resulted in foreign businesses applying preconceived notions of design to new locales. Under such conditions, unique place identities such as those embodied in designated world heritage sites, are considered valuable and precious resources that should be preserved.

Planning activities, themselves, can also pose a threat to distinctive landscape character. Hough (1990, 2), for example contends that “while traditional vernacular landscapes usually represent the diverse character of different places, conscious planning and design tend to negate those differences”. One of the expected tasks of planners, therefore, is to create localities by averting “placelessness” and providing “imageability” (Relph, 1976; Lynch, 1976). Hence, since the visual resource is valued as well as vulnerable, it has become a focus of management and planning activities. Origins of these practices are found in urban design and environmental planning disciplines. The design of urban sites, as architecture writ large, has historically been concerned with high visual and experiential quality in built places (as well as their functionality). The more recently born environmental movement related visual quality to the environmental quality of natural landscapes and instituted this consideration as part of environmental assessments (NEPA, 1970).

Visual resource management (VRM) is a well recognized term in the landscape literature. It generally involves the process of identification, assessment, allocation and manipulation of the visual resources at a site. These resources are sometimes analyzed in terms of their visual and spatial structure using factors such as visibility, distance from the observer, angle of observation, scene depth and light (Higuchi, 1983). VRM activities, such as visual impact assessment, viewshed analysis, estimation of visual absorption capacities and the identification of visually sensitive areas or obtrusive features are relatively short-term and focused on the design of a specific site (Sheppard, 1989; Smardon et al., 1986). When visual impact assessments of proposed development projects are performed, they are often part of a broader

environmental impact study (Smardon et al., 1986; Struckmeyer, 1994).

Since VRM activities are reactive and focused on specific sites, they address neither the long-term nor the regional focus of the planning process. Hence, there is a need to distinguish between VRM and *visual resource planning* (VRP). VRP can be defined as working towards the achievement or maintenance of a desirable state for a region's visual resources. In contrast to VRM, VRP is focused on the long-term visual identity of a region as a whole, rather than on specific views and objects. VRP informs VRM strategies by establishing the identity and character intended for a place.

In regions where there is an absence of an established visual identity, VRP activities must ascertain which landscape features are considered desirable for a place's identity. Studies of observers' perceptions of vegetative, water and architectural landscape features enlighten planners and managers as to what is perceived as "beautiful" (or of "high quality") by residents and visitors. The values attached to particular landscape elements dictate which features will be encouraged in the production of a regional image. The findings of perception studies may be instituted in visual quality objectives, which state a visual condition to be attained in the course of development (BC Ministry of Forests, 1996).

However, it is often the case that communities interested in VRP already possess a valued visual identity that they want protected. The aim of VRP, then, is to ensure that the existing landscape identity is maintained and encouraged in future development. In this process, planners must first define an acceptable conception of the region's visual identity and then write planning policies and regulations that call for developments to exhibit the intended visual character. In the first instance, planners must derive the landscape vocabulary and spatial syntax that defines the region's character. In the second instance, they are required to prepare new syntax rules, to be adopted as policy and enforced as regulations that will ensure the continuance of the region's distinctive character.

Few researchers have focused on the question of how to define the visual character of a region and still fewer have suggested how to translate such a definition into planning regulations. The latter interpretive step can involve a high level of uncertainty. While planners may have some confidence, through the use of public surveys, that their definition of a region's character is at least acceptable, the translation of that definition into planning regulations carries no such degree of assurance. Subjective modes of assessment are an alternative to more objective analyses and are used by decision-making boards or councils to determine the acceptability of individual development proposals. However, discretionary project-based activities are still based on the regulatory concepts of a regional plan. There is currently no way to verify whether the definition and subsequent enforcement of planning regulations will result in landscapes with the desired character, or even prevent undesirable, but conforming, landscapes to be produced.

Therefore, the challenges facing planners in planning for landscape character are (i) the definition

of a landscape character and (ii) the testing of the ability of such a definition, and subsequent planning regulations, to produce desirable landscapes. It follows from these observations that there is a need for (i) a framework for character definition and (ii) a mechanism for testing the efficacy of a definition. With the fulfillment of these needs, planners could better examine their ideas about landscape character by capturing them in a definitive framework and utilizing the suggested mechanism to visualize landscapes that might result from such ideas being encoded as planning regulations. By repeating this process, planners can arrive at more refined definitions and regulations. As a result of these better articulated definitions, decision-making boards have the ability to arrive at better informed decisions because the landscape character which they are empowered to conserve is subject to less conjecture. In addition, designers can better incorporate the visual objectives for an area in the early stages of their designs. The achievement of such a planning environment is by no means trivial. However, an appropriate starting point for addressing these challenges is the definition and nature of landscape character.

2.3 The Nature of Visual Landscape Character

Physical landscapes provide the contexts within which all life coexists. Through prolonged interaction with a place, humans learn to recognize its visual cues and associate meanings with them. Such cues may be manifest in objects, or groups of objects, as they occur in the landscape. It is these distinguishing traits that make places identifiable and that give them meaning. Landscape character is important in this context because of its contribution to establishing place identity.

Place identity is comprised of those qualities that make a place unique. It is highly valued by its residents and planners are often responsible for developing and promoting a distinctive identity. The qualities of a place can be derived from various sensory and emotional experiences that become associated with personal, social or cultural meanings. Such holistic definitions of 'landscape' are found in the literature of geographers and ecologists (Tuan, 1974; Relph, 1976; Porteous, 1982; Cosgrove, 1984; Bourassa, 1991; Motloch, 1991). The urban design literature also describes landscapes in terms of sensory encounters, often given by elaborate and qualitative descriptions (Jacobs, 1993). Landscape artists provide visual expressions of landscapes that also attempt to distill those essential elements that make a place unique (Raban, 2001). Paintings from the nineteenth century French Impressionist School provide a particular example of this effort, as does the Canadian Group of Seven in their renditions of nature in canvases predominantly drawn from the scenery of northern Ontario.

While such descriptions of 'place' may provide a vivid image, they do little for the development of planning tools to address the challenges noted above. Holistic definitions are not operationalizable since the incorporation of many different factors compounds the complexity of the planning problem.

Likewise, the designers' approach does not translate well to planning codes because the descriptions of place are often loosely worded to evoke a sensory experience. Hence a reasonable way to deal with this problem is to focus on the visual aspects of place, since landscape experience is primarily visual and the visual and physical aspects of landscape are most relevant to planning.

An alternative, and perhaps more practical, way to define place identity is in terms of singular and recurrent visual qualities. Singular qualities serve as icons of a place. It may be that a single component in the landscape is so visually significant, due to its size, position, colour, shape, etc., that a place may be identified from that component alone. For example, a picture of the Eiffel Tower might symbolize Paris, or the Empire State Building, New York. Built icons are often large, designed by a specialist, built with a high quality of materials and workmanship and are the property of the elite or the state (Matthews, 1994).

However, recurrent components of a place's identity are often the products of ordinary individuals in the local community. These common elements are the distinct objects, groups of objects and their qualities, found typically in the visual scenes that one might encounter while travelling within a place. Landscape elements of this type are often referred to as "vernacular" (Matthews, 1994; Rapoport, 1969, as cited in Motloch, 1991). The recurrent aspects of a visual identity are more relevant to planning than icons because, like planning regulations, they apply across whole regions². They are both generalizations. It is the recurrent qualities that comprise the visual landscape character of a place. Therefore, it is visual landscape character that is more relevant to planning than a singular icon. In addition, this model of landscape character coincides well with the above definition of VRP. Both maintain a 'regional' scope, and VRP policies and regulations are applied repeatedly across a region.

It may be argued that since a landscape character grows organically, a *laissez-faire* approach to VRP should be adopted to question the utility of a landscape character definition. While it is true that a regional landscape character evolves over time, it is not so rapid as to render the planning of it futile. In built environments, the rapid changes to character are often small-scale, because small changes typically involve less investment and are therefore incremental in nature. Moreover, existing planning controls may intentionally tend to inhibit the larger and more significant landscape changes, although controls are also caught unprepared for some types of events such as the advent of 'big-box' retailing. In natural environments, rapid changes such as fire or vegetative disease can be catastrophic and impossible to plan for. Despite the occurrence of chaotic events, Western rationalist thought favours the careful planning and management of its resources. Thus, despite its evolving nature, a landscape character definition still

² For the purposes of this dissertation, a 'region' refers to an area large enough that a traveler moving through it encounters many different views. In this context, a region could be comprised of a small town, an urban neighbourhood, a large residential community, or an expansive natural area.

has utility in VRP.

Landscape character is composed of those elements that recur throughout a region. As a result, since landscape character is an important determinant of a distinctive place identity, VRP activities should focus on such reoccurring elements and at a regional scale. Lynch (1976, 4) confirms this observation for sensory planning as a whole: “sensory quality ... must be considered in planning for an entire inhabited region, and for the everyday environment experienced in the whole range of daily action”. Given this definition of landscape character as a composition of recurring elements, the task of describing these elements and the nature of their recurrence remains to be addressed. To achieve this, a comparison of the patterns of landscape to the patterns of language is presented in the next section.

2.4 Landscape and Language

Language is a useful metaphor for examining landscape character. Like language, a landscape can be read and interpreted, its elements can have symbolic meanings and there is structure in the arrangement of its elements. Residents of a place can ‘read’ and understand the physical configurations of objects in their locality as well as use that knowledge to create new and meaningful constructions, much as they form sentences with their knowledge of the local language and dialect. This analogy of language and landscape is based on the premise that both contain patterns and that these patterns convey meanings. Alexander (1977, 1979) made great use of this premise in his seminal works on the use of patterns in building communities.

In language, words are arranged in linear patterns to form sentences. If the reader is familiar with the vocabulary and the accepted rules of syntax of the language, then these patterns are recognizable as valid sentences of that language. Such familiarity can also allow the reader to compose other valid sentences and to communicate verbally with others who possess a similar familiarity with the language. The salient difference for landscapes, and the central tenet of this dissertation, is that the vocabulary is mainly made up of physical objects and the patterns are visual or spatial in nature. The patterns of objects are manifested in three-dimensional landscape scenes, rather than linear sentences. If observers are familiar with a regional vocabulary of objects and their typical spatial arrangements, they can associate a sample scene with a regional landscape character. Observers can also construct new landscape scenes that could be recognized by others as examples of that character.

The landscape-language metaphor is frequently used in relation to designed rather than ‘organic’ landscapes. Architectural researchers often refer to the ‘visual languages’ of a specific designer, design movement, or folk traditions. Consequently, the word ‘vernacular’ has often been applied to architectural as well as linguistic styles (Rapoport, 1969, as cited in Motloch, 1991; Matthews, 1994). Urban designers

and other researchers dealing with the broader landscape have also made use of this metaphor. Motloch (1991), for example, suggests that a “culture’s autobiography” is read by means of its landscape. Alexander (1977) introduced the concept of a “pattern language” which can be used as a prescriptive means for the successful design of towns/cities and buildings. While the linguistic concept has mainly been applied to city- and townscapes, it can be argued that natural landscapes also have a syntax and structure which conveys meaning to observers. Ecological habitats of a particular type often show similar constitutions and structure (Roberts et. al., 2000, 2002).

It is possible to extend the metaphor to semantic interpretations of landscapes. For example, words in a sentence have meanings. Similarly, objects in a landscape, particularly cultural landscapes, can be symbolic of more abstract concepts. For example, a garden wall may serve as a property boundary marker, a pet enclosure or a privacy enhancement. Equivalently, a wall can be regarded as appealing (aesthetic) or as repelling, conveying a utilitarian or functional message (‘keep out’). A person literate in a local landscape understands the significance of such objects. Jakle (1987) maintains that “landscape semiosis ... the study of landscape as a symbol system”, is well established in Europe. In this field, “the constitutive elements of landscape are seen as analogous to constitutive linguistic elements or words” (Jakle, 1987, 14). Meinig (1979) emphasized the role of observer bias in the interpretation of landscape meanings by discussing multiple perceptions of a landscape, as well as proposing several caveats for those attempting to ‘read’ landscapes. Landscapes may also be considered to have “legibility”, a measure of the extent to which it may be recognized and understood by observers (Lynch, 1960).

In *The Language of Landscape*, Spirn (1998) makes lengthy reference to the landscape-language metaphor. She likens the general landscape to a “vast library of literature”, within which ongoing dialogues take place. It has rules of grammar that govern the forms of landscapes and allow meanings to be shared. Some of these rules are “specific to places and their local dialects, others universal”, while others may be imported from other locales. Landscape grammars are descriptive rather than prescriptive, and “embody collective wisdom accumulated over generations of life in a particular landscape” (Spirn, 1998, 188). While interesting and relevant, Spirn’s (1998) discourse on the grammar of landscape is more reflective and expressive than the systematic and technical approach that is the subject of this dissertation.

It should be clear that the investigation of superficial spatial arrangements of objects in a landscape is a less thorny problem than interpreting the meanings attached to those objects and patterns by residents. The two can be considered very different problems with different methodologies. The former for example, involves visual and spatial analysis, while the latter considers social, psychological, cultural and historical investigations. Notwithstanding the potential for examining the symbolic meanings of landscape patterns, this dissertation focuses on the surficial spatial syntax of landscapes.

This focus more adequately addresses the first stated challenge of VRP, namely the need for character definition.

The language-landscape metaphor may also be related to the roles of different individuals or groups in the planning process. Unlike linguists who must both read and write/utter sentences, planners do not directly 'write' the landscape. In VRP, planners are first required to be able to read the landscape character of their locality in order to recognize the spatial and visual patterns that exist in their region. In formulating new regional plans, they then decide on the type of 'message' that landscapes will convey. The 'message' is in essence the desired identity for the region, such as an urbanized hub of high-tech activity, a pedestrian-friendly, socially-oriented New Urbanist community, or a historical place with deeply rooted traditions expressed in a built vernacular. While visual resource planners determine the regional landscape message, architects and landscape architects are the 'writers' of the landscape patterns, designing a detailed arrangement of objects for each site – the landscape equivalent of 'pragmatics' (Spirn, 1998). The physical construction of these designs establishes them as 'written' into the regional landscape.

Under modern statutory planning contexts, a local planning authority must assess the proposed designs of architects using the regulations provided in the region's development plan. The task for visual resource planners is to write plans that encourage new development that embodies the desired visual identity, or, in linguistic terms, to encourage the design of landscape 'sentences' that express the desired 'message'. The plans, in effect, become a sort of grammatical rulebook, specifying the spatial and visual syntax (or landscape character) expected for the region. Sometimes it is human safety or efficient functioning rather than desirable visual features that determines the content of the rules, such as minimum road-widths and structural regulations. If planning regulations are written well, they allow the creation of functional landscapes without conflicting with the desired visual identity. By enforcing the assessment of development proposals, planners engage in an 'editorial' role in the review of the proposed designs. Thus, the input of various experts and interest groups, planners can assess and recommend revisions, or 'edits', to the draft designs before physical construction is permitted.

The roles discussed above are based on a modern statutory planning environment for a developed regional landscape. In a less modern environment, the editing of design proposals may be less rigorous. Architects may enjoy much more creative expression in their designs. Special institutional projects, such as museums or galleries, may also result in the intentional relaxation of landscape syntax rules. However, the absence of planning regulations does not necessarily result in less regular syntax. Developing countries may exhibit more vernacular settlements based on limited building materials and technology rather than planning regulations. In this case, the rules of syntax are more culturally derived than regulatory. With entirely ecological landscapes, one might consider the landscape writers and editors

to be the complex forces of nature.

Thus, the landscape-language metaphor offers a conceptual framework for defining landscape character, addressing the first challenge for VRP, namely the definition of a region's landscape character. The operationalization of this analogy is achieved through the definition of a 'grammar', comprised of the vocabulary and syntax rules which can be used to create the language of a landscape.

2.5 Grammar Structures

There are no apparent references in grammar-related literature to a generally applicable 'grammar theory'. However, given the wide range of applications of grammars, it is appropriate to refer to grammatical descriptions of various phenomena. A grammar theory is defined here as the assertion that a vocabulary of objects and a set of transformation rules can be used to describe a type, or style, of phenomenon. As the term 'grammar' suggests, the most obvious application is to linguistics, but other applications exist and are discussed later in this section. Much of the work in this and the next section is based on the early work of Chomsky (1957), who pioneered generative grammar as a theory of linguistics.

Grammar structure is in general comprised of two main components, namely a vocabulary and a set of syntax rules. The vocabulary defines the concepts, or types of objects, used in the grammar. In linguistics, a vocabulary contains words, such as 'table', 'car', 'long', 'went' or 'expeditiously'. Some vocabulary elements may refer to the vocabulary structure itself, such as the linguistic concepts of 'noun', 'verb', and 'clause'. Sentence constructions composed entirely of the former are 'terminal', in the sense that a collection of words can comprise a complete sentence (Chomsky, 1957). The latter meta-concepts cannot be used to create complete sentences, only intermediate sentence constructions that are later replaced with words. Thus, these two types of vocabulary objects may be referred to respectively as terminal and non-terminal vocabulary. The non-terminal abstractions represent intermediate, structural steps that influence the sentence structure and which are later replaced by the terminal vocabulary elements. Without these structural elements, sentences could only be random collections of words.

Grammar rules state how objects from the vocabulary can be related to each other, and thus arranged into patterns. Rules are generally of the form:

IF [precondition] THEN [consequent]

which is formally written as:

[precondition] \rightarrow [consequent].

The precondition is also referred to as the left-hand side lhs of the rule, and the consequent as the right-hand side (rhs). The following are some examples of simple rules:

a → b Rule 2.1

dog → bark Rule 2.2

x < 3 → category is LOW Rule 2.3

[Noun] → [Article][Noun] Rule 2.4

Rule 2.1 is read as ‘if a then b’, and the rest are read similarly. Rules can be used to infer an action as in Rule 2.2, or imply a new piece of information as in Rule 2.3. Rule 2.4 illustrates a linguistic rule in which an Article object is inserted before a Noun object.

A basic type of grammar rule that employs this formalism is the rewriting rule. The rewriting rule states that an occurrence of the lhs of the rule can be replaced with the rhs of the rule. In the above examples, rewriting rules may be read “if there is an ‘a’, then replace it with a ‘b’”, “replace the word ‘dog’ with the word ‘bark’”, “replace the observation that ‘x < 3’ with an observation that ‘category is LOW’”, and “replace a noun object with an article and itself”. Transformational rules are more complicated in that they restructure sentence constructions into alternate forms with the same meaning (Chomsky, 1957). For example, the restating of the active sentence “John can drive the car” into its passive form “the car can be driven by John” requires a transformational rule such as the following:

If a sentence is of the form
NP₁ + Aux + V + NP₂,
then the corresponding form
NP₂ + Aux + be + V + en + by + NP₁
is also a sentence. Rule 2.5

where ‘NP’ is a noun phrase, ‘V’ is a verb, ‘Aux’ is an auxiliary verb such as ‘will’ or ‘can’, ‘be’ is some form of the verb ‘to be’, ‘en’ stands for the past participle, and ‘by’ is the word ‘by’ (Chomsky, 1957). Chomsky also allows for phonemic rules that translate the transformed sentence construction into spoken utterances.

In addition to these basic principles, it is also possible to differentiate between context-free and context-sensitive grammar rules (Chomsky, 1957). In context-free grammars, rules are of the general form a → b, and rule preconditions are matched indiscriminately with any identical object. Rules 2.1, 2.2 and 2.4 above are examples of context-free rules since any case of ‘a’, ‘dog’ or ‘[Noun]’ can be replaced by the rules’ consequents. Context-sensitive rules are of the general form xay → xby. Here, the precondition specifies an object and an environment, which must both exist for the rule to be applicable.

That is, cases of ‘a’ can only be replaced by ‘b’ if the ‘a’ occurs between ‘x’ and ‘y’. An example of a context-sensitive rule from English grammar is “i before e, except after c”, which may be encapsulated in two rules as shown below. Rule 2.6 is a context-free rule that can be used to change all occurrences of ‘ei’ to ‘ie’. Rule 2.7 can be used to correct the behaviour of Rule 2.6 in special cases where ‘ie’ is preceded by a ‘c’.

$ei \rightarrow ie$ Rule 2.6

$cie \rightarrow cei$ Rule 2.7

The grammar framework is developed from and most widely applied to linguistics, in which a grammar is the form, or generalized structure, of a sentence, and a verbal language is described by a vocabulary of words and syntax rules. This was introduced most notably by Chomsky (1965) in his proposition of a “Universal Grammar” that underlies all languages of the world. Chomsky differentiated between the surface and deep structure of a grammar, the former referring to the arrangement of words in a sentence, and the latter, the combination of meanings that the words symbolize.

While the terminology used for the grammar framework is obviously derived from linguistics, further applications of grammars have been suggested. In these contexts, the nature of the vocabulary ‘objects’ may vary widely including, for example, architectural building blocks, rooms, furniture, parts of living organisms such as vertebrates or plants, colours, art and ornament, behaviours or even dreams (Seebom & Wallace, 1998; Stiny & Mitchell, 1978a, 1978b; Flemming, 1981; Knight, 1980; Radinsky, 1987; Prusinckiewicz & Lindenmayer, 1990; Knight, 1989b; Stiny, 1977; Knight, 1989a; Meyer, 1888; Foulkes, 1978). Different verbal languages/dialects vary in their vocabulary and their rules of syntax and can be thought of as ‘linguistic styles’. Grammars of other types of objects define the ‘types’ or ‘styles’ of those objects.

In the development of a landscape grammar, types of landscape objects are inventoried in a vocabulary and their configurations analyzed and generalized as grammar rules. These features describe the type, style, or character of landscape for a region. The grammar thus holds value for the first challenge of VRP – the definition of landscape character. The second challenge of VRP, namely the need for a meaningful mechanism to test such definitions, is addressed by putting grammar structures to use in generating geospatial arrangements of objects within a landscape scene.

2.6 Grammar Mechanisms

The composition of the rules of a given grammar is generally tied to the purpose they serve. Here, there are two functional types of grammars, analytical and generative. Chomsky (1957) suggested

that while grammar rules were traditionally used to assess the syntactic validity of sentences (the analytical function), they can also be used to produce sentences as well (the generative function). Analytical grammars attempt to parse an initial sentence as suggested by the grammar rules. With each rule application, the rule (as defined in the Section 2.5 above) is applied in reverse: the presence of the rule's consequent implies the rule's precondition. If the original sentence is able to be completely deconstructed then it is considered to be compliant with the grammar and thus part of the language to which the grammar pertains. Conversely, generative grammars iteratively combine words together, effectively building a sentence, according to the grammar rules. The complete sentence is the end-product rather than the initial object. Thus, in terms of rules, the two functions operate in opposite manners. An analytical landscape grammar would deconstruct a landscape to assess whether it conforms to the character represented by the grammar. The objectives of this dissertation, specifically the grammatical construction of simulated landscapes, relate more directly to generative grammars, which are now discussed in more detail.

2.6.1 Generative Grammars

Generative grammars produce patterns. By iteratively applying grammatical rules to an arrangement of objects, a construction is incrementally created. Hence, the implementation of a generative grammar is referred to as a *production system*. The incremental or iterative process (Figure 2.1) is sometimes referred to as the *grammar interpretation process*, performed by a *grammar interpreter*. The grammar interpreter is the central mechanism for processing grammar rules and objects. Its main functions are to match objects iteratively in the construction with the preconditions of rules, and to then modify the construction according to the consequent(s) of one or more rules. Thus, the interpreter takes as its inputs an initial construction and a grammar (vocabulary and rules), and outputs a modified version of the construction.

The generative grammar interpretation process shown in Figure 2.1 starts with an initial construction of one or more objects. The generative grammar is comprised of the vocabulary of object definitions and the set of rules that state the syntactical relationships between them. The set of rules is searched to find one or more rules whose precondition (lhs) matches some part of the object construction. Of the matching rules, one or more is applied to the construction, modifying it in some way according to the consequent (rhs) of the rule(s). This modification consists generally of the addition, deletion or alteration of the objects comprising the construction. On the next iteration, this process of comparison and modification is repeated but with the precondition of each grammar rule compared to the now modified construction. If the interpreter finds that there are no rules whose preconditions are applicable to the current construction, then the generative process is forced to halt because it is not possible for more modifications to be made. Alternatively, the precondition of a rule may intentionally

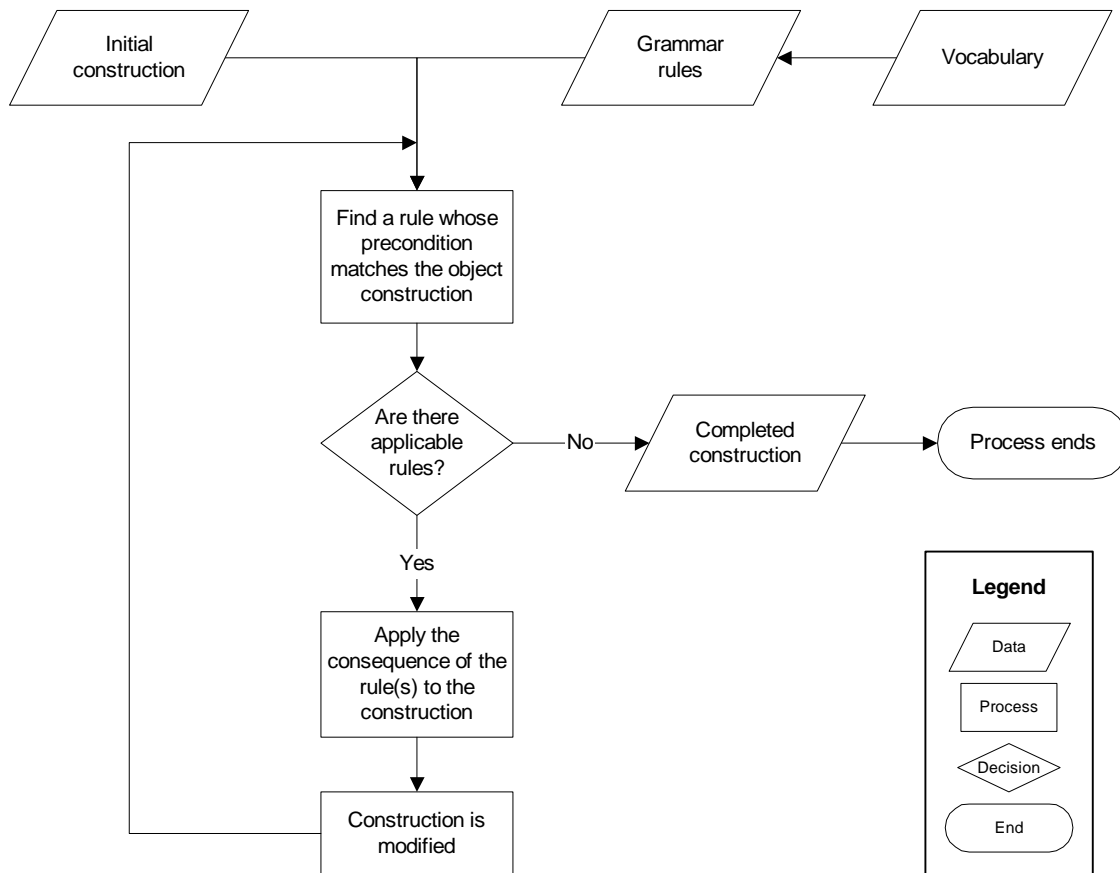


FIGURE 2.1 THE GENERATIVE GRAMMAR INTERPRETATION PROCESS

signal an interruption to the interpretation process. When the generative process ends, a completed construction is the result. Thus, it can be seen that while a grammar is a structure consisting of a vocabulary and a set of rules, it can also be used in conjunction with a grammar interpreter as a generative mechanism to produce sentence constructions.

The following is a simple grammar that may be used to construct limited sentences:

Vocabulary: [Subject], [Article], [Noun], [Predicate], [Verb], [Object],
a, an, the, man, car, tree, idea, stands, drives, has.

[Sentence] → [Subject] [Predicate] Rule 2.8

[Subject] → [Article] [Noun] Rule 2.9

[Predicate] → [Verb] [Object] Rule 2.10

[Object] → [Article] [Noun] Rule 2.11

[Article] → (one of the following : a, an, the) Rule 2.12

[Noun] → (one of the following : man, car, tree, idea) Rule 2.13

[Verb] → (one of the following : stands, drives, has) Rule 2.14

This grammar contains a vocabulary of sixteen objects and six rewriting rules. Rule 2.8 can be read as: if a [Sentence] object exists, then replace it with a construction in the form of [Subject] [Predicate]. By using this grammar as a production system, one of the many possible sentences can be constructed, for example:

Initial Construction:	[Sentence]
Application of Rule 2.8:	[Subject] [Predicate]
Application of Rule 2.9:	[Article] [Noun] [Predicate]
Application of Rule 2.10:	[Article] [Noun] [Verb] [Object]
Application of Rule 2.11:	[Article] [Noun] [Verb] [Article] [Noun]
Application of Rule 2.12:	The [Noun] [Verb] [Article] [Noun]
Application of Rule 2.12:	The [Noun] [Verb] a [Noun]
Application of Rule 2.13:	The man [Verb] a [Noun]
Application of Rule 2.13:	The man [Verb] a car
Application of Rule 2.14:	The man drives a car

No more rules are applicable.

This grammar represents simple knowledge about the structure of sentences. Note that abstract or non-terminal objects (Subject, Predicate, Object, Article, Noun, Verb) as well as terminal words (a, an, the, man, car, stands) are included in the vocabulary. It should also be noted that the use of these grammatical rules can result in syntactically correct, but not necessarily meaningful, sentences. Chomsky's (1957, 15) often-quoted example is the construction "Colourless green ideas sleep furiously". Examples such as this maintain a grammar's surface structure, but lack a meaningful deep structure.

The formal 'language' of the grammar is comprised of all the possible constructions that can be created from the rules. In the above simple example, the constructions, "a man has the car", "the man has an idea", "the idea stands the car", and "the tree drives an idea" among others, are included in the grammar's language. Of these examples, the first two have a meaningful deep structure, while the last two examples do not. The complete language of this example grammar contains 192 syntactically correct

sentences (assuming ‘a’ and ‘an’ are used correctly by Rule 2.12), of which only twenty have semantic meaning.

The above example is a simple phrase-structure grammar, consisting of rewriting rules, in which the lhs is replaced by the rhs in the construction. The model allows for the conceptual structure of a sentence to control the final form. Chomsky (1957) made the further observation that in order to produce meaningful sentences, certain rules need to be applied before others highlighting the need for the ordering of rules into a structural sequence. From this observation, he went on to demonstrate the limits of phrase-structure grammars in constructing anything but simple sentences and proposed a reformulation of the model called the “generative transformational grammar” in which phrase-structure rules lay out the core structure of a sentence and transformational rules convert one sentence structure into another (as in the conversion of a statement from an active form to a passive form – see Rule 2.5). The inclusion of transformations allows a much larger range of possible constructions, or language. The current state of the theory is considerably more complex than can be explained here and has been revised many times by Chomsky (referred to as Government and Binding Theory in Chomsky, (1981) and the Minimalist Program in Chomsky (1995)). However, the early generative grammar theory is sufficient to introduce the concept of a landscape grammar in this dissertation.

The generative grammar concept and mechanism are applicable to the dissertation objective for establishing and applying a grammar to landscape character. A generative landscape grammar would produce landscape scenes from a vocabulary and spatial syntax rules that are defined as the landscape evolves either gradually or through radical redevelopment. The interpretation process involves the iterative population of a scene with landscape objects, which are defined in the vocabulary. The process can include the use of abstract non-terminal objects to influence the structure of the scene. For example, the construction of new roads with vegetation buffers on vacant land can be created iteratively. The types of roads and species of plants found in the area can be defined in a landscape vocabulary. A network of roads can be generated on the site according to rules for finding acceptable paths (which may use abstract objects such as gradient zones) through the landscapes. Subsequently, roadside planting can be added to the scene according to rules that specify appropriate planting schemes.

The gamut of scenes that can be created in this manner comprise the ‘landscape language’ for the grammar as the scene has been generated using objects and their placements derived entirely of the grammatical rules used. If the grammar definition adequately represents the landscape character, then deterministically the generated scenes should each exhibit that character. In this way, the scenes can be used to assess the adequacy of the grammatical definitions. This relates to the second problem of VRP, namely the need to assess whether the landscape character definition can produce desirable landscape scenes.

Obviously, a production system using a landscape grammar does not use a vocabulary of words or rules that add words to form a sentence. The vocabulary describes physical objects and/or spatial concepts. The rules specify spatial or locational, rather than verbal, order. Thus, a definition of landscape character and a landscape production system must be built using spatial constructs. Spatial grammars provide a logical construct for implementing the landscape grammar concept as a conceptual framework and a production system and are the subject of Section 2.7.

2.6.2 Analytical Grammars

While generative grammars provide constructions, analytical grammars provide deconstructions. The purpose of an analytical grammar is to analyze a pattern of objects to see if it can be derived from a particular grammar's vocabulary and rules; that is, whether the pattern is within the language of the grammar. Analytical grammars may be thought of as applying rules in reverse to an already completed construction:

$$[\text{precondition}] \leftarrow [\text{consequent}].$$

Hence, if a pattern exists in a grammar, as the consequent of a rule, then the precondition of that rule is implied. This sequence is described in Figure 2.2. The process is similar to the generative interpretation process of Figure 2.1, except that the analytical grammar starts with a completed construction and iteratively analyzes its parts using the grammar rules. The matching operation searches for parts of the construction that match the consequent of each rule (instead of the precondition as in the generative process). Of the matching rules, one or more are applied to the construction by replacing the instance of the rule's consequent with an instance of the rule's precondition. This process of comparison and modification continues iteratively, reducing the construction, until it is halted by a lack of applicable rules. If the construction conforms to the grammar, then the end-result is some original state of the construction. In the simple example given above for generative grammars, the sentence construction "The man drives a car" would be reduced to "[Sentence]" by applying Rules 2.8 to 2.14 in reverse.

Several analytical grammars are usually tested to determine whether they can derive the initial object. When a set of possible grammars is identified, the objective of the search is to find the 'best' grammar (usually, the one that uses the smallest set of elements and rules). This is sometimes known as pattern recognition – if an object can be more efficiently represented as a set of primitives and rules, then it contains an identifiable pattern in its structure; otherwise, there is no advantage to representing the object in terms of a grammar, and the object's structure is considered random (Stefik, 1995). Analytical grammars are much more difficult to build than generative grammars, mainly because there are usually several ways in which a complex construction may be deconstructed (Gips, 1975). Thus, there may be

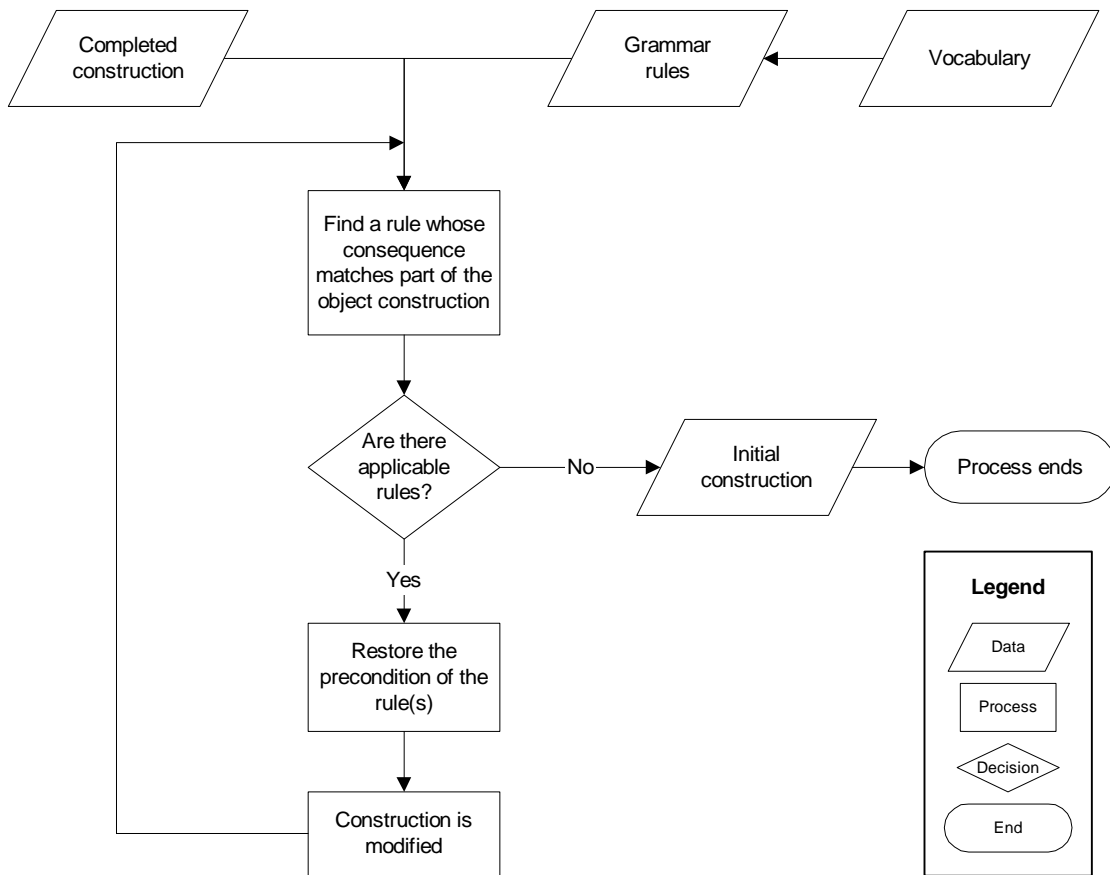


FIGURE 2.2 THE ANALYTICAL GRAMMAR INTERPRETATION PROCESS

several grammars from which a pattern can be derived.

An analytical landscape grammar can be used to deconstruct a scene in order to assess whether the scene could be produced from a given vocabulary and rules. It also determines, by implication, whether the scene is recognizable as an example of the landscape character defined by the grammar. Theoretically, this is a useful mechanism for VRM, in order to assess the appropriateness of a proposed form of land use relative to a defined landscape character (such as high-rise residential construction along a beachfront, for example). The observation that a given pattern may not be unique to one grammar suggests that a landscape scene could be derived from different landscape grammars. The different grammars might relate to different theories about the structure of landscapes. Since the above constraints make the development of an analytical grammar unwieldy, it is not considered further in this dissertation. Rather, in accordance with the identified challenges for VRP, namely to define a landscape character and then examine the effectiveness of that definition, the focus of this dissertation is the use of generative spatial grammars to construct landscapes of a desired character.

2.7 Spatial Grammars

Objects and rules in landscape grammars need to account for spatial dimensions, since landscapes exist in geographic space. The development of spatial grammars originated as a mathematical application, in which the grammar concept was adopted to define geometric patterns formed by the definition of a “shape grammar” (Gips, 1975; Stiny, 1975, 1980a). In its simplest form, a shape grammar contains a vocabulary of primitive geometric shapes and rules that specify how these shapes may be arranged in relation to each other. The processing of such rules results in a geometric two- or three-dimensional construction or pattern. In this context, a shape is often defined as an area or volume enclosed by a set of vertices. Krishnamurti and Stouffs (1993) have generalized the shape grammar concept beyond shape primitives to “spatial grammars”, which includes other entities such as strings, sets, and network graphs. It is argued here that rule-generated patterns in grids of cells can also be considered in the domain of spatial grammars. Although much of the research literature refers to this general area of investigation as “shape grammars”, the term “spatial grammars” is favoured in this dissertation as a more appropriate term for the application of grammar concepts to spatial dimensions, as it denotes more types of spatial entities than shapes alone.

In the spatial grammar formalism, a vocabulary is usually composed of zero- (0D), one- (1D), two- (2D) or three-dimensional (3D) figures. Examples of these vocabulary elements are provided later. Often, the spatial vocabulary is divided into a terminal vocabulary, containing objects which may not be changed and a non-terminal vocabulary, containing intermediate elements which may be replaced by terminal or other non-terminal objects. The non-terminal spatial vocabulary, like that of linguistic grammars, provides a facility with which to influence the structure of the pattern. A set of spatial rules specifies how the elements of the terminal and non-terminal vocabularies may be transformed, or arranged relative to each other. Spatial rules have the same form as the general grammar structures mentioned above, that is

$$[\text{precondition}] \rightarrow [\text{consequent}].$$

In this case, however, the precondition and consequent are spatial configurations. In the processing of a spatial rule, an instance of the precondition’s spatial configuration is replaced with an instance of the consequent’s spatial configuration.

2.7.1 *Types of Spatial Grammar Structures*

Krishnamurti and Stouffs (1993) identified the range of spatial grammars as including string grammars, set grammars, graph grammars, and shape grammars. String grammars refer to phrase-structure grammars, in which characters within text strings are rewritten with new characters. Thus, the

grammar does not operate directly on spatial objects, but rather on strings of text symbols in which each symbol represents a geometric shape. The generated text string has a geometric interpretation which may be constructed after grammar processing is complete.

Figure 2.3 shows a string grammar rule in which the text symbol ‘A’ is replaced with the symbols ‘I[F][–F]A’. The spatial interpretation of the string rule is given above it. In the rule’s precondition and consequent, ‘A’ represents a vertical arrow while in the consequent, ‘I’ represents a line and ‘F’ a line with a round end. The square braces represent a 45° change of direction, which is either clockwise or counterclockwise depending on the presence of a minus symbol (–). Lindenmayer-systems (or L-systems) use this formalism extensively to describe the structure and growth of living organisms (Lindenmayer, 1968). L-systems were originally developed, and have since been predominantly used, for the description of botanical structures from algae to flowering plants to trees (Morelli et al., 1991; Prusinkiewicz & Lindenmayer, 1990). In L-systems, each element of a string, such as “I[L]I[–L]IF”, represents a plant part which has a visual equivalent (I = internode, L = leaf, F = flower). The string can be used to generate a botanic pattern by replacing each text element with its corresponding geometric object (Prusinkiewicz et al., 1997). The rules of string grammars are often context-free, although context-sensitive rules are implemented easily (Prusinkiewicz & Lindenmayer, 1990).

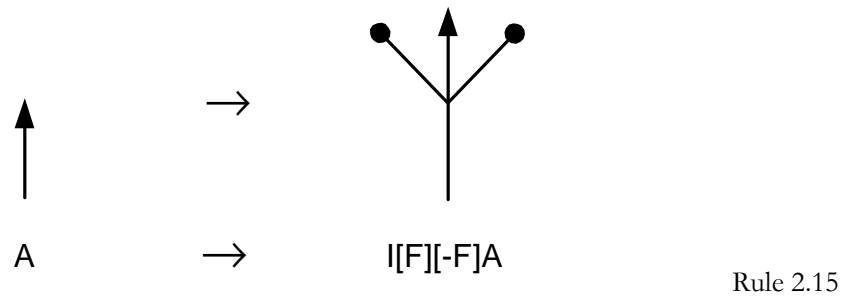


FIGURE 2.3 A STRING GRAMMAR RULE FROM AN L-SYSTEM

Set grammars operate on sets of objects. Structure grammars are a type of set grammar in which the spatial objects are represented as a set of pairs, such as $\{ (a, b), \dots \}$ in which a is a spatial symbol and b is a standard spatial transformation, such as scaling or rotation (Carlson et al., 1991). Unlike linear string grammars, structure grammars are unordered in that there is no significance placed on where an element occurs within the set. The use of rules to add or remove elements to the set corresponds to the set theory operations of union and difference (Krishnamurti & Stouffs, 1993). Solid grammars are a restricted type of set grammar, in which solids are represented as a set of triples with each triple composed of a face (f), a set of edges (E) and a set of vertices (V), or $\{ (f, E, V), \dots \}$ (Krishnamurti & Stouffs, 1993; Heisserman, 1991). The solids are restricted by constraints that ensure the

representation is a ‘true’ solid (for example, no open faces). Heisserman (1991) used the concept of “boundary solid grammars” to construct a grammar for 3D simulations of Queen Anne-style houses.

Graph grammars operate on graphs, or adjacency networks, in which there is no coordinate system, but only nodes connected by branches. These are especially useful for problems concerned with connectivity and flows along a network. Figure 2.4a shows an example graph grammar with a simple vocabulary and two rules. The vocabulary consists of nodes and two types of branches, terminal and non-terminal. Rule 2.16 states that a non-terminal branch may be replaced with a terminal branch, and Rule 2.17 states that a non-terminal branch may be replaced by a graph of four nodes containing one non-terminal branch. In Figure 2.4b, an initial graph structure is shown to which the two rules are randomly applied creating the second graph structure. This example illustrates how a graph structure can be constructed grammatically. Methods in the field of study known as ‘spatial syntax’ simplify urban geometry into axial maps and topological graphs that represent the connectedness of and social and functional relationships between elements in urban space (Hiller & Hanson, 1984; Hiller, 1996, 1999, 2001). Spatial syntax methods are designed for the analysis and discovery of syntax in the landscape rather than the generation of urban designs from a user-defined grammar. Graph grammars are also put to use in the recognition of 3D shapes where the nodes may for instance represent faces of a cube and the branches represent which faces share a common edge.

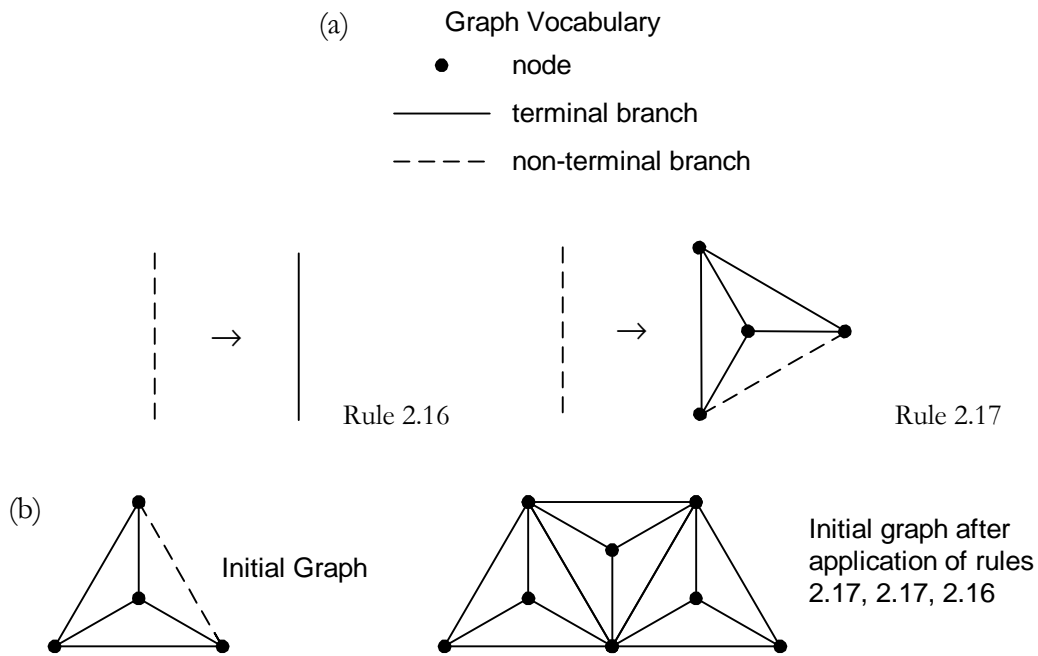


FIGURE 2.4 A GRAPH GRAMMAR AND EXAMPLE APPLICATION

SOURCE: KRISHNAMURTI & STOUFFS (1993)

Shape grammars operate directly on geometric forms comprised of points, lines, planes, and volumes (Krishnamurti & Stouffs, 1993). A shape vocabulary consists of a set of shapes and a set of symbols, which provide the elements with which to define shape rules Stiny (1980a). Stiny (1980a) defines a shape as a limited arrangement of straight lines, and allows for the use of labelling points with shapes. A labeling point is a point and an associated symbol. Consequently, a labelled shape is comprised of a shape and a set of labelled points (Stiny 1980a). In terms of the rule representation presented in previous sections, a shape rule contains a labelled shape as the precondition and another labelled shape as the consequent.

A shape rule applies to a shape construction, if the precondition shape exists as a subshape of the construction. Thus, determining the applicability of a shape rule is a search for subshapes within a working design. Figure 2.5a shows two shape rules. In Rule 2.18, a square shape with a label point is replaced with an inscribed and rotated labelled square and the first label is removed. Rule 2.19 removes the label from a labelled square. Figure 2.5b shows the successive application of the two rules, as indicated by the numbered arrows, to an initial labelled square shape (the rightmost point indicates the termination of the grammar interpretation process rather than a label). Some of the possible shape constructions in their completed form are presented in Figure 2.5c. The labelled shape serves as a non-terminal vocabulary element, as it is later replaced by a terminal shape via Rule 2.19. Similar to string grammars, the rules of shape grammars are generally context-free.

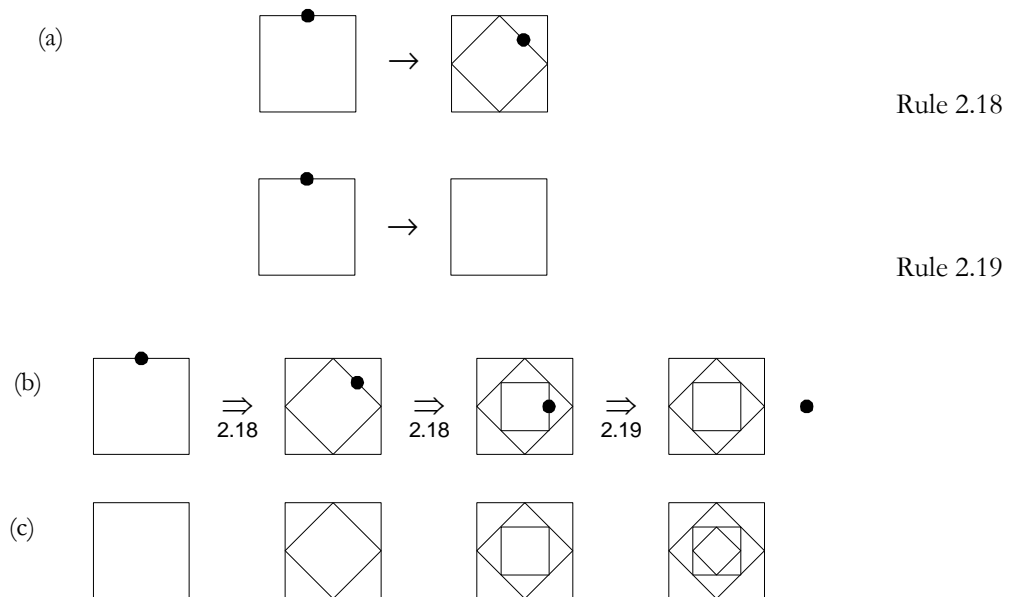


FIGURE 2.5 SHAPE GRAMMAR RULES AND EXAMPLE APPLICATION

SOURCE: STINY (1980A)

Spatial grammars using shapes are significant because they have received the most attention in the spatial grammar literature (for example, generally, Schmitt, 1991; Flemming & Van Wyk, 1993; Gero & Tyugu, 1994; and specifically, Stiny & Mitchell, 1978a, 1978b; Stiny, 1980a, 1980b; Krishnamurti, 1980, 1981; Koning & Eizenberg, 1981; Coyne & Gero, 1985; Krishnamurti & Giraud, 1986; Flemming, 1987; Chase, 1989; McCullough et al., 1990; Mitchell, 1990; Carlson et al., 1991; Stiny, 1993, 1994; Heisserman, 1994; Chiou & Krishnamurti, 1995; Jun & Gero, 1998; Piazzalunga & Fitzhorn, 1998; Knight, 1999a, 1999b; Stiny, 1999; Tapia, 1999). As mentioned at the start of Section 2.7, much of the research literature refers to “shape grammars” rather than “spatial grammars”, although the latter is favoured here. Applications of the shape grammar concept are found in architectural design in order to explore the range of designs that may be generated from a set of design rules (Mitchell, 1990). Architectural grammars are presented more fully in the next section.

In considering types of spatial grammars, it is also useful to include cellular automata (CA) as they entail a class of rule-based systems that operate in space (Wolfram, 1994). While they are not discussed within the literature on spatial/shape grammars, they have an essentially similar structure. CA operate on cells in a grid, rather than on discrete geometric objects, and are commonly used in geographic applications to model the spread or stochastic distribution of a variable over a surface. The grid structure used by CA is different from other spatial grammar applications in that the overall shape of the construction is constrained by a fixed cell size and overall grid dimensions. Furthermore, the only spatial vocabulary element is the grid cell. However, each cell can be associated with a range of symbols or values. In a CA rule, the precondition is expressed in terms of truth statements about cell values, and the consequent states the value that should be assigned to a cell if the precondition is true. The values assigned to cells serve a similar function as the label symbols in shape grammars, that is, the differentiation of identical shapes. Hence, using shape grammar terminology, the CA vocabulary can be seen as a set of labelled squares (assuming a grid of square cells) of a fixed size but with various label symbols.

Landscape grammars apply spatial grammar concepts to geographic space. The data models used in these various types of spatial grammars are already in use in landscape applications, albeit without a grammatical inference mechanism. String grammars have been used in the literature to represent botanical structures, such as Lindenmayer systems, in both two and three dimensions. The construction of botanical objects for simulated landscapes can take advantage of this research. Set grammars can be related to geographical information systems (GIS) in which set theory and restricted structures are a fundamental part, allowing topological relationships between entities. Set theory forms the basis of overlay operations in GIS, such as finding the intersection between two spatially coincident polygons. Vector-based GIS data, like Heisserman’s boundary representations of solids, are organized in restricted

sets, in which a line is a set of points and a polygon a set of lines. GIS data structures offer additional restrictions that a line must start and end with a node, contiguous lines join at a shared node and adjacent polygons join at a shared line. These vector-based data structures, similar to the boundary solid representations but in zero, one and two dimensions, are used widely to represent landscape phenomena such as trees, rivers, or land parcels.

Graphs are used in geographical representations to model connectivity in transportation or stream networks among other things. Thus, graph grammars can be put to geographical purposes by systematically adding and removing branches or nodes within a road or stream network. Landscape grammars can utilize graph structures to represent theoretical or actual linkages between objects in a landscape. The connectivity of landscape objects can subsequently influence their location in space and therefore the content of grammar rules that define spatial arrangement. The use of shapes for landscape representation has been applied to urban and architectural design. Each 2D or 3D shape represents one part of a spatial configuration, such as a floor plan, or a built structure. Collections of shapes together comprise the complete building construction. While cellular data structures, which are also common in GIS, can be used to imply the presence of physical objects, such as buildings, through the pattern of values in the cells (Figure 2.6a), they are more suitable for representing the distribution of spatial characteristics, such as ground elevation, ground slope, soil type and soil depth (Figure 2.6b).

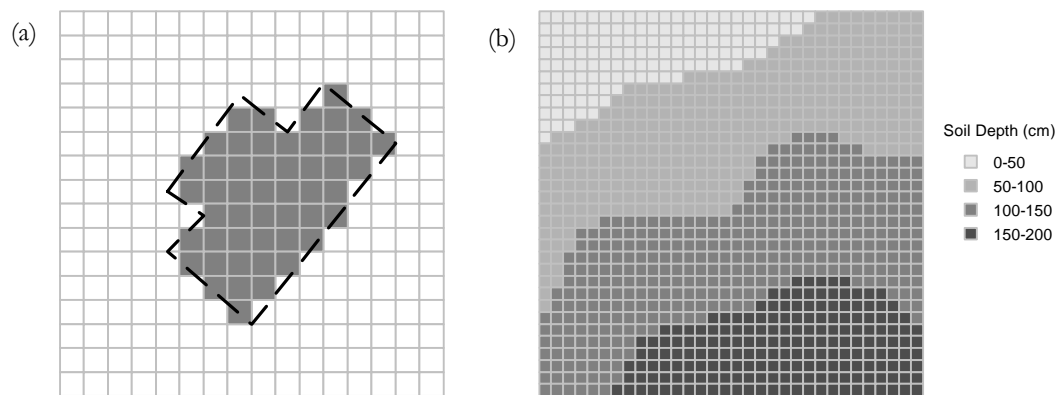


FIGURE 2.6 USE OF CELLS TO REPRESENT DISCRETE OBJECTS AND CONTINUOUS SPATIAL VARIABLES

The data models used in string, set, graph and shape grammars and cellular automata are therefore amenable to landscape applications, suggesting their applicability in the design of landscape grammars. However, the uses of these different data models in conventional landscape applications also suggest that there may be no one appropriate data model for a landscape grammar system. Continuous spatial variables, such as elevation, are best represented as a cell-based surface, while spatially delimited, static objects, such as built structures, may be best represented as geometric shapes. The evolution of

GIS technology over three decades has involved the use of both cellular and geometric structures (referred to as raster and vector), without one gaining dominance over the other. This is because each representation is suitable for different purposes. The extension of spatial models to a grammatical environment should not ignore this.

The nature and types of spatial grammars presented in this section suggest that spatial grammar structures are complementary with existing methods of landscape representation. This observation further suggests that the application of spatial grammar concepts to landscape representation is appropriate and, consequently, that it is possible to use existing landscape representation methods in the design of landscape grammars. Having established in this section the nature and types of spatial grammar structures, the following section discusses the operation of spatial grammar mechanisms in generating spatial patterns, just as Section 2.6 presented the operation of linguistic grammars in generating sentences.

2.7.2 Spatial Grammar Mechanisms

The method by which landscape scenes of a desired character may be created from a spatial landscape grammar definition is through the process of interpreting the grammar for a particular site. As landscape grammars are an application of spatial grammars, the processing of spatial grammars must be examined. Like linguistic grammars, the process of spatial grammar interpretation must start with an initial object. In general, a spatial grammar is made up of the initial object(s), a terminal and non-terminal vocabulary and set of spatial rules (Krishnamurti & Stouffs, 1993). The formal spatial 'language' is comprised of all of the possible spatial configurations of terminal objects that can be created with the grammar (Stiny, 1980a; Krishnamurti, 1980; Krishnamurti, 1981; Chase, 1989). As with linguistic grammars, spatial grammars may be used for generative or analytical purposes.

Generative spatial grammar mechanisms create spatial constructions in the same iterative manner as discussed previously (Figure 2.1), but using spatial objects and patterns. A *spatial grammar interpreter* is the mechanism by which rules are iteratively applied to the initial object(s), thereby constructing a working design. On each iteration, the interpreter mechanism attempts to find parts of the working design that match the preconditions of the spatial rules, and then, if successful, modifies the working design according to the consequent(s) of the selected rule(s). With each iteration, a working spatial construction, or design, is incrementally created. The process continues until no rules are applicable to the working design. Analytical spatial grammar mechanisms attempt to ascertain whether a pattern can be generated from a particular grammar definition. This application is closely related to pattern analysis and recognition, which focus on the problem of finding the minimum number of elements required to represent a meaningful pattern. In grammatical terms, the spatial pattern analysis problem is an effort to

create the smallest possible grammar that can generate the required pattern.

While the processing of generative spatial grammars is similar to the general interpretation process, some peculiarities arise when reasoning about spatial objects. For example, the first stage, in which the precondition of a rule is matched with elements in the working design, is not always a simple recognition of an object. It is often the case that a shape in the working design only matches a rule's precondition after one or more transformations, such as translation, rotation, and scaling. Figure 2.7 illustrates the application of a rule in which an object must be rotated 90° in order to match strictly the rule's precondition. While not typical of shape grammar implementations, other transformations, such as projections and reflections, could be incorporated in a grammar interpreter. By allowing spatial transformations, a landscape grammar interpreter is able to recognize types of objects or configurations of objects regardless of their position, orientation or size.

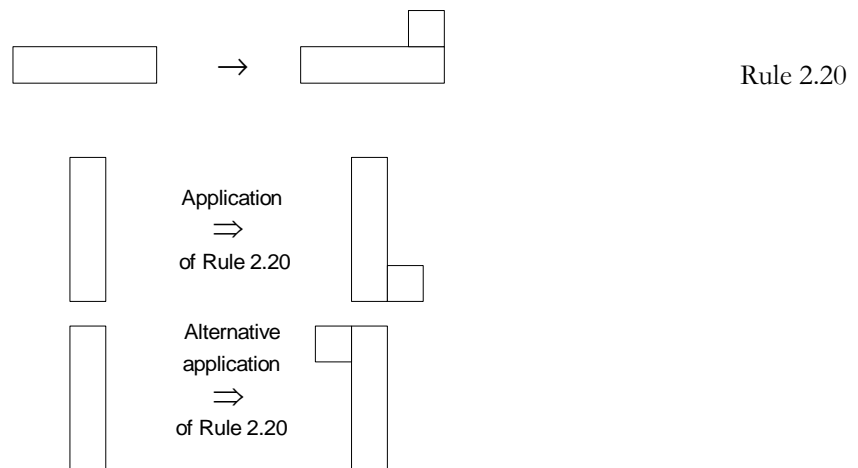


FIGURE 2.7 TRANSFORMATION IN MATCHING SPATIAL GRAMMAR RULES

The application of spatial rules may be controlled by making them context-sensitive. Context-free and context-sensitive spatial rules are illustrated in Figure 2.8. Context-free spatial rules allow the interpreter to match a rule's precondition freely with part of the working design regardless of the other objects in the design. In Figure 2.8a, Rule 2.21 states that a white square can be converted into a grey square, and its precondition does not include the presence of any other objects. Consequently, the iterative application of Rule 2.21 to the arrangement of white circles and squares results in a comprehensive colouring of squares. In contrast, Rule 2.22 of Figure 2.8b is context-sensitive in that its precondition requires the presence of surrounding objects (two white circles) before the colour of the square is modified. The application of the rule is more selective than that of Rule 2.21.

CA rules are almost always context-sensitive, in that a precondition tests the cell values of the

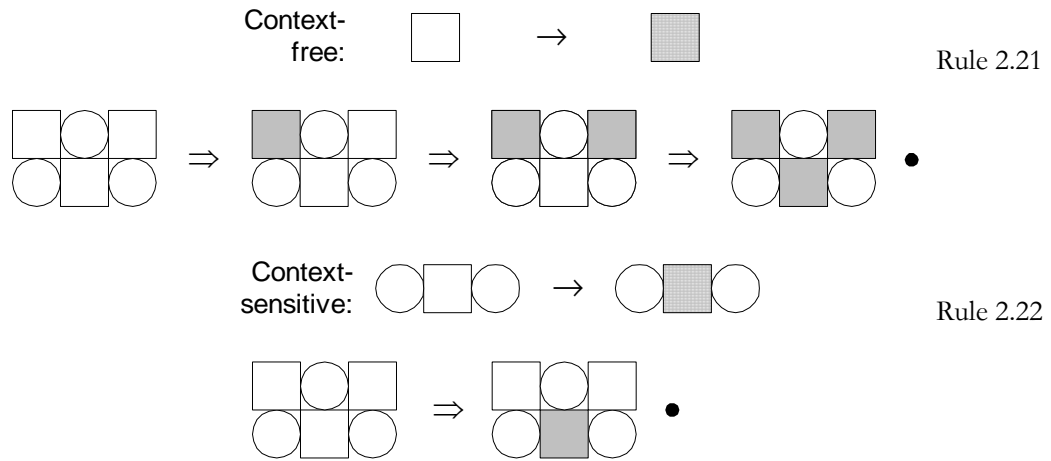


FIGURE 2.8 CONTEXT-FREE AND CONTEXT-SENSITIVE SPATIAL GRAMMAR RULES

‘neighbourhood’ surrounding a particular cell. Such neighbourhoods can be defined in various ways, and two simple examples are shown in Figure 2.9. The Moore neighbourhood of the centre (grey) cell consists of all eight surrounding (dotted) cells, while the Von Neumann neighbourhood is comprised of just four of the adjacent cells. It is through the use of neighbourhoods that spatial interaction models can be implemented. Landscape grammar rules are highly context-sensitive, whether the context involves physical landscape objects or localized values of a continuous spatial variable.

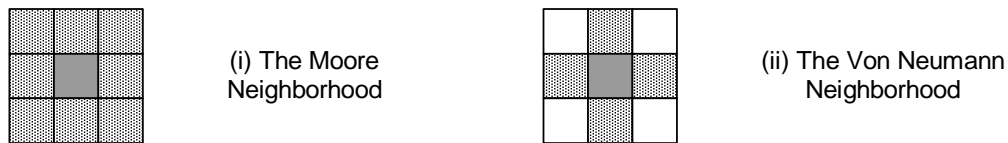


FIGURE 2.9 CELL NEIGHBOURHOODS IN CELLULAR AUTOMATA

Spatial rule preconditions may be further restricted by the use of label points (Stiny, 1980a, 1980b). Labels can provide reference points on shapes, and can be used to define orientation. Figure 2.10 shows a modification of Rule 2.20 from Figure 2.7, using a label point attached to the precondition shape. In this case, the square object may only be applied to the corner of the rectangle object that contains a label point. The alternate application of Rule 2.20 shown in Figure 2.7 is thus not valid here as the label restricts the number of ways in which the rule may be applied. Similarly, Knight (1989b) introduced the use of colors in shape grammars (“colour grammars”). By specifying colours in the consequents, generated designs may possess different colorings. By using colours in preconditions, they also act as labels by specifying further conditions under which rules may be applied (Knight, 1991). Labels are important for spatial landscape grammars for two reasons. First, they can be used to define an

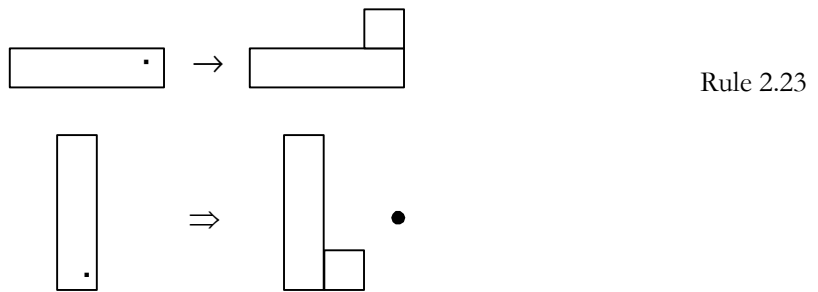


FIGURE 2.10 LABELLING IN SPATIAL GRAMMAR RULES

orientation for a landscape object, the ‘front’ of a building for example. Secondly, the symbols for a label can signify non-spatial information about an object (such as building use) that may affect the location of nearby objects.

A central and current problem for analytical shape grammars is that of shape emergence (Stiny, 1993). In certain cases, the shape for which the interpreter is searching, may be hidden as a subshape in the working design and thus may not be recognizable by the interpreter. In Figure 2.11, for example, the division of the square into quarter-triangles by Rule 2.24, precludes the recognition of half-triangles and the application of Rule 2.25. This problem also exists for a generative spatial grammar interpretation, since at each iteration the working design must be analyzed to see if a rule’s precondition is true. The

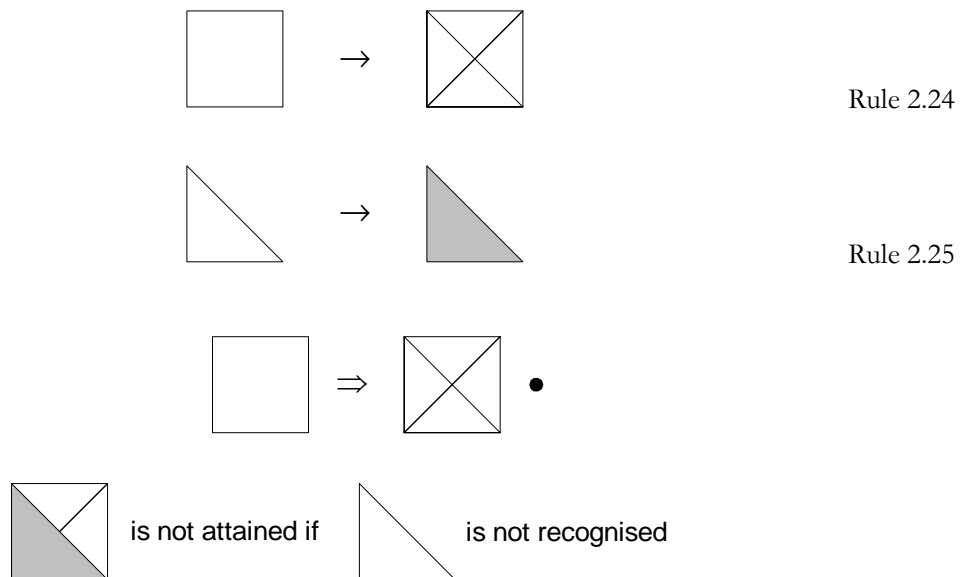
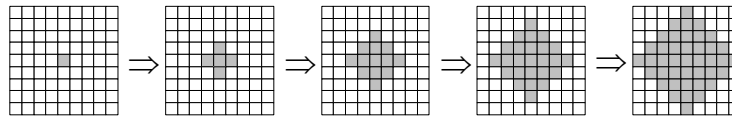


FIGURE 2.11 THE PROBLEM OF SHAPE EMERGENCE IN SPATIAL GRAMMARS

implications of this for generative landscape grammars is that the grammar interpreter, unlike the human eye, may not be able to recognize that a group of objects together imply a more abstract shape, such as a group of hedges enclosing a space to form an outside ‘room’.

In the process of grammar interpretation, the issues discussed thusfar occur during the search for objects in the working design that match the preconditions of each spatial rule. The next stage of interpretation involves the further selection of particular matching objects and rules and the application of rule consequent(s) to the working-design (Figure 2.1). When the consequent of a rule is effected in the working design, the rule is said to be *fired*. For any given rule, there may be several objects in the working design to which the rule is applicable, that is, several objects which make the rule’s precondition true. If the rule is fired on all of the matching objects in one iteration of the grammar interpreter, then the grammar interpretation is said to be parallel (Krishnamurti & Stouffs, 1993). Cellular automata are interpreted in parallel since in each iteration a rule is applied to every cell in the grid, thus changing the value of every cell. For example, Figure 2.12a shows a simple model of dispersion via a cellular automaton, that could represent the spread of urbanization in a landscape or spread of disease through a community. Rule 2.26 states that if a cell has one or more cells in its Von Neumann neighbourhood that are ‘occupied’ (grey) then it also becomes occupied. The rule is fired on one cell in the first iteration, five cells in the second iteration, thirteen in the third, and so on. This exhaustive method is allowed by the

If a cell has one or more occupied neighbours, then it becomes occupied. Rule 2.26

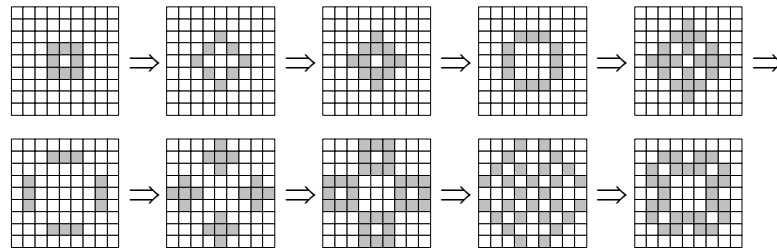


(a) Simple dispersion via one or more cells in the Von Neumann neighbourhood

If a cell is empty and has exactly 3 live neighbours, then it becomes live. Rule 2.27

If a cell is live and has 2 or 3 live neighbours, then it continues to live. Rule 2.28

If a cell is live and < 2 or > 3 live neighbours, then it becomes empty. Rule 2.29



(b) The Game of Life using the Moore neighbourhood

FIGURE 2.12 PROCESSING CELLULAR AUTOMATA

grid structure and independence of cells.

In contrast, if a rule is only fired on one object in the working design per iteration then the grammar is said to be serial (Krishnamurti & Stouffs, 1993). Shape grammars are serial because, given a set of shapes from the working design that match a rule’s precondition, applying the rule’s consequent to one of those shapes may invalidate the rule’s precondition for the other shapes in the matching set, thus making them no longer ‘matching’. To generalize from these observations, it may be said that a spatial grammar rule can be fired in parallel if the objects that are to be modified by the rule are independent of each other with respect to the actions specified in the rule’s consequent.

Not only might there be several different objects in the working design to which a spatial grammar rule is applicable, but conversely there may also be several rules whose preconditions apply to a single object. Because the application of a rule may change the working design, the preconditions of the other rules may no longer match after the first rule has been fired. This necessitates serial grammar interpretation, that is, only one rule is selected and applied to the working design per iteration. It should be possible, however, for rules to be fired in parallel on the same object if it can be assured that the changes to the design caused by the rules are independent of each other. For example, grammar rules that modify different sets of label symbols could be fired in parallel (assuming the label symbols of a shape are not logically related to each other). CA rules are fired in parallel although not on the same cell, as demonstrated in Figure 2.12b. Illustrating the classic ‘Game of Life’ automaton, Rules 2.27, 2.28 and 2.29 are all fired in each iteration for any cells that match the rules’ preconditions. Thus, in the first iteration, each of the three rules are fired on four different cells. In the second iteration, Rule 2.27 is fired on four cells, Rule 2.28 on eight cells, and Rule 2.29 is not fired.

Table 2.1 displays the possible combinations of serial and parallel spatial grammar interpretation. The first column specifies how, in a single iteration of the interpreter, objects that match a rule’s precondition are selected for rule application, while the second column specifies how rules whose preconditions match some part of the working design are selected for firing. The selection of these objects and rules for firing is achieved through selection strategies. In landscape grammars, strategies for selecting object(s) to which a rule will be applied can be based on priorities given to certain types of

Rule Application (selection of objects)	Rule Firing (selection of rules)
serial selection/single object	serial selection/single rule
serial selection/single object	parallel selection/multiple rules
parallel selection/multiple objects	serial selection/single rule
parallel selection/multiple objects	parallel selection/multiple rules

TABLE 2.1 TYPOLOGY OF SPATIAL GRAMMAR INTERPRETATION MECHANISMS

landscape objects, the proximity of objects to a central landscape object or area, the size of an object, or a random selection. If the selection of objects is based on proximity to the objects modified in the previous iteration of the interpreter, then the changes to the working design are localized. Similarly, strategies for selecting landscape rule(s) for firing can be based on the areal extent of the effects of firing a rule's consequent, the type of landscape objects affected by its firing, the type of operations performed by the rule's consequent, or a random selection. Rule selection strategies can be used to influence the order in which a set of rules is applied to a working design, thus imposing a logical sequence of construction. In landscape grammars, such sequencing of rules can be used to represent the typical evolution of a landscape character as exhibited in a particular scene.

At the final stage of an interpretation iteration, some 'cleaning' operations may need to be performed. If some standard spatial transformations were applied to an object in the working design in order to match the precondition, those transformations must be reversed after the consequent of the rule has been applied. This will return the altered object to its original position and orientation.

Generative spatial rules, as with those of linguistic grammars, may take on a number of forms. Like phrase structure grammars, spatial grammars may be straight-forward rewriting rules, in which one spatial object is replaced by another, either sensitive to its context or not. Alternatively, the rules may be transformational, altering the object by some geometric process. Spatial rules may also be parametric, in that their preconditions are not matched literally to the dimensions of an object and their consequents are applied relative to the object's existing geometry (Krishnamurti & Stouffs, 1993). It is also possible to apply rules probabilistically, greatly increasing the range of possible spatial designs.

Generative spatial grammars are a theoretical solution for addressing the issues of VRP, as posed earlier in this chapter. The ability to define landscape character and to test such a definition by producing simulated scenes represents one practical application of the spatial grammar concept. The recurrent physical features that comprise a landscape character can be defined as elements in a vocabulary and the nature of their spatial arrangement captured as spatial grammar rules. Visual resource planners can define their ideas in this manner and subsequently use them to generate landscape scenes to examine these ideas. Thus, the two stated problems for VRP are addressed by the application of generative spatial grammars to real-world landscapes.

2.7.3 Architectural grammars

A particular illustration of the application of generative spatial grammars to real-world phenomena is the development of architectural grammars. Architectural grammars are more regionally limited than landscape grammars, but typically include a greater level of detail. The concept of shape grammars was adopted quickly by architectural researchers, as a framework for studying the styles of

famous architects and of cultural groups. The application is particularly apt if there is a fairly consistent set of building elements and design principles to which the architect or group subscribes.

The vocabulary of an architectural grammar contains representations of each building element used in the creation of an architectural structure. Grammar rules specify how the building elements can be modified and combined with other elements. The formal architectural language contains all of the possible structures that could be built from the vocabulary elements according to the grammar rules. The resulting language is a universal set of designs, from which only a few may actually have been built physically. The styles of Palladian villas (Stiny & Mitchell, 1978a & b; Stiny & Gips, 1978), Frank Lloyd Wright prairie houses (Koning & Eisenberg, 1981), Mughul gardens (Stiny & Mitchell, 1980; Knight, 1990), fire station floor plans (Woodbury & Griffith, 1993), African villages (Herbert, Sanders & Mills, 1994), and Queen Anne homes (Heisserman, 1994) have been described with architectural grammars.

An architectural analytical grammar can be used to test whether a given design can be produced from a set of building elements and construction rules. This approach is useful in correcting generative grammars without searching all of the possible design generations. For example, it can be determined whether the Thomas House is reproducible from a particular Frank Lloyd Wright prairie-house grammar specification (Koning & Eisenberg, 1981). In addition, analytical grammars can be used to evaluate the performance of a design with respect to certain criteria. This has particular applicability with regards to the energy and lighting efficiency of buildings.

With an architectural generative grammar, sets of designs are produced from the application of construction rules to an initial site or structure. This illustrates one of the utilities of the architectural generative grammar – to explore the range of designs that are possible, given a set of objects and building principles (Mitchell, 1990). The architectural researcher can then more fully appreciate the potential implications of using a certain set of design principles. Another main feature of generative architectural grammars is the ability to make modifications to the grammar and to visualize the effects of the modifications on the generated designs. By making additions, deletions or modifications to the vocabulary or the grammar rules, the architect can ask many ‘what-if’ questions of the grammar and visualize the potential results. Hence, the application of grammars to architecture involves an attempt to mimic the design process.

Grammatically generated architectural designs have a visual surface structure or syntax. However, it is also possible to suggest that they possess a semantic meaning similar to that of verbal sentences. Since architectural elements are often considered symbolic or functional, they can be considered to have the additional dimension of a ‘semantic’ meaning. Despite this, grammatical research in architecture has been specifically limited to the surficial syntax of shape designs, omitting the more unwieldy deeper structures. The application of shape grammar research to the field of architecture

demonstrates the applied merits of the grammatical approach. This dissertation extends these ideas into the broader realm of a generalized landscape ‘style’, or character, as articulated in a spatial grammar.

2.8 Implementation of Spatial Grammars

While many of the implementations of shape grammar research have been performed using manual analytic methods, there are good grounds to pursue the development of spatial grammars using computing technology. The storage and processing of rules can be maintained in computer databases and programs and advances in computer graphics technology offer visualization of the resultant constructions in 2D and 3D. The purposes of landscape grammars, as defined here, require particular features in an implementation environment. A facility for defining concepts is important for the construction of the landscape vocabulary. Similarly, there must be a capacity for defining rules, including functions for the manipulation of object geometry and attributes and for the expression of spatial relationships. Because the results are spatial, the need to visualize results is crucial. Therefore, a graphics environment is necessary for any landscape grammar implementation. Finally, in order to combine these elements, there must be a mechanism for processing rules to create and modify the objects in a landscape. This section examines potential issues for the computer-based implementation of landscape grammars. First, aspects of existing computer-based spatial grammar implementations are discussed. Second, existing models for landscape representation in spatial information systems are assessed in terms of their ability to meet the above requirements for landscape grammars. Finally, it is asserted that applications of spatial grammars to the real world are essentially applications of knowledge representation and that relevant techniques may be gleaned from the field of artificial intelligence. Examples from the literature of the application of artificial intelligence to landscape are presented as systems that resemble in some measure landscape grammar systems.

2.8.1 Computer-based Spatial Grammars

The basic structure of a computer-based grammar system includes a rule-base, inference engine or ‘interpreter’, a working-memory, and input/output capacities. The rule-base is a storage facility for a series of IF-THEN statements or their equivalent. The inference engine is comprised of a pattern-matcher, selection strategies, and firing-module for the application of rules. The computer-based inference process is analogous to that described for grammars in general (Figure 2.1). An initial set of objects are placed in the working-memory from which the pattern-matcher attempts to find objects that correspond with the preconditions (IF expressions) of the rules. The matching rules and objects are passed to another module that selects the required number of rules and objects before passing the

selection to the firing-module where the rule consequents (THEN expressions) are fired and the working-memory is modified. When a goal is reached, all rules have been exhausted, or the user elects to quit, the process finishes and the contents of the working-memory are output. The use of computer-based implementations of spatial grammars to create visual designs has been mainly restricted to experimental research rather than practical applications in the workplace. Implementations of the types of spatial grammars identified in Section 2.7.1 are presented here, followed by discussion of some of the new issues that have arisen in attempts to implement spatial grammars.

String grammars have been successfully implemented as L-systems, providing detailed growth models of plants (Prusinkiewicz & Lindenmayer, 1990; Prusinkiewicz et al., 1997). Their implementation is relatively straightforward since the processing involves the replacement of characters in text strings, and the geometric aspects can be isolated in a separate visualization module which merely substitutes characters with geometric objects. Thus, L-systems require less geometric manipulation capabilities than other spatial grammar implementations. Likewise, graph grammars are composed of links and nodes without a specific coordinate system, so they are relatively easy to maintain without demands for computational geometry. Heisserman's (1991, 1994) grammar system implementation that demonstrated the creation of Queen Anne style houses represents solids as a 3D boundary graph.

Shape grammars, as originally devised by Stiny (1975, 1977, 1980a, 1980b), have not been widely implemented because computer-based pattern matching relies on pre-defined elements, while human shape recognition capabilities are much more adaptive. Hence, the problem of shape emergence (discussed in Section 2.7.2) in which new, incomplete or inaccurate shape configurations cannot be recognized becomes more significant in computer-based than in manual shape grammar implementations. Due to these implementation difficulties, effective shape grammar interpreters are neither common nor widely used in practice. Early interpreters included Krishnamurti (1982), Coyne & Gero (1985), Flemming et al. (1985), Krishnamurti & Giraud (1986), and Fawcett (1986) (as cited in Fawcett, 1986).

Heissermann (1991, 1994) and Heisserman & Woodbury (1994) introduced boundary solid grammars, in their computer-based generation of 3D models of Queen Anne style houses. Seebohm & Wallace (1998) have also developed a shape grammar implementation for generating 3D architectural details. Piazzalunga and Fitzhorn (1998) demonstrate an implementation for a 3D shape grammar interpreter. Gips (1999) provides an informal review of shape grammar implementations. Friedell & Schulmann (1990) presented a computational grammar interpreter that constructed simple 3D rectilinear landscapes based on rewriting rules that act on points and areal objects. While this system is the only landscape grammar interpreter described in the literature, it does not recognize the implications for landscape character definition, nor does it develop a theoretical context of landscape grammars in

planning.

Because of their use of a grid structure, computer-based cellular automata (CA) are relatively easy to implement and have been applied to landscape representations. Spatial grid structures are essentially 2D arrays of alphanumeric values which have widespread use in computer programming. Because each iteration of a CA entails the calculation of a new value for every cell in the array, the CA can be programmed as a function rather than a separate knowledge-base and interpreter. However, the latter approach is entirely appropriate though less computationally efficient. Applications of CA involve the modelling of the spatial spread of variables, such as incidence of land uses, vegetative species, or fire propagation (Batty & Longley, 1994; Batty et al., 1997; White & Engelen, 1997; Clark et al., 1994). Artificial life models have also been implemented using CA to model the movement of humans or wildlife (Openshaw & Openshaw, 1997).

The translation of spatial grammars into an executable form has led to the use of additional features that were not part of the original grammatical formalisms. As landscape grammars are an application of spatial grammars, these new features are valuable for the development of computer-based grammars to represent a landscape language. A common concern in grammar implementation is the sequencing of rules in order to manage the complexity of generated designs. Traditional spatial grammars are not structured with explicit rule sequences – the first encountered rule that fits the matching condition is applied. The computer-based grammar interpretation process is heuristically broken down into parts and processed through incremental methods. A hierarchical decomposition is often employed in this context (Carlson, 1993; Woodbury & Griffith, 1993). This can involve a top-down refinement from schematic outlines or massings towards a detailed model (Mitchell, Liggett, & Tan, 1990; Mitchell et al., 1991). It may alternatively involve a bottom-up construction centred on a small, but focal, detail. These incremental methods involve intermediate steps using non-terminal vocabulary that provide vague descriptions of the design without a fixed coordinate geometry (Friedell & Schulmann, 1990; Heisserman, 1991, 1994; Heisserman & Woodbury, 1994). The sequencing of rules can be achieved through the selection strategies implemented in the grammar interpreter mechanism. One strategy is to establish rule priorities that influence the sequence in which rules are fired (Carlson, 1993). Seebohm & Wallace (1998) used classes of rules that determine when they may be applicable, such as a “root rule” which is the first of a group of rules to be fired.

Thematic groupings of rules allow the spatial grammar interpreter to limit its search for appropriate rules to a specific section of the rule-base (Seebohm & Wallace, 1998). They also allow the user to manage large sets of rules more easily, by partitioning his/her knowledge (as expressed in rules) into domain-specific modules. Groups of rules in landscape grammars can apply to vegetation, building construction or road alignment. Note that groups of rules may also be sequenced in a similar manner as

individual rules. Alternatively, the interpreter can be aided by partitioning the space itself into localized areas of rule application. Spatial partitioning allows a set of rules to be applied progressively over an area, as would occur in the phased large-scale development of a landscape. This approach requires spatial indexing of objects or operations on selected sets of objects in order for processing to be efficient.

To alleviate some of the complexity in designing the selection strategies that the interpreter mechanism uses for selecting and sequencing rules for firing, spatial grammar implementations have employed user interaction facilities (Woodbury & Griffith, 1993). In some generative grammar systems, once rules that match objects in the working-design have been identified, the user is asked to select one of these rules or objects, or both, to use in the rule application. User interaction provides an exploration mechanism, through which the user may seek out the more interesting design solutions, rather than waiting for the interpreter to find them. This is especially applicable in design applications where the designer would typically possess control of the overall construction. Constructions in landscape grammars, in contrast, are often the result of numerous stakeholders influencing the form of their properties (or no stakeholders in the case of natural landscape systems) and user-based control is less appropriate.

Another common feature of computer-based grammars is the inclusion of parametric representations which allow variational geometry (Stiny, 1980a; Tapia, 1992). In parametric grammars, the representation of geometric quantities as variables allows a rule's precondition to be applicable to a much larger range of spatial objects than if it were fixed to objects with specific dimensions. Hence, a parametric rule that modifies rectangles will do so for rectangles of any size or location (Carlson et al., 1991). Similarly, when the consequents of a parametric rule are fired, the objects in the working-design are modified relative to their existing geometry or that of other objects (Paoluzzi et al., 1991). Parameters can vary any aspect of a spatial object such as angles and ratios of lengths of lines (Stiny, 1980a). Parametric spatial grammar systems are much more useful in practical applications than non-parametric grammar implementations, because parametric grammars are flexible in their accommodation of objects of varying dimensions. Seebohm and Wallace (1998) allow their parametric construction details to be resized until they meet a restrictive 3D envelope. Parametric L-systems vary from standard string grammars in that each module in the generated character string has an associated parameter that determines, for example, the length of a tree branch that is represented by that module in the string. The selection of an object or dimension to be the independent parametric variable is problematic. Does the number of windows on a wall depend on the wall's length, or does the wall's length depend on the number of windows desired, or other variables? Chains of dependencies can be developed in this way for use in top-down design systems (Mitchell et al., 1991).

Conventional spatial grammars are deterministic in that they have a degree of regularity in the

designs that are output. Even parametric spatial grammars exhibit this regularity if supplied with the same parameters. A much greater variability in grammatical designs can be achieved by introducing randomness and probabilities to the spatial objects and rules of a grammar implementation. Stochastic grammars can introduce probabilities in different ways. Stochasticity may be applied in the selection of rules and objects for firing, in the selection of actions that are performed in a rule's consequent when fired, or in the dimensioning of objects that are produced by a rule's consequent. In this manner, spatial grammar rules may randomly size an object's dimensions within a numeric range, or assign the colour of an object according to a set of colours and associated probabilities. Stochastic L-systems associate a probability that a character will be inserted into a text string. Stochasticity is especially relevant in grammar applications where the user does not directly control the variability of the design, such as the simulation of plant growth in L-systems. This is the case for landscape grammars where the probabilities of types or characteristics of objects occurring in a regional landscape may be incorporated into generative rules in order to simulate landscape complexities.

Unfortunately, the introduction of stochasticity has the undesirable effect of expanding enormously the set of possible design solutions and limiting the ways in which a grammar can be used. Even in non-stochastic grammars, the set of possible grammatical designs can be very large. With stochastic grammars, the set becomes so large that comprehensive exploration and systematic sampling of the language of designs is not feasible. This is a limitation in which grammar implementations deviate from theoretical notions of spatial grammars. However, spatial grammar implementations still prove useful for exploration of design spaces, even though that exploration may not be complete.

The new features of spatial grammars that arise from their implementation as a computer application are informative for the development of a computer-based landscape grammar system. The sequencing of landscape grammar rules can allow structure to be inserted into the iterative process of landscape construction, in the same way that linguistic structural concepts, such as 'subject' and 'predicate', influence the formation of a sentence (Rules 2.8–2.14). The grouping of rules within a grammar permits the user to modularize their knowledge and apply it in a controlled context. User interaction can be employed to direct the grammar interpretation process, although it could work against the discovery of new and unexpected landscapes. Parametric landscape grammars allow objects to be sized and oriented in various ways, while stochastic landscape grammars can mimic the frequency of objects occurring in a region.

While computer-based spatial grammars are usually programmed from scratch, technologies that represent landscapes have been in use for three decades. Landscape professionals such as planners and architects use spatial representation technologies to construct digital versions of landscapes. This activity is not just a technological analogy to 'pen and paper' landscape productions, but also, to return to the

earlier metaphor, a linguistic analogy to the typing of sentences. Any general knowledge of how the words or objects relate to one another is held in the mind of the designer and not explicitly stated in the modelling activity. The computer-based spatial grammars discussed in this section make that knowledge explicit. A landscape grammar implementation must, however, incorporate the methods of these conventional spatial technologies in order to represent its grammatically generated landscape simulations. Thus, in addition to existing spatial grammar implementations, these landscape representation technologies also inform the implementation of a landscape grammar system and are the subject of the next section.

2.8.2 Models of Landscape Representation

There are currently a number of existing technologies and associated data models used to represent landscapes. These include grid-based and vector-based GIS, 2- and 3D CAD systems, and ortho- and oblique photography. Each have useful features for landscape grammars, but none are equipped to define generalized concepts as a vocabulary, or to represent and process rule structures. In essence, each of these technologies records or implies the specific locations of landscape objects, rather than generalized object definitions encompassing a range of possible instances. The latter is required for a landscape vocabulary and its rendition via a grammar into a meaningful landscape scene. Similarly, while conventional landscape representation technologies often contain functions that can modify individual objects, there are typically no mechanisms to represent general rules nor to apply such rules to a particular landscape scene. This section details the strengths and weaknesses of these landscape representation technologies for use in the implementation of a landscape grammar.

The conceptualization of landscape as space is implemented in the grid data model, which represents an area as a uniform cellular grid of fixed dimensions. This model is used in the cellular automata model structure discussed earlier. The most common partitioning scheme is a rectangular tessellation of square cells. This model is used in conventional remote sensing, in which each cell of the image often contains numeric values relating to the energy emitted from the landscape. Orthophotography is a planimetric aerial photograph that is registered to a geographic coordinate system, while oblique photography is taken at an oblique angle to the landscape, as on the ground or from a vantage point. The grid model is also employed in raster GIS to represent landscape as a continuous surface of values for a particular variable, such as elevation, slope, soil depth or soil type. The grid model does not deconstruct a landscape into its constituent objects, but instead orders 2D space into an artificial structure. Objects are only implied by displayed arrangements of cells (Figure 2.6) and the unit to which attribute data may be ascribed and spatial operations applied, is the individual cell. Because of this lack of object definition capabilities, the grid model proves difficult for describing the recurrent entities that

contribute to a landscape character. However, without a grid model component, a landscape grammar cannot refer to continuous spatial data values in its rules. For example, rules to position buildings according to ground slope or plant trees according to soil depth would not be possible.

In comparison to the grid model, an entity-based conceptualization of landscape organizes reality into a collection of discrete entities and groups of entities. The vector data model represents such real-world entities as 2D geometric objects, using point, line and area primitives, as is found in vector GIS environments. This model provides a better facility for describing landscape character because it allows a landscape to be decomposed into its components and new landscapes to be constructed based on the rules of a spatial grammar. As a consequence, non-spatial attribute data, such as age, colour, function, and condition, can then be attached to spatial objects in a vector database. For landscape grammars, this is significant because the attributes can contain categorization information that may allow easier recognition of objects. For example, a rule precondition that attempts to find 'sheds' in a landscape scene can be matched much more efficiently if there are attribute data that identify an object as a shed, rather than attempting indirect methods such as finding buildings with a footprint area of less than four square metres. Attributes can also be used in a similar manner to the symbols of labelled shapes in shape grammars. Because they are not represented visually in the working design, it is easier to store more non-spatial information as attribute data than as label symbols. Attribute data provide a potential wealth of information that can be used for reasoning about objects in the processing of landscape grammar rules.

Vector-based GIS environments typically store the topology of objects, that is, how objects are spatially connected to each other (which lines share an endpoint or which polygons share an edge, for example). The use of topology in a vector data model is advantageous for landscape grammars because it provides convenient data on spatial relations between objects which might otherwise have to be calculated via computational geometry. Topological data therefore allow for more efficient landscape grammar processing.

Some GIS software provides the representation of 3D surfaces through the use of grid-based digital elevation models (DEMs) or vector-based triangulated irregular networks (TINs). Using these models, 2D spatial data can be 'draped' over a 3D surface. This aids somewhat in the visualization of landscape simulations but only if the landscape is viewed from a significant distance since the lack of surface objects in a 'flat' landscape is readily apparent. The vector model of computer-aided design (CAD) technology, while typically limited in its use of attribute and topological data, accommodates 3D solid objects and thus provides more realistic representation of the 3D world than the DEMs and TINs of a GIS. More sophisticated modelling environments allow the association of attribute data to 3D objects.

Though an improvement over the grid data models, object-based GIS and CAD technologies

still hinder the examination of landscape character because they lack facilities for defining generalized rules between types of landscape objects. In order to describe a regional character definition, the articulation of general spatial relations between classes of objects needs to be accommodated. However, both the grid and object data models have value in developing a regional landscape grammar. Each is appropriate to the particular type of phenomenon it represents, whether continuous or discrete, and therefore landscape grammar implementations can make use of both. It is advantageous from a data processing perspective for such implementations to use attribute and topological data. Three-dimensional representations of landscape objects are beneficial for the visualization of a grammar interpreter's generated scenes. Ideally then, a landscape grammar implementation would make use of a full 3D environment, incorporating 3D spatial grids and solid geometric objects, the latter with attribution and topology. Unfortunately, such a data model is not currently available in an implemented form. Incorporating techniques from knowledge representation technologies can compensate the weakness of conventional landscape representation technologies in the definition of grammatical rules.

2.8.3 Models of Knowledge Representation

When grammars are used to model real-world phenomena, grammarians are encoding their knowledge of the phenomena, rather than the phenomena themselves. Thus, grammar implementations can be considered as knowledge representation systems. Landscape grammars, in particular, can be used to encode personal knowledge of the landscape character of a region. The field of artificial intelligence (AI) is relevant in this regard, particularly through its implementation of knowledge-bases. A knowledge-base is a repository of logical structures (often but not always rules) that can be used to reason about situations (Stefik, 1995). In many AI applications, the knowledge-base is applied to a given set of facts, represented as text statements, in order to infer new facts. This inference process is very similar to the grammar interpretation process discussed above. A set of rules is iteratively applied to an initial set of facts, causing a chain of inference leading to a final deduction.

The main application of knowledge-bases in AI has been the development of expert systems, which encode the heuristics of human experts for analyzing a situation and recommending actions. Expert systems have been applied to solving operational and configurational problems for engineering systems, such as heating and ventilation, elevator and computer systems (Jackson, 1999). Medical diagnostic expert systems, such as the seminal MYCIN and its many successors, take a set of symptoms as an initial input and process rules that ask the user for further information and iteratively work towards a diagnosis and treatment prescription (Jackson, 1999). In pattern recognition applications, rules encode heuristic knowledge for identifying objects of interest in a general situation (Russ et al., 1996). Perhaps the most ambitious example of a knowledge-based system is the CYC project which aims to build a

general common-sense knowledge-base that can reason about many different situations from various domains (Lenat, 1995).

The use of knowledge-bases benefits AI because it maintains the independence of knowledge from specific situations. In procedural/algorithmic approaches, useful knowledge is either integrated directly into the programming code or included in the situational data. By generalizing the knowledge and separating it from the control mechanisms, it may be applied to various data sets and be accessible for modification. The use of rules in knowledge-bases is advantageous because they provide a uniform and easily understood method of representation, rule interpreters are relatively straightforward to construct, and the extension and modification of a knowledge-base is essentially straightforward involving the addition or modification of rules without rewriting the programming code (Bench-Capon, 1990). This presumption can prove false with large, complex rule-sets, in which potential conflicts between new and existing rules should be accounted for (Navinchandra, 1993). Another caveat for the use of rule-bases is that the rule interpreter's strategies for controlling the execution of rules must occasionally be fine-tuned (Bench-Capon, 1990).

The structures and mechanisms for rule-based knowledge representation are similar to those of spatial grammars. When the use of spatial grammars moves from abstract shape primitives and patterns as in shape grammars and the cellular Game of Life to simulations of real-world phenomena such as architectural structures and the spread of urbanization, the grammars come to represent a knowledge-base of such phenomena. Representation of the visual character of a region as a landscape grammar can be thus considered an exercise in knowledge representation, since it is the subject's knowledge about the local landscape character that is being encoded. To this extent, the application of knowledge-based techniques to landscapes is informative for and pertinent to the study of landscape grammars. Conventional landscape representation technologies discussed in the previous section, such as GIS and CAD, typically store large amounts of data but do not reason about them. They are 'information-rich' but 'knowledge-poor'. There is a need for a new movement in geographic data management to facilitate not just the recording of geographic objects digitally, but also the modelling of an organization's knowledge of how geographic objects relate to one another (Levinsohn, 2000a, 2000b). Some groundwork for the development of this new functionality lies in the integration of knowledge-based systems with landscape representation technologies, which has usually taken the form of expert systems.

Early developments and many current planning expert systems were capable of only reading and writing text statements about a situation (Kim et al., 1990). More recently, GIS have been more tightly coupled with expert systems, allowing the inference engine to access objects stored in the GIS and perform spatial reasoning on them, as well as displaying the results in map format (Maidment & Evans, 1993; Navinchandra, 1993; Wright & Buehler, 1993). Expert GIS are most typically applied to site

selection principles (Findikaki, 1990) or the allocation of land areas to the most suitable uses (Davis & Grant, 1990; Davis & McDonald, 1993). Other applications include transportation modelling, emergency management planning, environmental dispute mediation, and the evaluation of hazardous waste sites (respectively, Brail, 1990; Southworth, Chin & Cheng, 1990; Lee & Wiggins, 1990; Fang, Mikroudis & Pamukcu, 1993). These applications resemble analytical grammars and diagnostic expert systems more than generative spatial grammars that construct patterns.

In contrast, cartographic generalization is a field of spatial expert systems that involves a generative process of object placement. It aims to automate the production of various types of maps from GIS databases by using heuristics for the symbolization of features and placement of annotation (Brassel & Weibel, 1988; Buttenfield & McMaster, 1991; Muller & Mouwes, 1990; Muller et al., 1995; Elias, 2002; Sester, 2002). Kada (2002) and Forberg & Mayer (2002) also extend such generalization to 3D models. Insofar as a map is a landscape simulation, cartographic generalization systems resemble landscape grammar systems in producing a meaningful landscape scene from generalized rules. However, the locations of landscape objects in cartographic applications are fixed by the records in their GIS data source. Landscape grammars are afforded the further freedom of determining the locations of objects as well as their appearance and labeling.

Expert CAD systems applied to mechanical and electronic engineering and architectural design are also typically generative in nature, and have been quicker to adopt geometric representations than their GIS counterparts. Since many CAD modelling functions are available to programmers in the form of computer graphics libraries, 'expert CAD' systems are often self-contained, custom-designed software rather than a coupling of expert systems and CAD software. In this respect, many of the shape grammar implementations that have already been mentioned in Section 2.8.1 can be considered expert CAD systems, especially Woodbury & Griffith (1993), Heisserman (1994), Piazzalunga & Fitzhorn (1998), Seebohm & Wallace (1998), and Tapia (1999). Three-dimensional implementations either extrude from the 2D models with additional 3D embellishments (Woodbury & Griffith, 1993), or manipulate 3D models directly (Heisserman, 1994; Seebohm & Wallace, 1998).

Spatial expert systems, whether integrated with CAD or GIS, illustrate the combination of AI techniques of knowledge representation with conventional spatial technologies to iteratively analyze planning situations, or to construct maps and models. The knowledge representation techniques, conventional spatial technologies and existing implementation of spatial grammars presented in this section inform the development of landscape grammar systems. Together, these fields of study encompass a considerable body of literature from which concepts, methods and techniques may be drawn. The implementation of spatial grammar systems has highlighted areas in which grammar software must depart from the theoretical notions of spatial grammars. The data models and technologies already

used to represent landscapes are valuable for representing landscape grammar data. The landscape grammar may also be understood as a 'landscape knowledge-base', and thus related to AI techniques, particularly spatial expert systems.

The literature reviewed in this section of the dissertation suggests that the features of a computer-based landscape grammar implementation should include a landscape representation system that uses both grid and vector models. The latter should be augmented by topological relationships, attribute data and a modular knowledge-base of spatial landscape rules which state how landscape objects typically occur in the regional landscape. Moreover, such an implementation must be capable of constructing landscape scenes in a sequential manner with the use of parametric and stochastic elements. The application of such an implementation allows planners' or others' knowledge about the landscape character of a region to be stored and then employed to generate example landscape scenes that are consistent with that knowledge-base.

2.9 Summary

This chapter has established the theoretical and technological foundations for the concept of 'landscape grammars'. Visual resource planning, while not recognized in literature, has identifiable goals and challenges. VRP's main topic of concern, landscape character, can be modelled as the recurrent types of elements within a region. Two main challenges for VRP are (i) the definition of a landscape character and (ii) the testing of the ability of such a definition to produce landscapes of the desired character.

The common metaphor of landscape and language is based on the view of both phenomena as patterns of symbols that are arranged consistently in relation to each other whether in verbal sentences or in real-world space. These patterns also convey meaning to the observers of landscape or language. Regional landscape planners are required to be able to 'read' their local landscape and produce plans that act as grammatical rulebooks stating the basic syntax that designers should use to create landscapes consistent with the desired landscape character. The landscape-language metaphor can be put to more rigorous use via the use of grammar structures. A grammar is comprised of a vocabulary of objects and a set of syntactical rules. Whether applied to linguistic words or spatial landscapes, grammars can be used to analyze object constructions to assess their conformity to a syntax or to generate new object constructions according to the syntax rules. In this way, definitions of landscape character encoded as a landscape grammar can be used subsequently to construct new landscape scenes that embody the stored character definition.

Generative landscape grammars are necessarily spatial in nature and there is a considerable

literature on spatial grammar models incorporating strings, sets, shapes, or grids. In a landscape-related application, shape grammars have been applied to the description of architectural styles and used to generate new architectural constructions of those styles. The existing implementations of spatial grammars in a computational environment suggest some common desirable features including topology, attribution, rule sequencing and grouping, space partitioning, rule prioritization, user interaction, parametric geometry and stochasticity. The established use of geographic technologies suggests that no single data model may be appropriate for representing landscape objects in the grammar vocabulary. Artificial intelligence techniques, specifically those related to knowledge-based approaches, may also be employed in implementing spatial grammars.

The concept of landscape grammars, as presented here, is based on a considerable and diverse body of knowledge, both theoretical and technological. This chapter has provided the foundations from which the concept of landscape grammars arose. A more detailed development of the concept of landscape grammars and its application to VRP is presented in the following chapter.

Chapter 3

Landscape Grammar Theory and Definitions

In the review presented in the previous chapter, several concepts were discussed in relation to linguistic and spatial grammars with assessments of their contribution to landscape grammars. Before a demonstrative application of a landscape grammar can be developed, it is necessary to formalize these concepts in a manner that is amenable to computer-based implementation. This chapter presents the nomenclature for generative spatial landscape grammars, as well as formalisms that define in more precise terms their components and processes, consistent with the discussion presented in Chapter 2.

3.1 Landscape Grammar Concepts

As suggested in Chapter 2, landscape grammars offer a framework for conceptualizing and representing visual landscape character. Landscape character was defined as the recurrent elements of a region's visual identity. Such elements are related to each other forming recognizable patterns that contain experiential meaning and legibility, which evoke human attitudinal and behavioural responses. It was asserted that a regional landscape character is defined by abstract definitions that represent our knowledge of real-world objects and their relationships, rather than precise definitions of the specific locations and attributes of individual objects (just as an English language grammar is defined by the relationships between types of words (Noun, Verb, etc.) rather than every possible word in its vocabulary). The concepts and relationships of a landscape character definition are manifested in a physical landscape by the specific patterns of objects and the relationships between them, as with words in sentences.

Reflecting this definition of a regional landscape character, the landscape grammar model presented in this chapter is composed, first, of a vocabulary (V) that contains the conceptual definitions of types of real-world objects that are in the region. These definitions are manifested as objects that comprise a landscape scene (S) that embodies the character being modelled. The spatial and non-spatial relationships between the vocabulary's concepts, and therefore the scene's objects, are expressed in a set of rules (R) that bring order and meaning to the objects found in the landscape. These three elements (V, S, R) comprise the structure of the landscape grammar model as a vehicle for landscape character definition. To examine the effectiveness of a landscape character defined in this manner, the grammar

must be applied to a particular site. Given an initial scene (IS), the landscape grammar interpretation process iteratively applies the rules to the objects of the scene. At each iteration, the scene is modified until a final landscape scene is reached. Using this process and a single initial scene, many final scenes may be generated, each embodying the character that has been defined in the grammar. The complete set of these scenes comprises the formal language (L) of the grammar for that site. Hence, a landscape grammar (LG) that produces landscape scenes is comprised of these three elements:

$$LG = \{V, R, IS\}. \quad \text{Eqn 3.1}$$

Using set theory notation, the following sections elaborate on these elements of a landscape grammar structure and then the processes involved in interpretation of the grammar. Uppercase letters are generally used to represent a set, while lowercase letters represent single elements of a set. Members of a set that are themselves sets are represented with uppercase letters. Ordered sets are signified using parentheses ($()$), while unordered sets are enclosed with braces ($\{\}$).

3.1.1 Landscape Vocabulary

A landscape *vocabulary* (V) represents knowledge about the types and characteristics of landscape objects in a region. It is a set of defined concepts that describe the types of objects found in a landscape. Each of the elements of a landscape vocabulary is termed an *object-type* (OT). Formally,

$$V = \{OT_1, OT_2, \dots, OT_n\}, \quad \text{Eqn 3.2}$$

where a given combination of object-types comprises the types of landscape objects in a region. Examples of simple landscape object-types include, for example, Tree, House, Garage, Fence, River or Road. The object-type definitions are general templates that serve as the basis for specific instances of objects that are found in a physical landscape. They represent a person's or group's classification scheme for the identifiable components of the landscape. The examples given above are fairly intuitive as representing physical features of a landscape, however an object-type may also describe a spatial but non-physical concept that influences the pattern of physical objects. These abstract object-types may be socio-spatial constructs such as Front-Yard, Building-Axis, Building-Envelope, Access-Route, Catchment-Area, Public-Park, and Private-Amenity-Space. Lynch's (1960) well known typology of landscape elements – Path, Edge, District, Landmark, and Node – can be considered as further examples but for a larger landscape. Using the terminology of spatial grammars from Section 2.7, the abstract object-types are non-terminal, while the physically-based object-types are terminal. Some of these abstract object-types, such as Boundary-Marker or Protective-Barrier, may be represented in a landscape by other physically-based object-types, such as Garden-Wall or Chain-Link-Fence respectively.

For landscape grammars, the object-types in a vocabulary are also hierarchical. The use of a conceptual hierarchy is a general knowledge representation method (Stefik, 1995) that allows object-types to be *subtypes* or *supertypes* of other object-types. The supertype of any landscape object-type can be defined by generalizing the object-type definition to a greater level of abstraction. Similarly, its subtype is defined by refining the definition into more specialized object-types. For example, a hierarchy of types of built structures may be devised as shown in Figure 3.1.

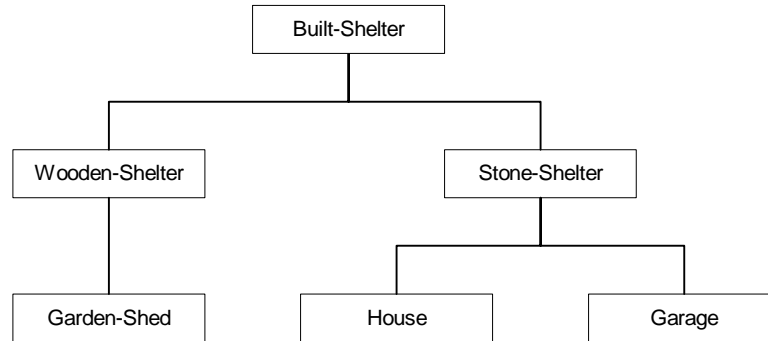


FIGURE 3.1 AN OBJECT-TYPE HIERARCHY FOR BUILT SHELTERS

In this example, Built-Shelter is the most abstract type of object, that is, it is a supertype for the other five object-types. Wooden-Shelter and Stone-Shelter are more specific types of a Built-Shelter and are hence subtypes of the Built-Shelter object-type. Wooden-Shelter is both a subtype of Built-Shelter and a supertype of Garden-Shed. Similar landscape hierarchies are imaginable for types of trees (palms, pines, cedars, citrus, and their subtypes) or ancillary structures such as fences (wooden or chain-link) or swimming pools (wading, diving, swimming, etc.). There is a tendency for object-types higher in the hierarchy to be more abstract or non-terminal, than those in the lower or terminal branches of the hierarchy. An object-type is thus related to other more general or refined object-types in its hierarchy. In formal terms, an object-type's place in the hierarchy can be represented by its set of supertypes (γ_{OT}):

$$\gamma_{OT} = \{OT_1, OT_2, \dots, OT_n\}, \quad \text{Eqn 3.3}$$

where $OT_i \in \mathcal{V}$. In order to symbolize the supertype and subtype relationships between object-type, \mathcal{A} is a supertype of \mathcal{B} shall be denoted as $\mathcal{A} \succ \mathcal{B}$, and conversely, \mathcal{B} is a subtype of \mathcal{A} as $\mathcal{B} \prec \mathcal{A}$.

Since an individual landscape object can be defined in terms of its spatial and non-spatial characteristics, the conceptual type definition for that object in the vocabulary must also have spatial and non-spatial components. The spatial characteristics of an object-type are defined by its *shape-type*. Shape-types define geometric objects that reside in the same coordinate space. A *Point* is a zero-dimensional

(0D) shape-type represented as an ordered pair, (x, y) where x and y are the real world coordinates of the point. A *Line* is a one-dimensional (1D) shape-type represented as an ordered set of points, (p_1, p_2, \dots, p_n) . A *Polygon* is a two-dimensional (2D) shape-type represented as an ordered set of closed points, (p_1, p_2, \dots, p_n) where $p_1 = p_n$. These three shape-types correspond to the three spatial classes typically used in the entity-based landscape representation model (Section 2.8.2). Other shape types are possible. If the Point shape-type is expanded to a third dimension to be the ordered triplet, (x, y, z) , then a *Surface* can be defined as a set of Polygons which are each a set of 3D Points. A *Solid* may be defined as a set of Surfaces, or as a set of 3D Points, such that none of the surfaces are open. The treatment of landscape grammars in this dissertation will only utilize the two-dimensional shape-types for the purposes of demonstration. A pseudo shape-type, *Group*, allows several objects to be treated as one, such as a Forest object-type defined as a group of Tree objects. The constituent objects of the group are restricted to the same shape-type in order that an operation on one object of the group is valid for the others. A set of shape-types for all types of landscape objects (the universe) is defined as:

$$ST_U = \{ Point, Line, Polygon, Surface, Solid | Group(st) \}, \quad \text{Eqn 3.4}$$

where st is a member of the set of proper shape-types in ST_U (left of the vertical bar).

While shape-types define the spatial characteristics of object-types, *attribute definitions* describe their non-spatial characteristics. They effectively serve as variable definitions, such as Age, Species or Colour. While spatial attribute definitions such as Area or Length could conceivably be included, the approach taken here is that spatial attributes can be calculated from a shape's geometry and therefore do not need to be stored explicitly as an attribute of the shape. The universal set of possible attribute definitions for all object-types in the vocabulary is defined as:

$$AD_U = \{ ad_1, \dots, ad_n \}, \quad \text{Eqn 3.5}$$

where ad is the name of a variable. The range of attribute definitions for a landscape vocabulary can be vast, but each object-type has its own subset of attribute definitions ($AD_{OT} \subseteq AD_U$). The attribute definitions for an object-type allow for a much richer description of the objects found in a regional landscape than if labelled shapes alone were used as in traditional shape grammars. As a result of the above discussion, an object-type can be defined as a set containing a set of supertypes (\mathcal{Y}), a shape-type (st) and a set of attribute definitions (AD):

$$OT = \{ \mathcal{Y}, st, AD \}, \quad \text{Eqn 3.6}$$

where $st \in ST_U$ and $AD \subseteq AD_U$.

An advantage of hierarchical representation is that object-subtypes inherit a shape-type and attribute definitions from their supertype(s). Each subtype can possess its own specialized attributes as well, depending on the object-type represented. Thus, while the Garden-Shed object-type may inherit an attribute definition such as Wood-Material from its supertype, Wooden-Shelter, it may also have its own attributes such as Paint-Colour. Formally, we can say that the set of attribute definitions for an object-type is the union of the set of attribute definitions for its supertypes and the combined set of attribute definitions that are peculiar to it. Expressing this relationship formally:

$$OT = \{ \gamma, st, AD_{\gamma} \cup AD \}, \quad \text{Eqn 3.7}$$

where AD_{γ} is the set of attribute definitions belonging to the supertypes of the object-type. In Figure 3.2, for example, the object-types from Figure 3.1 are shown with their attribute definitions immediately below the name of each object-type. The lower five object-types inherit the Roof-Type attribute definition from the Built-Shelter object-type. In addition, the House and Garage object-types inherit the Stone-Type attribute definition from their immediate supertype, Stone-Shelter.

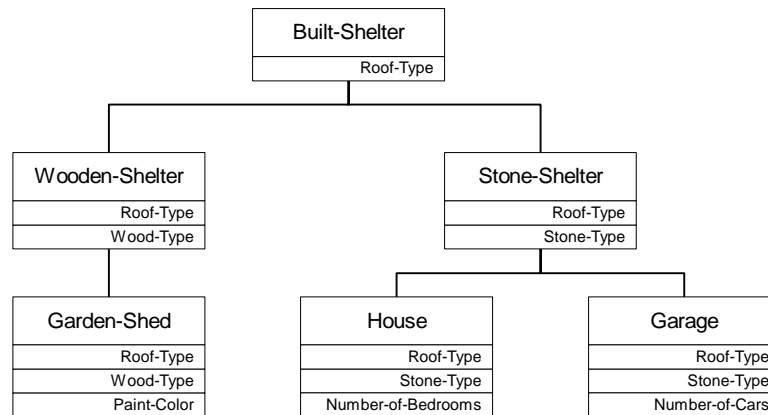


FIGURE 3.2 INHERITANCE OF ATTRIBUTE DEFINITIONS

Using attribute definitions and the object-type hierarchy, the landscape vocabulary for a region can be defined in different ways. The object-type hierarchy in Figure 3.2 could be formed as shown in Figure 3.3. In this case, the hierarchy has been simplified because the difference between wooden and stone shelters is represented by an attribute definition, Built-Material, rather than separate subtypes in the hierarchy that are based on the building material. In this way, attribute definitions can be created from object-subtypes, or conversely, object-subtypes can be created based on attribute definitions. Either representation is acceptable and the decision to use one over another can depend upon how the landscape rules will reason about the object-types during the grammatical construction of landscape

scenes. For instance, the introduction of the new attribute, Built-Material, in Figure 3.3 replaced not only the Wooden-Shelter and Stone-Shelter subtypes but also the Wood-Type and Stone-Type attributes of them and their subtypes. The information previously associated with those attributes (for example, “cedar”, “pine”, “limestone” or “granite”) is now associated with the Built-Material attribute. Using the hierarchy of Figure 3.2, a rule would differentiate a wooden building from a stone building easily: “IF the object is of the type Wooden-Shelter THEN ...”. In the simplified hierarchy of Figure 3.3, this same reasoning would have to involve a string of comparisons: “IF the Built-Material of the object is cedar OR the Built-Material of the object is pine THEN ...”. The development of a landscape character vocabulary thus involves a blurred distinction between the use of subtypes and attribute definitions, the suitability of which is effectively left to the scheme that is most meaningful and useful to the person or group building the grammar.

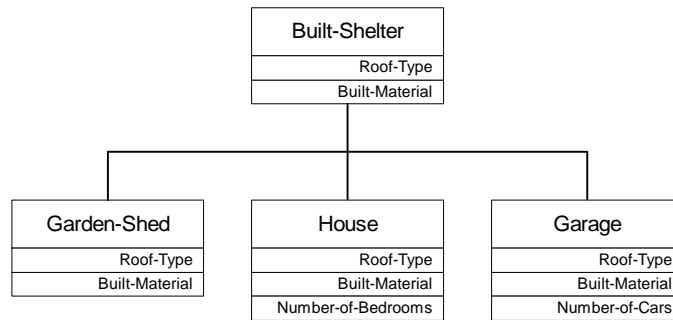


FIGURE 3.3 AN ALTERNATE CLASS HIERARCHY FOR BUILT SHELTERS

It was identified in Section 2.8.2 of the previous chapter that a landscape grammar model should incorporate both grid and object landscape representation models to facilitate continuous spatial variables as well as discrete spatial objects. A *Grid* is incorporated here as a separate ‘shape’-type that is represented as an ordered set of *Cells*, each of which represents a regularly shaped unit of space in the landscape. Formally, the grid is defined as follows:

$$Grid = \{ncol, nrow, cs, (c_1, c_2, \dots, c_n)\}, \quad \text{Eqn 3.8}$$

where *ncol* and *nrow* are respectively the numbers of columns and rows in the grid, *cs* is the cell-size or resolution in real-world coordinates, *c* represents a cell, and its dimensionality is $n = ncol \times nrow$. By themselves, the cells hold no information until attributes are defined. A grid-based ‘object’-type may be defined on a grid of cells with a set of attribute definitions each of which corresponds to one set of cell values. As an example, an object-type Soil may be represented as a grid with attribute definitions for Depth and Soil-Type. Because a Grid represents continuous spatial variables rather than discrete objects,

it does not fit well into the hierarchical relationships of object-types. The set of supertypes for a Grid object-type is therefore null. The explanation for an object-type in Equation 3.6 may therefore be modified for grid-based object-types as follows:

$$OT_{Grid} = \{\emptyset, Grid, AD\}. \quad \text{Eqn 3.9}$$

In summary, the vocabulary of a landscape grammar describes landscape character through the definition of general types of objects found in the regional landscape. These definitions are related to each other through an object hierarchy allowing the characteristics of abstract object-types to be shared with more refined object-types. The vocabulary utilizes object- and grid-based landscape representation models in defining the spatial components of object-types. The definitions are also enriched with types of non-spatial information that are defined as attributes for each object-type.

3.1.2 Landscape Objects and Scenes

The vocabulary's generalized definitions of the types of objects that can be found in a regional landscape are manifested in a simulated landscape scene as specific objects with their own characteristics. These objects represent real-world, physical and non-physical, spatial objects and are instantiations of the landscape concepts defined in vocabulary. A scene is a collection or set of objects, and landscape grammars, as defined in this chapter, therefore resemble set grammars as discussed in Section 2.7.1. The choice of objects and the patterns in which they are arranged ascribes meaning to the scene, just as the choice of words and their grammatical relationships ascribe meaning to a sentence. In a landscape grammar, an initial scene becomes populated with objects through the iterative application of grammatical rules (rules and the interpretation process are discussed in the next sections). The visualization of the resultant scene of objects reveals the landscape character as it is defined by the grammar. In this section, the symbol ψ is used to represent "is an instance of", for example, Toronto ψ City and Lake Ontario ψ Water-Body.

An *object* is defined as an instance of an object-type ($obj\psi OT$). Through the inheritance relationships of the object-type hierarchy, an object is not only an instance of its immediate object-type but also an instance of each of the supertypes of that object-type. Stating this relationship in formal terms, given $obj\psi OT$, and γ_{OT} is the set of supertypes of OT :

$$\forall ot \in \gamma_{OT}, obj\psi ot. \quad \text{Eqn 3.10}$$

Since an object-type has spatial and non-spatial components and a landscape object is an instance of an object-type, the object too must have spatial and non-spatial characteristics. An object has a shape or

geometry (G) that defines precisely its spatial extents and is an instance of the shape-type defined in the object's object-type. An object also has a set of attribute values (A) corresponding to the attribute definitions in the object's object-type. Hence,

$$G = (p_1, p_2, \dots, p_n) \quad \text{Eqn 3.11}$$

$$A = (v_1, v_2, \dots, v_n) \text{ and} \quad \text{Eqn 3.12}$$

$$obj = \{OT, G, A\}, \quad \text{Eqn 3.13}$$

where $G \psi st$ and is comprised of a set of points (p) defining the shape, and A is a set of values (v) that correspond to the attributes defined in the object's object-type. More specifically in relation to A , the number of attribute values associated with an object (the cardinality of A) is the same as the number of attribute definitions for its object-type (the cardinality of $AD_y \cup AD$). In other words, an object has an attribute value for each attribute defined in its object-type. Each attribute value can be considered an instance of the attribute definition, that is, $v_i \psi ad_i$ where $v \in A_{obj}$ and $ad \in AD_{OT}$. As an illustration, a particular object might be an instance of the object-type House from Figure 3.2. The object is comprised of a Polygon shape with a set of points defining its perimeter and a set of attribute values (slate, concrete, 3) that correspond to the attribute definitions for the House object-type (Roof-Type, Stone-Type and Number-of-Bedrooms). Formally,

$$House = \{ \{ BuiltShelter \}, Polygon, \{ RoofType, StoneType, NumberofBedrooms \} \}$$

$$obj = \{ House, (p_1, p_2, \dots, p_n), (slate, concrete, 3) \}$$

where $obj \psi House$,

$$(p_1, p_2, \dots, p_n) \psi Polygon,$$

$$slate \psi RoofType,$$

$$concrete \psi StoneType, \text{ and}$$

$$3 \psi NumberofBedrooms.$$

Each object in a landscape scene can be represented in this way, that is, as a shape geometry and set of attribute values. Because a Grid is fixed in terms of its size and resolution, an instance of a Grid contains no geometry. It does however contain an ordered set of values each of which corresponds to a cell in the grid. In addition, one of these sets of values exists for each attribute defined for the grid. An

instance of a Grid therefore contains multiple ordered sets of attribute values:

$$grid = \{(v_{11}, \dots, v_{1n}), \dots, (v_{m1}, \dots, v_{mn})\}, \quad \text{Eqn 3.14}$$

where n is the number of cells in a grid and m is the number of attributes defined in the grid object-type.

A landscape *scene* (\mathcal{S}) is a set of landscape objects whose types are defined in the vocabulary. This includes both spatial objects and grids of values. The formal definition of a scene is as follows:

$$\mathcal{S} = \{obj_1, obj_2, \dots, obj_n, grid\}, \quad \text{Eqn 3.15}$$

where $\forall obj, obj \psi OT \in \mathcal{V}$. In Section 3.1, a landscape grammar was defined as $LG = \{\mathcal{V}, R, IS\}$ in which an initial scene is provided. In accordance with the above definition of a scene, the initial scene is a collection of objects and grids which will be acted upon by the consequents of landscape rules. For spatial grammars defined in the research literature, initial objects must be defined in the vocabulary. The application of spatial grammars to landscape brings the additional requirement that, at a minimum, the initial scene must define a base terrain upon which objects may be placed. The terrain would usually be represented as a grid of elevation values. As will be presented in detail in Section 3.2, the initial scene is iteratively augmented with new objects according to the rules of the landscape grammar. Each iteration results in a new landscape scene with different objects than that of the previous iteration. This process ultimately results in a final landscape scene which embodies the character defined in the grammar.

3.1.3 Landscape Rules

The rules contained in a landscape grammar express the relationships between object-types described in the vocabulary. Rules are grouped into sets (*rulesets*) which allow rules to be grouped together thematically. The landscape grammar contains one master ruleset (R_α) that contains all other rulesets (R_1, \dots, R_n) and rules (r_1, \dots, r_n),

$$R_\alpha = \{R_1, R_2, \dots, R_n\} \quad \text{Eqn 3.16}$$

$$R = \{r_1, r_2, \dots, r_n\}. \quad \text{Eqn 3.17}$$

Individual landscape rules express permissible relationships between object-types. Rules are of the same general form noted in Chapter 2 in the context of linguistics, namely, IF [precondition] THEN [consequent], or formally:

$$[\text{precondition}] \rightarrow [\text{consequent}].$$

The precondition of a landscape rule is a predicate function, that is, a statement about the contents of a landscape scene that can be affirmed or denied. Examples of a predicate statement include “the object is

a detached residential building”, “the height of the tree is less than ten metres”, or “the wall is adjacent to a road”. There may be and often are several such predicate statements in a landscape rule’s precondition. In such cases, statements are combined into a single compound predicate using the logical operator ‘AND’. It is possible, however, to combine and nest predicates in several layers by use of parentheses and logical operators such as ‘AND’ and ‘OR’. The consequent of a rule is an instruction that, when carried out, modifies the contents of the landscape scene. Like the precondition, the consequent of a landscape rule contains a set of actions each of which are carried out in sequence when the rule is fired by the interpreter. Formally, then:

$$precondition = \{ predicate_1, predicate_2, \dots, predicate_n \} \quad \text{Eqn 3.18}$$

$$consequent = (action_1, action_2, \dots, action_n) \quad \text{Eqn 3.19}$$

and each landscape rule (r) contains a precondition and a consequent as follows:

$$r = (precondition, consequent). \quad \text{Eqn 3.20}$$

Each predicate of a precondition is a truth value function, that is it evaluates to a true/false statement. A predicate states a relationship between objects or values and thus is made up of a relation (φ) and one or more objects or values as arguments (arg) supplied to the relation. The predicate thus takes the form of a function:

$$predicate = \varphi(arg_1, arg_2, \dots, arg_n). \quad \text{Eqn 3.21}$$

In order for reasoning about specific objects in a landscape scene to be performed, predicates must have placeholders into which specific objects from the scene may be inserted. In logical terms, these placeholders are termed ‘variables’ within the predicate, such as a and b in the statement “ a is above b ”. In addition to objects, predicates can contain values that are either constants (such as “3” or “green”), attributes of objects (“roof-type of a house” or “height of a tree”), or returned by a function on a set of objects (“area of a parcel” or “distance from a house to a boundary line”).

The object variables and values are compared to each other using relations. A predicate may relate an object or value to itself, such as “a house (obj) exists (φ)” or “the number 2 (v) is even (φ)”, but more often relates two or more objects or values to each other, such as:

“a wall (obj) is next to (φ) a cliff (obj)”

“a tree’s height (v) is greater than (φ) 20 (v)”

“a tree (obj) is between (φ) a house (obj) and a road (obj)”.

Some relations are spatial in nature, for example, “the house contained within (ϕ) the parcel” or “the hedge is parallel to (ϕ) the wall”. Spatial relations are numerous and can include the following: contains, intersects, inside (or is contained by), outside, between, adjacent to, perpendicular to, parallel to, collinear to, identical to, left of, right of, on, above, below, in front of, and behind. Not all possible spatial relations are rigorously defined here, as the person or group defining a landscape grammar define the only relations suitable to their purposes. It is worth noting, however, that fundamental theoretical research on the nature and definition of spatial relations is still ongoing and could provide a basis for general formalism of landscape grammars as well as other spatial systems (some foundations are found in Egenhofer & Herring, 1990, 1991; Mark & Egenhofer, 1994). Non-spatial relations are also used in landscape grammar rules, but to compare values rather than objects. These relations include standard arithmetic operators ($=$, $>$, and $<$) and logical relations (and, or, and not). Relations between text strings can also be included: $text_1$ begins with $text_2$, $text_1$ ends with $text_2$, $text_1$ contains $text_2$, or $text_1$ matches exactly ($=$) $text_2$.

The examples of predicates presented above are composite predicates that can be deconstructed into relations and arguments according to the general form of Equation 3.21 as follows:

$$\begin{aligned} &=(\text{type}(a), \text{Wall}) \text{ AND } =(\text{type}(b), \text{Cliff}) \text{ AND } \text{is-next-to}(a, b) \\ &=(\text{type}(a), \text{Tree}) \text{ AND } >(\text{height}(a), 20) \\ &=(\text{type}(a), \text{Tree}) \text{ AND } =(\text{type}(b), \text{House}) \text{ AND } =(\text{type}(c), \text{Road}) \text{ AND } \text{between}(a, b, c). \end{aligned}$$

Because the precondition of a landscape rule is a composite predicate, the above logical formulae may be taken as examples of preconditions. In this form, the values are either expressed as functions or constants. Hence, the definition of a precondition may be modified as follows:

$$\text{precondition} = f(\text{obj}_1, \text{obj}_2, \dots, \text{obj}_n, \text{grid}), \quad \text{Eqn 3.22}$$

where the values and predicates of the precondition have been combined into one truth value function, f . Table 3.1 shows further example precondition statements with translations into logical form.

The consequent of a landscape rule is a function that is evaluated only if the precondition of the rule evaluates to true. The consequent can be comprised of a set of actions (Eqn 3.19) which are carried out in sequence resulting in the modification of the landscape scene. The objects used to populate the variables of the precondition populate the same variables in the consequent function. Subsuming the set of actions and any non-spatial values into one function, the previous definition of a rule’s consequent (Eqn 3.19) can be modified as follows:

<u>Rule</u>	<u>Precondition (IF)</u>	<u>Logical Form</u>
r ₁	The distance from a house, <i>a</i> , to a road, <i>b</i> , is less than 25 feet	type(<i>a</i>) = House AND type(<i>b</i>) = Road AND distance(<i>a</i> , <i>b</i>) < 25
r ₂	The average slope of a parcel, <i>a</i> , is greater than 0.2	type(<i>a</i>) = Parcel AND slope(<i>a</i>) > 0.2
r ₃	There exists a tree, <i>a</i> , the species of which is cedar, and is inside a building envelope, <i>b</i>	type(<i>a</i>) = Tree AND species(<i>a</i>) = "cedar" AND type(<i>b</i>) = Building-Envelope AND inside(<i>a</i> , <i>b</i>)
r ₄	The zoning of a parcel, <i>a</i> , is Residential, and the area of a building envelope, <i>b</i> , inside of <i>a</i> , is greater than 2500 square feet	type(<i>a</i>) = Parcel AND zoning(<i>a</i>) = "Residential" AND type(<i>b</i>) = Building-Envelope AND inside(<i>b</i> , <i>a</i>) AND area(<i>b</i>) > 2500
r ₅	A tree, <i>a</i> , is designated a "shade tree"	type(<i>a</i>) = Tree AND type(<i>a</i>) = Shade-Tree

TABLE 3.1 EXAMPLE PRECONDITIONS FOR LANDSCAPE RULES

<u>Rule</u>	<u>Consequent (THEN)</u>	<u>Logical Form</u>
r ₁	Plant a five-foot hedge along the edge of the road, <i>b</i>	let <i>c</i> = copy(edge(<i>b</i>)) set type(<i>c</i>) = Hedge
r ₂	Designate the parcel, <i>a</i> , as inappropriate for development	set develop(<i>a</i>) = false
r ₃	Reduce the building envelope, <i>b</i> , until it does not contain the tree, <i>a</i>	until(not inside(<i>a</i> , <i>b</i>), reduce(<i>b</i>))
r ₄	Apply the ruleset that examines the topographic suitability of a site to parcel, <i>a</i>	interpret(TopographicSite-Ruleset, (<i>a</i> , <i>grid</i>))
r ₅	Let the species of the tree, <i>a</i> , be assigned by these probabilities: <i>delonix regia</i> (35%), <i>eriobotrya japonica</i> (20%), <i>plemeria rubra</i> (20%), <i>melia aedarach</i> (15%), <i>ficus retusa</i> (5%), <i>araucaria excelsa</i> (5%)	set species(<i>a</i>) = selectbyprob("delonix regia", 0.35, "eriobotrya japonica", 0.20, "plemeria rubra", 0.20, "melia aedarach", 0.15, "ficus retusa", 0.05, "araucaria excelsa", 0.05)

TABLE 3.2 EXAMPLE CONSEQUENTS FOR LANDSCAPE RULES

$$consequent = g(obj_1, obj_2, \dots, obj_n, grid), \quad \text{Eqn 3.23}$$

where g modifies the scene and $obj_1 \dots obj_n$ is the same ordered set of objects that caused the precondition to evaluate to true. Table 3.2 shows example consequent statements and their translations into a logical form. The rule numbers in the left hand column correspond to the rule numbers in Table 3.1. Given the revised definitions of a precondition and consequent in Eqns 3.22 and 3.23, the definition of a rule is modified accordingly from Eqn 3.20 to:

$$r = (f(obj_1, obj_2, \dots, obj_n, grid), g(obj_1, obj_2, \dots, obj_n, grid)), \quad \text{Eqn 3.24}$$

where f is a compound predicate, g is a function modifying the scene of objects and the rule may be read as “if f returns true, then evaluate g ”.

It was noted at the beginning of this section that rules are grouped into rulesets. The consequent of the rule, r_4 , in Table 3.2 contains an instruction to apply a ruleset to a set of objects. This example demonstrates the nested nature of grammar interpretation. A new interpretation process is begun in this instance with a subset of the master ruleset and a subset of the objects in the scene. This serves two practical purposes: (i) to modularize landscape knowledge thematically into more specific domains and (ii) to localize geographically the application of rules to a specific part of the landscape scene. Thematic rulesets can contain rules designed to represent specialized knowledge on building placement, road configuration or garden landscaping. The rules of a ruleset can be applied to a particular set of objects that are located in one part of the landscape scene. The consequent of the rule, r_4 , of Table 3.2 makes use of this feature by containing an action that invokes the interpretation of rules in another ruleset on a particular subset of objects from the scene. The thematic and geographic restriction of a landscape grammar can therefore be considered a sub-grammar where, if $LG(V, R, S)$ represents the application of landscape grammar to an entire scene, $LG(V, R', S')$ represents the restricted application of a subset of rules to a subset of the scene where $R' \subseteq R$ and $S' \subseteq S$.

Landscape grammars are also parametric. The parameters can be non-spatial or spatial and supplied either as constants within a rule, attribute values associated with an object, or as values returned by a function on an object. The preconditions of rules r_1 , r_2 and r_4 in Table 3.1 illustrate this. The spatial geometries of landscape objects are also parametric. The vocabulary of a parametric shape grammar has a set of shape primitives with a fixed set of parameters, such as rectangles with width and height parameters. The spatial model for landscape grammars is more flexible as lines and polygons are sets of points that may be positioned anywhere in space and ordered in different sequences, allowing the landscape objects to take many sizes and shapes.

Rule r_5 from Table 3.2 illustrates the use of stochasticity in landscape rules. The outcome of the

rule, as specified in the consequent, is determined probabilistically. In this example, the consequent establishes the species of a tree object according to a set of possible species, each of which has a probability or likelihood that that species occurs in the situation identified by the rule's precondition. Instead of having a determined outcome, as is typical of a traditional shape grammar rule, the consequent of a stochastic landscape grammar rule is thus selected probabilistically. This allows objects and their features to display some controlled randomness within the landscape. The use of stochastic rules is particularly suited to an empirical approach to landscape grammar development in which the landscapes of a region are surveyed to determine the likelihood of particular objects and configurations occurring in specific places.

The formal structure of a landscape grammar is now defined in terms of its vocabulary of object-types, sets of rules each containing a precondition and a consequent and as a scene of landscape objects. The original landscape grammar definition can thus be expanded from:

$$LG = \{V, R, \alpha, IS\} \quad \text{Eqn 3.25}$$

$$\text{to } LG = \left\{ \begin{array}{l} \{OT_1, OT_2, \dots, OT_n\}, \\ \{R_1, R_2, \dots, R_n\}, \\ \{obj_1, obj_2, \dots, obj_n, grid\} \end{array} \right\} \quad \text{Eqn 3.26}$$

$$\text{and } LG = \left\{ \begin{array}{l} \{\{\gamma_1, st_1, \{ad_1, ad_2, \dots\}\}, \{\gamma_2, st_2, \{ad_3, ad_4, \dots\}\}, \dots\}, \\ \{\{r_1, r_2, r_3, \dots\}, \{r_4, r_5, r_6, \dots\}, \{r_{29}, r_{13}, r_{44}, \dots\}, \dots\}, \\ \{obj_1, obj_2, \dots, obj_n, grid\} \end{array} \right\}. \quad \text{Eqn 3.27}$$

This definition of a landscape grammar provides an overview of the parts that have been reviewed in this section. The vocabulary is made up of object-type definitions that are manifested in scenes as objects and utilized in the rules of rulesets to express typical relationships between object-types. The vocabulary provides generalized definitions of those features found recurring in a regional landscape. The rules state how they recur relative to each other. When the rules are applied to an initial scene, the scene is populated with objects that are instances of the general concepts expressed in the vocabulary. This provides the structural framework in which landscape character may be defined. The process of applying the grammar to an initial scene addresses the challenge of testing a definition of landscape character using this framework.

3.2 Landscape Grammar Processing

While the preceding section described the structure of a landscape grammar, this section

describes the process in which the grammar is interpreted, generating a landscape scene. The grammar is said to be applied to a scene by a *landscape grammar interpreter* which is a person or computer program that facilitates the interpretation process. The general process is shown in Figure 3.4. The processes outlined in bold rectangles are subsequently decomposed into further flow diagrams. The interpretation process for a landscape grammar must start with a vocabulary, master ruleset and initial scene (V , R_{α} and IS). The scene is iteratively modified and termed the *working-scene*. The interpretation process involves three main phases. First, a set is compiled of each rule in the ruleset for which the precondition of the rule is true, when objects from the working-scene are substituted for the precondition variables. If no rules can be applied to the scene, the scene is complete and the process ends. Second, one or more of the matching rules are selected and third the consequent of each selected rule is fired or instituted in the working-scene.

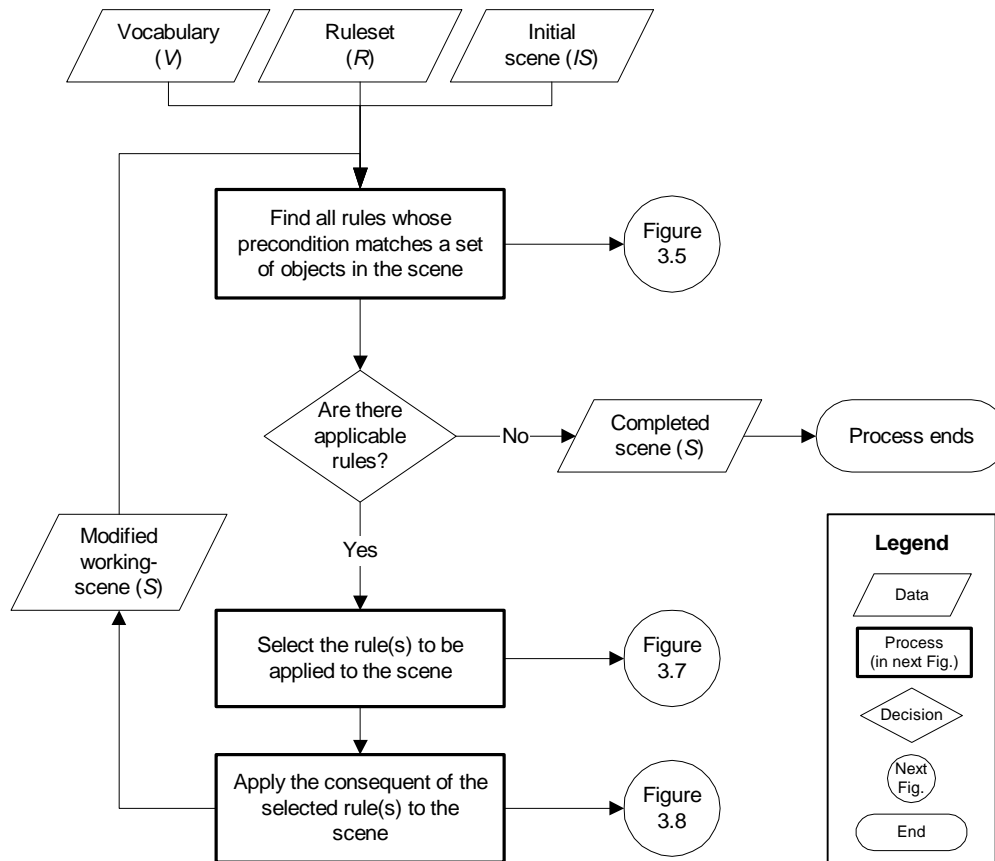


FIGURE 3.4 THE LANDSCAPE GRAMMAR INTERPRETATION PROCESS

The first process is the identification of all rules in the ruleset whose precondition matches some subset of the landscape objects in the working-scene. Figure 3.5 shows this process in more detail. Each

rule in the ruleset must be compared to the working-scene to identify all of the combinations of objects that match the precondition of the rule. If no matching sets of objects are identified, the rule is not applicable to the scene and is discarded from consideration until the next iteration of the interpretation process in Figure 3.4. When all of the rules that have a matching set of objects in the scene are identified, these ‘matching rules’ are returned to the interpretation process in Figure 3.4 to determine which of those rules and objects will be fired.

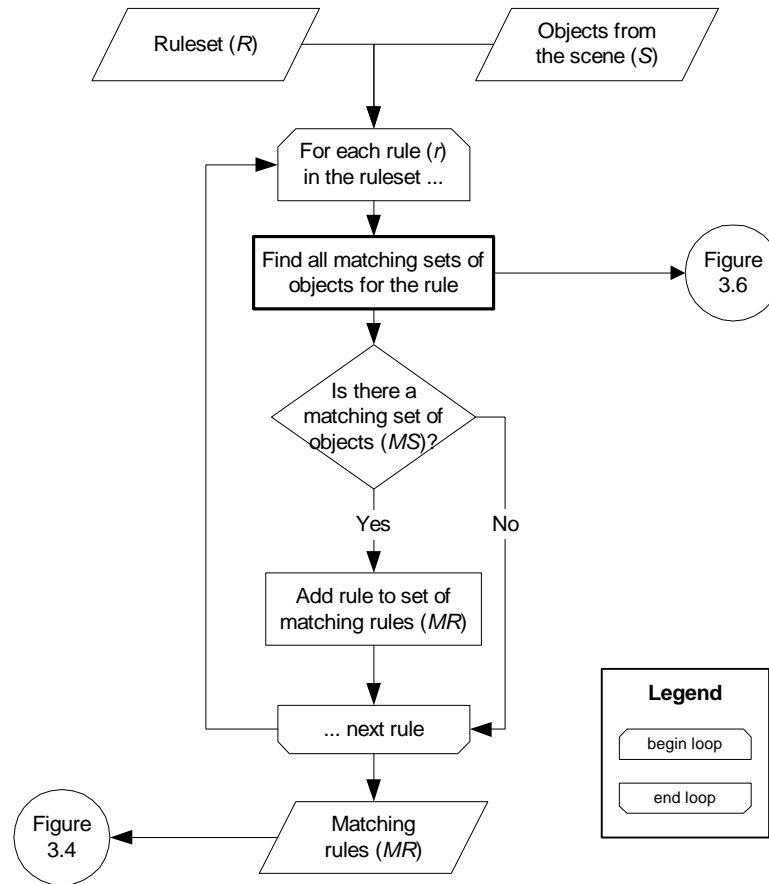


FIGURE 3.5 IDENTIFICATION OF THE SET OF MATCHING RULES

The process for identifying the sets of objects from the scene that match a rule’s precondition is presented in Figure 3.6. Objects from the scene are progressively substituted for variables in the rule’s precondition. The scene is comprehensively swept to find all possible combinations of objects that can be used for the set of variables. With each set of objects, the precondition function (Eqn 3.22) is evaluated to see if the statements in the precondition are confirmed. If so, the set of objects is referred to as a *matching set (ms)* for the rule:

$$ms(S, r) = (obj_1, obj_2, \dots, obj_n \mid precondition_r), \quad \text{Eqn 3.28}$$

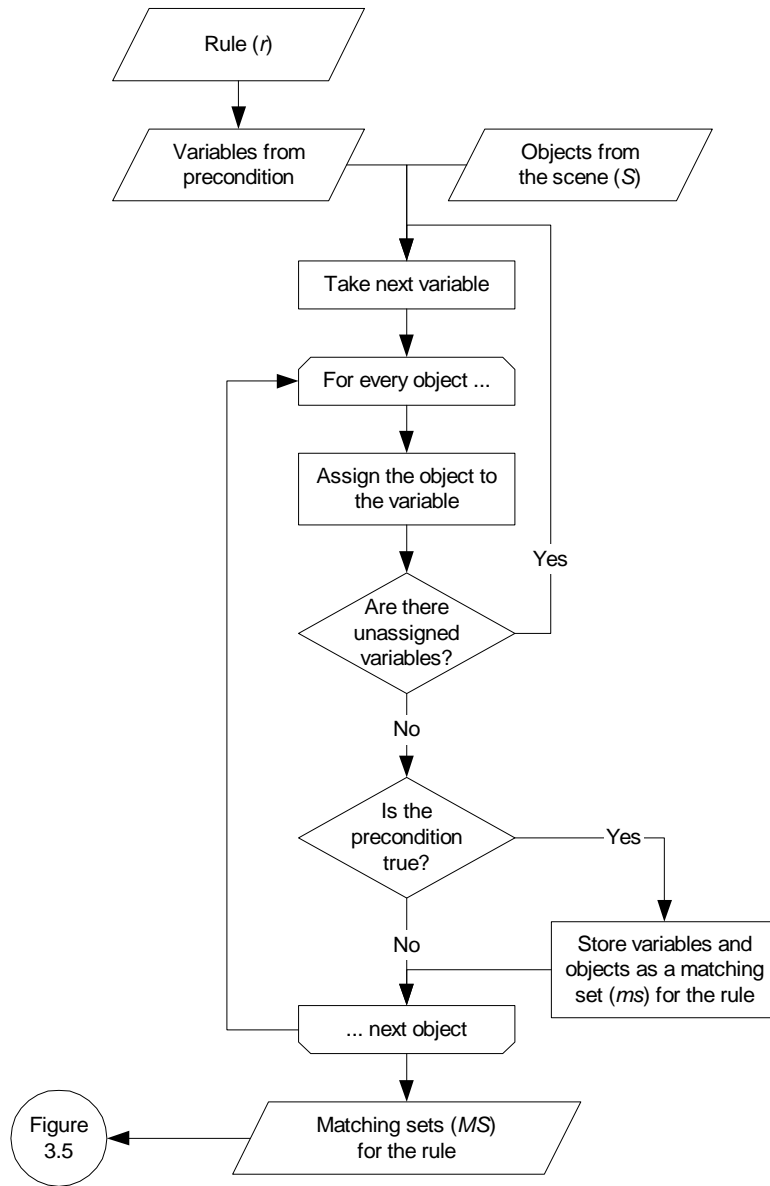


FIGURE 3.6 IDENTIFICATION OF THE MATCHING SETS OF OBJECTS FOR A RULE

where $ms \subseteq S$ and the cardinality of ms is the number of variables in the precondition. It is possible that a rule will have more than one matching set. The process continues to search for all of the matching sets for the rule that can be found in the current working-scene. This phase of the process may then be formalized as the compilation of the set of all matching sets (MS) from the working-scene:

$$MS_r(S, r) = \{ms_1, ms_2, \dots, ms_n\}. \quad \text{Eqn 3.29}$$

Since this process is a comprehensive search through all permutations of a set of objects, MS_r is also a subset of the power set of S (the set of all possible subsets of S),

$$MS_r \in \rho(S). \quad \text{Eqn 3.30}$$

Returning to Figure 3.5, any rule (r) that has a matching set (that is, the set MS_r is not empty) is termed a *matching rule*. The set of all matching rules (MR) is given as:

$$MR = \{r \mid MS_r(S, r) \neq \emptyset\}, \quad \text{Eqn 3.31}$$

that is, all of the rules for which there exists some set of objects from S , that makes the precondition of r true. Thus, the first phase of the interpretation process (Figure 3.4), given V , R , and S , entails finding MR , as well as MS for each r in MR .

The second phase of the general interpretation process (Figure 3.4) is to select, by some criteria, a subset of matching rules and a subset of matching objects to which each selected rule will be applied, or *fired*, in the third phase. In Section 2.7.2, methods for selecting and firing rules were presented in terms of being serial or parallel. Rules are fired serially if only one of the matching rules from MR is fired per iteration and in parallel if multiple matching rules are fired. Similarly, any selected rule is applied to objects in the scene serially if it is only applied to one set of matching objects (ms), or in parallel if applied to more than one ms . Caveats were noted in that section about parallel operations, since the firing of one rule modifies the objects in the scene and in so doing modifies objects that are part of a matching set for a matching rule. The danger here is that the scene objects will be modified in such a way that the precondition of another rule is no longer true and the set of objects no longer matching, therefore rendering the firing of that formerly matching rule invalid. For example, consider a simple scene that contains two objects {Tree-1, House-2} and a rule set that contains two rules {Rule-1, Rule-2}. Suppose Rule-1 says, “if a tree is less than 3 metres from a house then remove the tree”, while Rule-2 says, “if a tree is labelled ‘rare’ then insert a protective fence surrounding the tree”. If Tree-1 is a ‘rare’ tree and within 3 metres of House-2, then both rules are eligible for firing. Clearly, if both are selected for firing and Rule-1 is fired first, then Tree-1 will be removed and there will no longer be a rare tree in the scene to satisfy the precondition of Rule-2 or to refer to in erecting protective fencing. Rule-2 can thus no longer be fired. In this case, these rules cannot be processed in parallel.

In Section 2.7.2, it was concluded that spatial grammar rules can be fired in parallel if it can be assured that the changes to the design caused by the rules are independent of each other and that a spatial grammar rule can be fired in parallel if the objects that are to be modified by the rule are independent of each other with respect to the actions specified in the rule’s consequent. Shape grammars were identified

as being serial (one rule is fired on one object per iteration) while cellular automata are often fired in parallel (on each iteration, each cell-value is modified according to some rule).

Because landscape grammars incorporate object- and grid-based landscape representations, both serial and parallel operations are desirable as long as the above caveats are satisfied. In terms of the formulae in this section, the second phase of landscape grammar interpretation involves the selection of one or more rules from MR , and for each selected rule (r^*) the selection of one or more matching sets of objects from MS_{r^*} . Figure 3.7 displays the serial process of selecting a rule and a set of matching objects for firing. A rule is selected from the set of matching rules (MR) according to an ordered set of rule

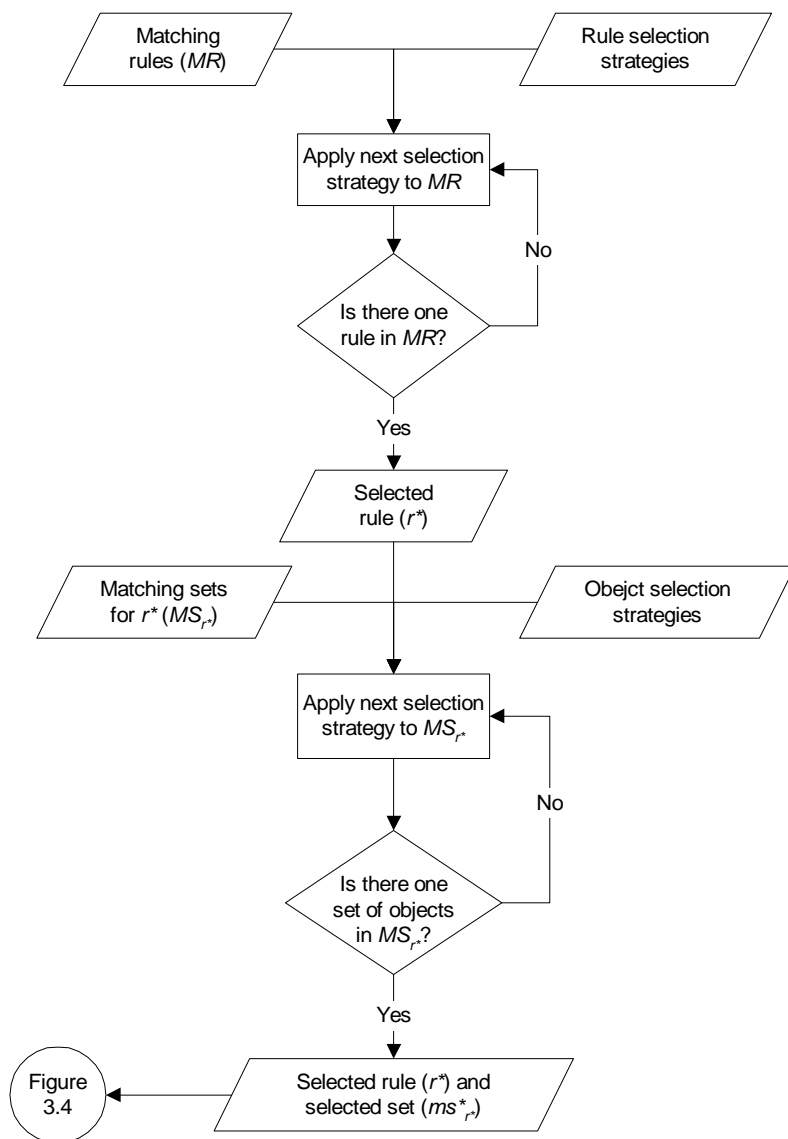


FIGURE 3.7 SERIAL SELECTION OF RULES AND OBJECTS FOR FIRING

selection strategies. A *strategy* is a function that reduces the size of MR . Each strategy may be viewed as an evaluation function on a rule, $f(r)$, that returns some value for that rule. Rules are thus selected from MR on the basis of the minimization or maximization of $f(r)$. The general problem for generating the set of selected rules for firing (MR^*) is given formally as follows:

$$MR^* = \{ r \in MR \mid \min \text{ or } \max f(r) \}, \quad \text{Eqn 3.32}$$

where f is an evaluation function. In serial processing, the cardinality of MR^* is one. The first strategy is chosen from the strategy set and applied to MR . Each of the subsequent strategies in the strategy set are chosen in turn and applied to MR until, in the case of serial processing, only one matching rule remains in MR . In parallel processing, the strategies reduce the numbers of rules to an acceptable limit or simply allow all rules to be processed.

Each of the selected rules ($r^* \in MR^*$) has a set of sets of matching objects (MS_{r^*}), each of which makes the precondition true. Like rule selection strategies, object selection strategies reduce the number of these matching object sets to one set (for serial processing) or an acceptable number of sets (for parallel processing). Each object selection strategy is an evaluation function on a set of matching objects, $f(ms)$, and sets of objects are selected from MS_{r^*} by minimizing or maximizing $f(ms)$:

$$MS_{r^*}^* = \{ ms \in MS_{r^*} \mid \min \text{ or } \max f(ms) \}, \quad \text{Eqn 3.33}$$

where f is an evaluation function and the cardinality of $MS_{r^*}^*$ is one for serial processing. Like the rule selection strategies, the object selection strategies are applied to $MS_{r^*}^*$ one at a time until the acceptable number of object sets remains. The final outcome of this phase of the interpretation process is a selection of rules and objects that are then passed to the third phase for firing.

The criteria used in the rule and object selection strategies are varied and may differ in different circumstances. The selection strategies that are used will depend largely upon what is useful for the regional landscape character modelled by the grammar. However, some categorization of selection strategies can be made. The simplest strategies merely take the first element or a random element of a set, whether rules or objects. These strategies can be used as “last-resort” strategies to ensure that only one rule is selected. Some strategies have a functional purpose within the grammar. For example, a “least recently fired rules” strategy selects rules that have not been fired in recent iterations of the interpreter. The purpose of this strategy is to avoid firing the same rule in iteration after iteration. Similarly, the “rules with matching sets not fired” strategy eliminates those rules whose matching objects sets have all been fired previously. This reduces the occurrence of the same rules firing on the same objects in the scene and so duplicating their effects.

Selections can also be made using thematic criteria. “Attribute strategies” select the matching objects to which a rule will be applied using information associated with the objects as attribute values. For instance, a “tallest first” strategy function might return for a rule the set of matching objects with the largest values in their Height attribute. The selection equation above (Eqn 3.33) would be modified as:

$$MS^*_{r^*} = \left\{ ms \in MS_{r^*} \mid \max \sum Height(obj \in ms) \right\}. \quad \text{Eqn 3.34}$$

Likewise, a strategy function can be designed to select objects that are instances of a particular object-type. The values used to evaluate objects for selection need not be stored as attributes but may instead be derived from a function on the objects. Thus, a strategy could minimize an index calculated by dividing the average slope of an area of land by its average soil depth, thereby forcing the rule to favour flat areas with deep soil. The selection function might appear as follows:

$$MS^*_{r^*} = \left\{ ms \in MS_{r^*} \mid \min \frac{\overline{slope}(ms)}{\overline{depth}(ms)} \right\}. \quad \text{Eqn 3.35}$$

Rules can also be selected thematically. In rural landscapes, rules that subdivide a scene using concession roads might be fired before rules that place buildings on the landscape. Selecting rules according to their function in the landscape allows a definitive order to be established in the grammatical construction of a scene. The sequencing of rules in this way can be used to represent the typical evolution of a landscape as exhibited in a particular scene. Rules can also be discerned thematically based on the attributes of the objects in the rules’ matching sets. Hence, a rule selection strategy can favour rules whose matching sets contain, or do not contain, objects of a certain object-type. For example, given a scene on sloping ground, rules that operate on split-level houses may be selected preferentially over rules that operate on single-floor houses.

The selection of objects and rules can be performed using spatial criteria instead of attributes or functional values. A “proximal strategy” selects objects that are closest to a significant object in the scene. For example, buildings may be inserted in lots that are closer to a town centre before more distal locations. The selection function such a strategy can be expressed as:

$$MS^*_{r^*} = \left\{ ms \in MS_{r^*} \mid \min \sum_{i=1}^n dist(obj_i, TownCentre) \mid obj \in ms \right\}. \quad \text{Eqn 3.36}$$

The strategy could be modified to select objects that are proximal to those objects that were modified by the rule fired in the last iteration of the interpreter. The effect of using a proximal strategy in this way is to localize the grammar’s operations to a central area in the scene during the early iterations of the interpreter, and then progressively move its operations away from that core with further iterations.

Conversely, a “distal strategy” maximizes this distance rather than minimizing it, moving grammar operations towards a central area. A distal strategy can also be used as an initial rule to select objects that are most distant from a dangerous object such as a ravine.

A “clustering” strategy minimizes the sum of the distances between each of the objects in the set, as represented by the equation:

$$MS^*_{r^*} = \left\{ ms \in MS_{r^*} \mid \min \sum_{i=1}^n \sum_{j=1, j \neq i}^n dist(obj_i, obj_j) \mid obj \in ms \right\}. \quad \text{Eqn 3.37}$$

Alternatively, a clustering strategy could be based on minimizing the area of the convex hull surrounding a set of objects:

$$MS^*_{r^*} = \left\{ ms \in MS_{r^*} \mid \min area(ConvexHull(ms)) \right\}. \quad \text{Eqn 3.38}$$

Since rules are generalizations about landscape character, they are not grounded to the specific geography of a scene. However, when the variables of a rule’s consequent are populated with objects from the scene, the rule is effectively related to a specific geographic part of the scene that includes the selected objects. Each matching rule can therefore be associated with an area within the scene and this information can then be put to use in discriminating between rules. For instance, the proximal strategy for selecting objects for a rule can be further employed to select rules whose matching objects have the minimal cumulative proximity. It should be clear that the rule and object selection strategy functions employed in a landscape grammar interpreter are diverse and limited only by the creativity of the landscape grammar developer.

The third phase of the grammar interpretation process is to perform the consequent of the selected rule(s) using the matching set(s) of objects as parameters. This action is referred to as *firing* the rule(s) and entails the evaluation of the consequent function as defined in Eqn 3.23. The variables of a rule’s consequent are populated with the objects from the selected matching set and the consequent is evaluated, thereby carrying out the actions stated in the consequent. As shown in Figure 3.8, in the case of parallel processing, each selected rule (r^* in MR^*) is evaluated in turn using each of its selected sets of matching objects (ms in $MS^*_{r^*}$). In serial processing, there is only one r^* and one ms . The firing of a rule typically modifies the working-scene in some way, either through the addition of new objects, modification of existing objects, or the modification of the attributes of existing objects. This marks the end of one iteration of the landscape grammar interpreter. The modified scene then becomes the new working-scene and is introduced as an input to the next iteration of the interpreter using the same vocabulary and ruleset (Figure 3.4).

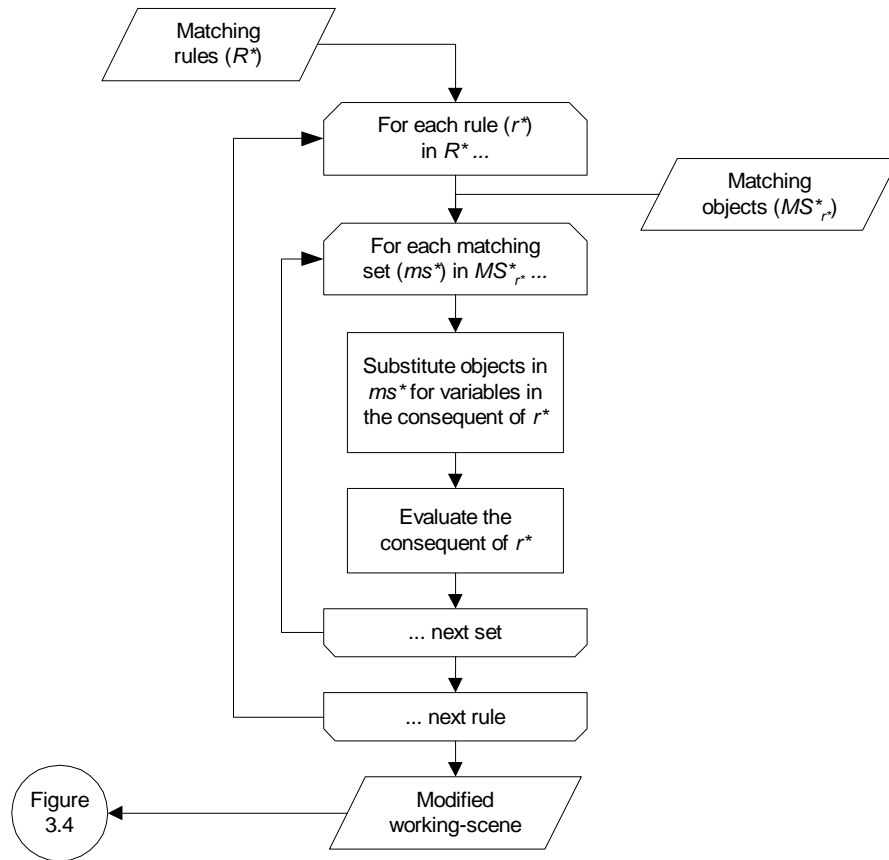


FIGURE 3.8 FIRING OF SELECTED RULES ON SELECTED OBJECTS

A simplification of one full iteration of the interpreter with excerpts from the equations of this section is described in Figure 3.9. Only one rule is used in the illustration and three matching sets of objects are identified that match the rule's precondition. One of these sets is selected using object selection strategies and subsequently inserted into the variables of the rule's consequent. The consequent is then fired resulting in a new working-scene that subsequently becomes the input for the next iteration of this process (as symbolized by the arrow at the top-right of Figure 3.9).

The firing of a landscape rule need not always modify the scene. A rule's consequent can signal the artificial termination of the interpretation process. Alternatively, the consequent can start a new interpretation process with restricted subsets of landscape rules and objects, as was illustrated previously by the consequent of the rule, r_4 , in Table 3.2. This means that the interpretation process in Figure 3.4 is begun anew as a result of evaluating the consequent of a rule in Figure 3.8. The new interpretation process (Figure 3.10) is supplied with a subset of the rules from the current ruleset (R) and a subset of the objects from the current working-scene (S), which serve respectively as the master ruleset (R') and initial scene (IS') for the new process. As mentioned earlier, such nesting of grammars allows for subsets

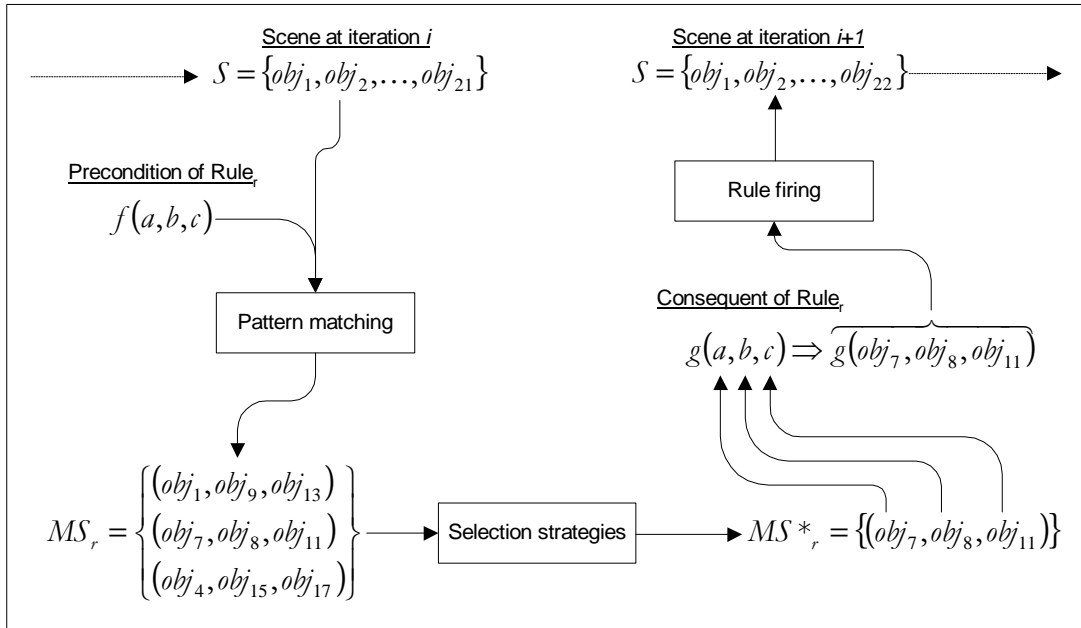


FIGURE 3.9 THE LANDSCAPE GRAMMAR INTERPRETATION PROCESS WITH FORMULAE

of landscape knowledge to be applied in a controlled context allowing, for example, rules relating to road design to operate separately from rules for the design of a formal garden. Likewise, the scene of the nested interpretation is restricted to a subset of objects thereby localizing the grammar operations to those objects that are appropriate for the subset of rules. Rewriting the equation for a rule (Eqn 3.24) for this case in which the consequent commences a new landscape grammar interpretation on a subset of the rules and objects is achieved as follows:

$$r = (f(obj_1, obj_2, \dots, obj_n), LG(V, R', S')), \quad \text{Eqn 3.39}$$

where $R' \subseteq R$ and $S' \subseteq S$ hold for the current grammar. The matching set of objects $(obj_1 \dots obj_n)$ may or may not be part of S' and vice versa. It is worth noting that nested interpretations can be used to achieve the same localization of grammar operations that was described above for rule and object selection strategies. The nested interpretation iterates until all rules in R' have been exhausted at which point a completed landscape scene (S') is returned to the main interpretation process, and rule-processing returns to the point of the main iteration at which the rule had been fired.

3.3 Visualization of the Landscape Language

At each iteration of the interpretation process, a scene of landscape objects is available and this scene is the product of a grammatical interpretation. The general equation for a scene, S , at iteration, i , at

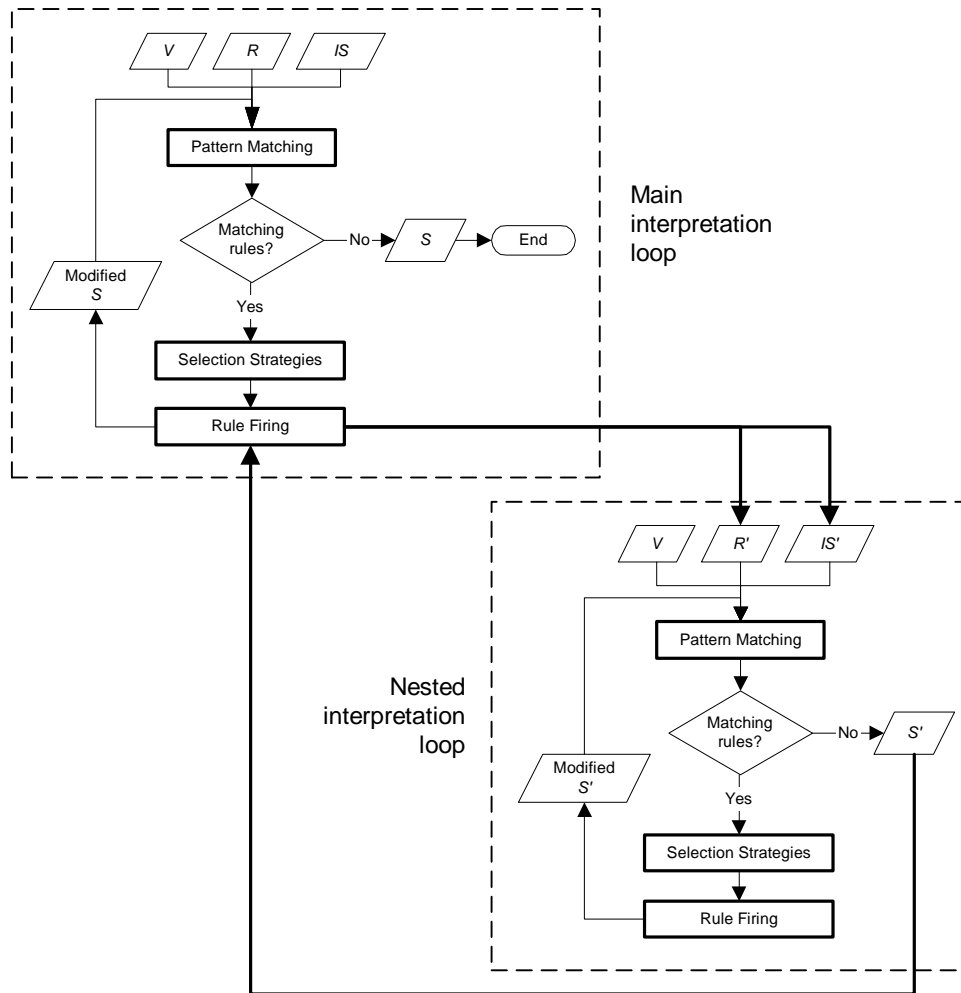


FIGURE 3.10 NESTED GRAMMAR INTERPRETATION

which a rule, r , is fired, is given as:

$$S_{i+1} = [S_i - precondition_r] + consequence_r \quad \text{Eqn 3.40}$$

(adapted from Stiny 1980a and Krishnamurti & Stouffs, 1993). Since each object has a geometry, the scene can be represented graphically as a collection of geometric objects with specific locations in the same coordinate space. The display of these objects together, whether in two or three dimensions, constitutes the visualization of the scene. The graphical scene represents the visual product of the landscape grammar, that is, a landscape simulation that represents one of the many possible landscapes that embody the landscape character defined in the grammar. If the landscape grammar interpreter were to be processed again, with the same vocabulary, ruleset and initial scene, a different final scene would

more than likely be generated. Thus, a different scene of objects is assembled with each completed grammar interpretation.

The full range of these scenes is the *landscape language*, which comprises the set of all scenes that can possibly be generated by a given landscape grammar and its interpretation as described above. This is completely analogous to a linguistic language that is comprised of all the possible sentences that can be constructed using the vocabulary and grammar of that language. Theoretically, the landscape character defined by the grammar is visualized by generating and visualizing all of the possible scenes in the landscape language. The landscape character is visualized through a series of generated scenes, rather than a single scene, because it is the recurrent qualities of a place that define a visual landscape character (Section 2.3) and recurrence must be visualized through multiple scenarios.

When landscape grammar rules are stochastic, the size of the language, that is the number of possible scenes that could be generated, is increased enormously. This presents a fundamental problem inherent in grammatical systems: unless the grammar is highly limited in scope, the full and potentially infinite range of scenes cannot be generated in anything other than a theoretical formalism. The problem for the evaluator of the landscape grammar is then to determine the number of scenes that should be viewed in order to represent the nature of the landscape character in a reasonable manner. While it is difficult to derive a formula to calculate the number of scenes that should be generated for a given landscape grammar, it can be stated that more scenes should be visualized for grammars that contain higher degrees of variance in their rules in order for the sample of scenes to be considered representative of the landscape language. For example, a rule that states that the area of a building's footprint will be a number between 1,000 and 2,000 square feet produces less variance in the generated landscapes than if the rule required a parameter value between 1,000 and 5,000 square feet. For practical purposes, the variance between landscape scenes is assessed visually.

The unmanageable size of the landscape language may be addressed to some extent by letting the evaluator choose the path that the interpretation process follows through the solution space. However, if one of the purposes of the grammar is to find its unexpected landscape scenes, then such intervention defeats that purpose to some extent. Another approach is to bias the interpretation process toward particular parts of the language. For instance, in order to avoid the generation of scenes representing all possible building heights, the values of a building height parameter can be maximally biased to produce scenes that contain relatively tall buildings. In visual landscape planning scenarios, these scenes might represent 'worst-cases', which usually warrant the highest concern. This can be achieved methodologically by either modifying the range of values assigned to a building height directly within the rules or through the use of selection strategies that select rules with maximal height values. A third method is to apply constraints to the results that cull unacceptable scenes from the landscape language

Despite these methodological approaches to traversing the vast range of possible landscapes, the purposes to which the grammar is put can alleviate some of the need for comprehensive landscape generation. Planners are not currently able to explore a large number of possible landscape models. Landscape grammar systems such as that described in this chapter can thus prove useful even if they do no more than allow a greater exploration of the possible landscape outcomes than is currently available through a planner's intuition. Exhaustive scene generation is not then a requirement.

3.4 Summary

This chapter has laid the theoretical foundations for the structure and processing of a landscape grammar. The theory is derived from the concepts introduced in Chapter 2 pertaining to linguistic grammars and spatial grammars. Landscape grammars are an application of spatial grammars that utilize conceptual hierarchies as well as set structures and object- and grid-based landscape representations. Objects are instances of the conceptual object-types and collections of objects comprise a scene. A landscape rule is comprised of a precondition and a consequence. The former is a compound predicate that can be decomposed into relations and variables, and the substitution of objects for the variables leads to a true/false evaluation of the precondition statement. Selection strategies are employed to reduce the number of rules whose preconditions are found to be true in a landscape scene and to reduce the numbers of objects to which the rule will be applied. The execution of the rule's consequent modifies the landscape scene and the interpretation process begins again with the modified scene. This process iterates until no more rules are applicable, producing a final scene of objects that when visualized exhibits the landscape character as defined by the grammar. By generating and visualizing several landscape scenes, a sense of the nature of the landscape character definition may be gained. The theoretical structures and processes laid out in this chapter serve as the foundation for the implementation of a functional landscape grammar system. This is described in the following chapter.

Chapter 4

Implementation of Landscape Grammars

The language-landscape metaphor elucidated in Chapters 2 and 3 allows the definition of landscape character through the use of a grammatical framework. It was shown in Chapter 2 how the adaptation of generative linguistic grammars to spatial objects has informed the methodical construction of spatial patterns. The subsequent application of spatial grammars to landscape provides the means to define a landscape character and to use it to generate landscape scenes. This definitional framework and generative process can be deconstructed into a formal landscape grammar theory using set theory formalisms and an iterative grammar interpretation mechanism as set out in Chapter 3. The implementation of this theory into a functional system allows the examination of how the theoretical notions can be put into practice. Hence, this chapter presents the transformation of theory into practice by outlining the implementation of a computer-based system to define and apply landscape grammars. Specifically, the chapter introduces a new landscape grammar interpreter system that takes the theoretical constructs from Chapter 3 and implements them on a desktop computer. The system facilitates the definition of a vocabulary and rules and transforms these into a stored knowledge-base on landscape character. The system also provides an interpreter mechanism that processes rules to build a grammatically generated landscape scene that is represented as a geographically referenced database of spatial objects.

The objectives for the implementation are first presented followed by an assessment of programming development environments and the methods that were used with the selected platform. The implementation of a facility for defining a landscape grammar is then described, specifically implementing landscape object-types, objects and rules from Chapter 3 in a knowledge-based geographic information system (GIS). The functionality of the interpreter mechanism for processing the defined landscape grammar is then presented. The chapter concludes with an example of the generation of a simple landscape scene and an explanation of the export of data for 3D visualization.

4.1 Implementation Objectives & Methodology

The main objective of this aspect of the dissertation research is to describe a ‘proof-of-concept’ prototype system and explore its potential for use in defining landscape character and generating scenes

in a simulated environment. It is required that the system be able to demonstrate the capabilities discussed in the previous two chapters, namely to represent the descriptive landscape concepts as generic object-types, to allow the relationships between concepts to be defined by grammatical rules each with a precondition and consequent and to utilize those rules to generate scenes of spatial objects with attribute data. Due to the demonstrative and research nature of the grammar implementation, the creation of a user-friendly interface that could be employed practically in a planning agency and used by their staff was not a requirement and therefore not developed.

4.1.1 Selection of a Development Environment

Several programming and software environments were considered for the development of a functional landscape grammar system. In AI research, there is a well-established retinue of object-oriented approaches to general knowledge representation. In object-oriented programming, 'classes' of objects are defined with predetermined 'properties' and 'behaviours' and are organized into a hierarchy of subclasses and superclasses. Programming routines are executed on object 'instances' that are created from the class definitions. When an instance of a class is created, each property of the class refers to a 'slot' in the instance and each slot contains a data 'value'. Since landscape grammars are a representation of knowledge pertaining to a landscape character, it was decided early in the research to use an object-oriented programming paradigm for development of the landscape grammar interpreter. Figure 4.1 illustrates the object-oriented terms using a landscape-based example from Chapter 3. The equivalencies between the terms of object-oriented programming and those of landscape grammar formalisms from Chapter 3 are shown in Table 4.1. The object-oriented concepts of 'classes' and 'properties' (or 'slots') are the equivalent of object-types and attribute definitions, respectively. The classes are manifested as 'instances' and 'values' which are respectively the equivalent of objects and attribute values from Chapter 3. In an object-oriented landscape grammar implementation, therefore, a vocabulary is comprised of landscape classes and generates scenes of instances of those classes. In describing the implementation of this dissertation, the programming terminology will be adopted: a vocabulary of landscape classes is used to create scenes of instances (or objects) whose slots contain values.

Section 2.8 of Chapter 2 identified desirable features for a computer-based landscape grammar system. These included the use of both grid and vector landscape representation models augmented with topological relationships and attribute data. A modular knowledge-base facility for storing spatial landscape rules was also required, preferably involving the use of parametric and stochastic elements. Finally, an interpreter mechanism capable of constructing landscape scenes from the knowledge-base in a sequential manner was necessary. In addition to these three required components, an ideal landscape grammar system would incorporate some further features. Three-dimensional solid modelling, that is the

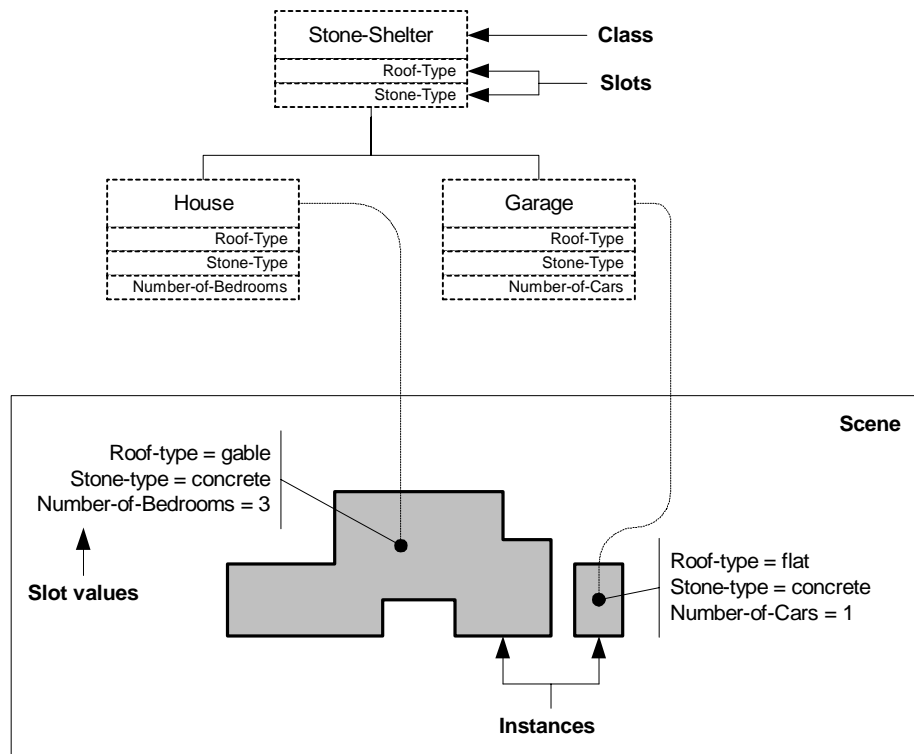


FIGURE 4.1 THE COMPONENTS OF OBJECT-ORIENTATION FOR LANDSCAPES

Object-Orientation term	Landscape Grammar term
Class	Object-type (<i>OT</i>)
Superclass/subclass	Supertype/subtype
Property/Slot	Attribute definition (<i>ad</i>)
Behaviour	(not defined)
Instance	Object (<i>obj</i>)
Value	Attribute value (<i>v</i>)

TABLE 4.1 RELATION OF OBJECT-ORIENTATION TO LANDSCAPE GRAMMAR THEORY

representation of landscape objects in three dimensional objects, with the ability to combine and manipulate these objects would be advantageous for more realistic landscape simulations. A large library of spatial functions is also advantageous for use in the preconditions and consequents of rules. These functions would provide the facilities for determining spatial relations between objects, such as whether a line intersects another line or whether a point lies inside a polygon, as well as spatial operations, such as calculating the intersection point of two lines, the centroid of a polygon or the minimum bounding rectangle of a set of points.

Ideally, these features would all be available within an open programming environment for the development of a comprehensive and extensible landscape grammar system. One can thus imagine a

software application in which 3D landscape simulations are constructed iteratively by a grammar interpreter that operates directly on 3D landscape objects which have topological relationships and attribute data. In this 3D landscape grammar interpreter, the preconditions of rules would make use of advanced spatial reasoning capabilities and the consequents would have a substantial array of spatial operations available to them. Since no such integrated environment was found to exist, a modular approach was adopted in which different pieces of software were required, each offering different functionality.

Rather than develop a complete 3D grammar interpreter as described above, an alternative development platform is an existing 2D, object-oriented, GIS or CAD software environment that also has the capability for representing and processing knowledge through an internal programming language. Using this approach, 0D, 1D and 2D objects, that is points, lines, polygons and grid cells, store data relating to the third dimension as attributes (for example, elevation above sea level for height above ground level). The grammar rules then utilize 2D spatial functions and the grammar interpreter constructs the scene of 2D objects. The completed landscape scene can be exported to an external 3D file format by making use of the attribute data, and imported into appropriate 3D rendering software to visualize a grammar-generated scene in three dimensions. This approach takes advantage of knowledge already gained from previous research, regarding the semi-automated transfer of 2D GIS data to a 3D environment (Mayall, 1993). By externalizing the 3D visualization of the scene in this way, advantages are also gained in implementing the functionality for the geometric manipulations that are utilized in the rules. Many such geometric calculations are complex when the third dimension is included and thus by limiting the system to two dimensions efficiencies in development effort are gained.

Zero-dimensional to two-dimensional landscape objects with associated attribute data and grid-based landscape representations are handled typically by a GIS. At the time that development of the landscape grammar system was started, most GIS software tools were layer-based in which one layer of data represents a single type of object or a group of related types of objects. There were few options for object-oriented GIS software environments, that is platforms in which objects are not restricted to separate layers and can be related to generic conceptual definitions as is required by a landscape vocabulary. In addition, the languages available for commercial GIS software for application development were functionally limited to customizing the user interface and scripting existing commands, rather than providing broad enough functionality for facilitating the management of generic classes, rules, and the creation of new spatial functions as would be provided by general programming languages. This constraint has changed somewhat as GIS software vendors have come to recognize the need to move their systems to an object-oriented data model and the benefits of tying their software to standard programming languages. For example, in 2000, the Environmental Systems Research Institute

(ESRI) launched ArcInfo version 8.0 that introduced their “geodatabase” model in which user-defined object classes can be added to the traditional layer-based model. Thus, users may define classes of objects such as “land-parcel” and “easement” as well as requisite behaviours for those classes which are used as validation routines (ESRI, 1999, 2000). ArcInfo 8.0 also facilitates application development using the more general language Microsoft Visual Basic for Applications (VBA) in addition to its interpreted Arc Macro Language (AML).

In general, therefore, conventional GIS and CAD technologies lack the capabilities for abstract landscape class definitions and the expression of relationships between classes that are required for knowledge representation. By the same token, conventional knowledge representation and processing systems from AI generally do not manipulate spatial objects. Where this does occur, it is usually highly specialized to an application domain (such as mechanical engineering or construction details) and not readily adaptable to other uses. Expert systems that manipulate solid models typically do not make use of a large terrain surface which is crucial for the representation of landscape objects in three dimensions.

The two most popular programming languages for developing knowledge processing applications in AI have traditionally been Prolog and Lisp (Norvig, 1992; Jackson, 1999). Prolog is a goal-driven language that has logical statements and pattern-matching as an inherent part of its syntax and is therefore appropriate for many rule-based applications. It is based on theorem proving and is more commonly used for evaluating the truth of an assertion (analytical processing) rather than for generative purposes. This type of knowledge processing is primarily goal-driven, that is, it involves the search for a solution that satisfies certain constraints. Prolog can, however, also be used for data-driven knowledge-processing (the transformation of a set of data into a new set of data) which is closely associated with generative landscape grammars developed in this dissertation.

Lisp has been in use for approximately forty years (McCarthy, 1960) and much of its use has been in AI (Norvig, 1992). Because Lisp is so flexible and extensible, a family of Lisp languages and dialects have evolved, although recent implementations are centred around Common Lisp (CL), the American National Standards Institute (ANSI) standard for the Lisp programming language adopted in 1994 (ANSI, 1994). There are numerous enhanced implementations and dialects of Lisp and CL, including precursors to CL (MacLisp, Franz Lisp, Xlisp), freely available academic implementations of CL (EuLisp, CLISP, CMU Common Lisp, GNU Lisp, PowerLisp), commercial implementations of CL (Franz’s Allegro Common Lisp, Harlequin’s LispWorks, Digitool’s Macintosh Common Lisp), and new variants of Lisp (Scheme, Dylan). The popular drafting software, AutoCAD, offers a scripting language called AutoLISP, which is a limited derivative of Xlisp. The Common Lisp Object System (CLOS) used in this dissertation is an object-oriented extension to CL that facilitates the definition of classes and objects.

Widely used programming languages such as Microsoft's Visual Basic and various versions of C++ as well as standard C language could also be used to implement a grammar interpreter although they are less commonly applied to AI applications programming. Such languages are primarily designed for the procedural manipulation of data rather than abstracted information such as rules and symbols, and thus knowledge-based applications developed with these languages require more programming effort than Lisp or Prolog. Visual Basic is a high-level language that, while object-oriented, is designed for the simplified development of user interfaces and is not used in AI research. The C programming language offers great efficiency when compiled and has many function libraries available, but it is not object-oriented. The object-oriented adaptation of C is C++, which typically offers fast performance of tasks over large datasets and is consequently used in AI applications that are computationally intensive. In order to gain such efficiency, a significant amount of data and memory management is required thus making the language difficult to learn. While attractive for the production of commercial software applications, these features are not desirable for the rapid prototyping of a landscape grammar system such as that developed in this dissertation.

4.1.2 Benefits of Common Lisp and the Common Lisp Object System

The Common Lisp (CL) programming language and the Common Lisp Object System (CLOS) were used extensively in implementing the landscape grammar interpreter in this research. In the CLOS environment, object classes are templates for real objects, which are referred to as instances of a class. Classes are defined with slots for attribute data and methods that operate on instances of the class. Classes are arranged in a hierarchy in which subclasses may inherit slots and methods from their superclasses.

CL has significant benefits as a programming language for the development of a knowledge-based landscape grammar system. Most components of CL are represented as 'lists'. Lists are easier to use than arrays in other languages as they do not have a fixed length and thus require less management from the developer. Lists are also naturally suited for implementing the set theory of Chapter 3. CL also features 'dynamic typing' which frees a variable from restrictive data types such as integer, decimal or character. This feature of CL allows landscape classes and rules to be developed without having to declare a data type for each attribute definition. Hence, the use of lists and dynamic typing makes CL easy to use for system development.

CL's core functions for accessing and modifying lists are openly accessible and extensible to the programmer allowing other core functions to be created easily and fit seamlessly into CL. This flexibility is extended magnitudes further by CL's powerful macro facility which can be used to extend the basic language and create a new language suited to the purpose at hand. This allows landscape grammar rules

to be much more expressive than if restricted to the standard functions of a programming language. CL programs are also defined as lists and are therefore data, a feature that is unique to the CL language. This is highly relevant to the definition of landscape grammar rules because the precondition and consequent of a rule are stored as data, yet they are both executable functions (the precondition returning a true/false value and the consequent returning a modified scene). This feature also allows code in CL to be modified while a program is being executed. This can be utilized with landscape grammars by allowing a rule's consequent to change the behaviour of the interpreter program while it is running (for example, from serial to parallel processing) or to change the precondition or consequent of another rule.

As an object-oriented environment, the CLOS extension offers unique advantages that are not currently available in other languages such as C++. In addition to CLOS's sizeable and comprehensive set of functions for object-oriented programming, CLOS's automatic memory management feature (termed "garbage collection") is convenient because no special routines need to be written to free the computer memory used by deleted landscape objects. Because CLOS uses dynamic objects rather than static objects, landscape classes can be redefined while the program is running. In addition, the class of a landscape object in the scene can be changed by the consequent of a landscape rule during the interpretation process. CLOS classes also support multiple inheritance, that is, a subclass can have more than one superclass. This allows, for example, a Hibiscus class to be a subclass of a more general Hedge class as well as a subclass of abstract functional classes such as Boundary-Marker and Screening-Device.

Due to these and other benefits, CL and CLOS are often touted as particularly beneficial for the rapid prototyping of software applications, especially those in the representation and processing of knowledge (Steele, 1990; Wilensky, 1986; Keene, 1989; Lawless & Miller, 1991; Paepcke, 1993; Hasemer & Domingue, 1989; Norvig, 1992; Stefik, 1995). While there are performance limitations of CL and CLOS compared to low-level languages such as C++, the ease of use, flexibility and extensibility of the former are attractive features for the prototyping of a landscape grammar system. These benefits were bolstered by existing on-site experience in a Lisp environment in the Faculty of Environmental Studies at the University of Waterloo (Seebohm & Wallace, 1998).

A graphical environment in which geometric objects may be managed and displayed on screen is not part of the CL/CLOS specifications, nor was it found to be strongly integrated into the available CL programming environments. This is a drawback for the representation of landscapes with CL/CLOS. Of the PC-compatible CL implementations, Allegro Common Lisp for Windows by Franz Inc. of Berkeley, California, is the most widespread in the Lisp community and has a basic suite of functions for graphic displays named Common Graphics. Franz once supplied a Lisp-based solid modelling suite of functions named AllegroSolid, but this has since become unavailable and the price of the software was beyond the funding available for this dissertation.

4.1.3 The Selected Development Approach

Considering the above factors, the landscape grammar implementation created for this dissertation was undertaken in a Lisp programming environment, specifically Franz's Allegro CL/CLOS for Windows (version 3.0.2). The system was developed progressively by first writing a simple grammar interpreter that operated on a set of text strings. Capabilities were subsequently added to store spatial objects, define a hierarchy of classes for landscape objects and reason about and manipulate landscape objects using spatial functions. This culminated in the final implementation of the landscape grammar system (detailed in subsequent sections of this chapter) using rules to arrange spatial objects into landscape scenes.

With the initial grammar system, a vocabulary and scene comprised an unordered set of text strings such as "the lot is vacant", "the wall is 3 metres high", "the house faces the road". No classes or hierarchy were defined. For each rule, the precondition and consequent were each text strings, for example, IF "the house faces the road" THEN "erect a barrier near the road". In terms of the formalisms from Chapter 3, and recalling equation 3.20:

$$r = (\textit{precondition}, \textit{consequent}),$$

yields

$$r = (\text{"the house faces the road"}, \text{"erect a barrier near the road"}).$$

No functions or explicit reasoning capabilities were used in this version of the interpreter. The interpreter merely searched the scene for the existence of the text string from a rule's precondition. The first rule found to match the scene in this way was fired. The firing of the rule simply replaced the text string in the scene that matched the rule's precondition with the text string from the rule's consequent. The interpreter iterated until no more rules matched the scene.

This initial facility in which a list of 'facts' (text strings) are examined to induce further 'facts' is fairly typical of textbook exercises for expert systems (Hasemer & Domingue, 1989; Jackson, 1999). The representation of the landscape in this case is highly simplified, but the mechanism of processing rules to assemble a set of objects iteratively is present. In accordance with the landscape grammar theory of Chapter 3, the functionality of the prototype system was progressively enhanced to incorporate the use of variables in rules, the use of rule selection strategies, a comprehensive search for all matching sets of objects for a rule's precondition, selection strategies to refine the matching set of objects and parallel processing abilities to fire more than one rule with more than one set of matching objects.

With a text-based rule interpreter in place, the system was then extended to process geometric objects. Due to the absence of GIS components and libraries available for Lisp, a customized, object-oriented, Lisp-based GIS was programmed to facilitate this research. Object definitions were built for

the basic 0D, 1D and 2D shape-types presented in Chapter 3, that is, points, lines and polygons (both irregular and regular grids). Visualization capabilities were built using the Allegro Common Graphics Interface (CGI) that provides basic functions for displaying a window, setting its coordinate system, and drawing points and lines in colour. The programs were later enhanced to manage and display grid-based spatial data. Substantial development time was expended building a large library of spatial functions that could be used to test relations between objects and to modify and create objects (see Appendix A for a listing of functions developed for the dissertation). The existence of these functions allowed the creation of rules that manipulate spatial objects in various ways. Simple rules whose variables could be populated with 0D, 1D and 2D spatial objects were then developed and tested. For example, the following rule, using a variable name “?a”, states that if a spatial object from the scene is a Polygon, then the interpreter should create a Point object from its centroid:

IF (type(?a) = Polygon) THEN (make 'Point (centroid ?a)).

Following the addition of the GIS system, a class hierarchy was established using CLOS to allow user-defined classes of landscape objects and attribute definitions. The sophistication of the system was extended further by the incorporation of grid-based landscape data, the addition of many more spatial functions (increasing the power of the rules) and the development of macros to simplify the syntax used in rules, tools to aid in the inspection and maintenance of the grammar and functions to import and export geographic data to and from external files.

To facilitate the 3D visualization of the generated landscapes, a process was developed for exporting the object data to intermediate files for import into a 3D rendering software program, specifically Graphics Design System (GDS) software version 5.8 (Informatix, 2000). GDS is a well-established CAD software program that facilitates the creation of 3D solid models including terrain solids for landscapes. It is useful not only for its Solid Modeller module for creating 3D objects, but also for its Assembly Modeller module that assembles many 3D objects into a scene and its Scene View module that provides robust rendering algorithms over large numbers of objects for visualizing landscape scenes. The cost for 3D software was eliminated by using GDS, as a site licence was already available at the University of Waterloo's Faculty of Environmental Studies. In addition, previous research in which 3D landscapes were generated from GIS data and visualized in GDS was available for use in the landscape grammar system (Mayall, 1993; Mayall, Hall & Seebohm, 1994; Mayall & Hall, 1994; Mayall, Seebohm & Hall, 1994).

4.2 System Overview

A modular decomposition of the current Landscape Grammar System (LGS) is shown in Figure 4.2. As discussed above, the application is implemented in CL/CLOS with 3D functionality externalized to the GDS software. The core of LGS's structure contains the main theoretical components from Chapter 3, that is the landscape vocabulary, rules, working-scene of objects and an interpreter mechanism. The first three components are inputs to the interpreter which modifies the landscape scene. The LGS also contains a graphical viewer to visualize the scene and an import/export facility for saving the grammar, importing objects to the initial scene and exporting the final scene to GDS for 3D visualization. A significant part of the LGS is the object-oriented GIS that provides the facilities for managing landscape objects in a spatial database and a library of functions for analyzing and manipulating those objects. The spatial functions are part of the broader array of functions in LGS that are available to the user for composing rules.

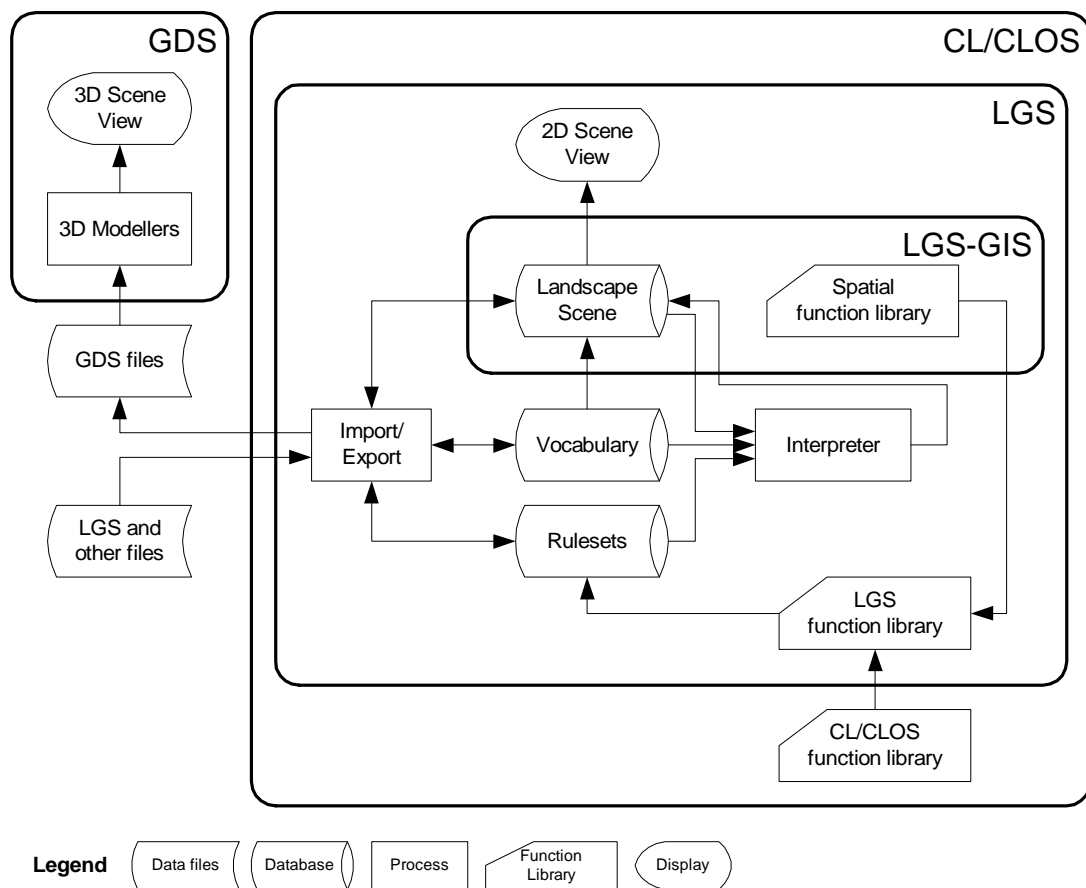


FIGURE 4.2 COMPONENTS OF THE LANDSCAPE GRAMMAR SYSTEM

The functional architecture of LGS is presented in Figure 4.3 and describes generally how the functionality of the LGS application is derived. The LGS function library is built upon the already large library of functions in Allegro CL/CLOS. The LGS system functions, written in CL, perform the internal operations of the system such as the interpretation procedures, internal data management and import/export routines. The GIS component is built from CLOS objects and all of the spatial functions that operate on those objects are defined using the CL language. In contrast to the system functions, the LGS library has numerous other functions that are designed for use in writing landscape grammar rules. Some of these functions extend the utility of the CL resource, while others reconfigure CL functions in a more appropriate syntax for the purposes of landscape grammars. At the top level of the diagram, the full range of CL/CLOS functions, LGS spatial functions, and other LGS functions become available for use in the composition of landscape grammar rules.

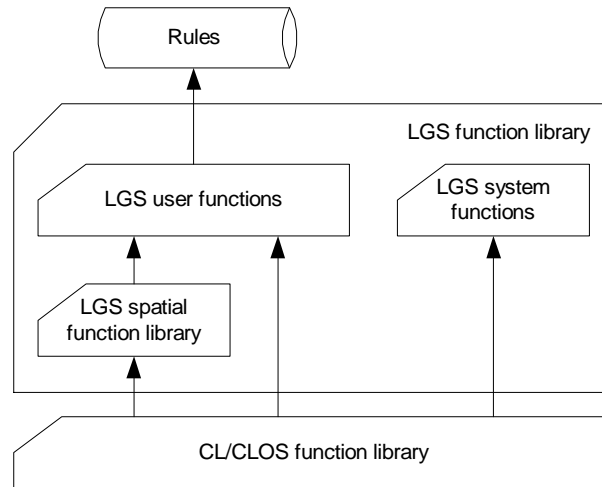


FIGURE 4.3 FUNCTIONAL ARCHITECTURE OF THE LANDSCAPE GRAMMAR SYSTEM

The current landscape grammar interpreter application runs inside the Allegro CL main window rather than as a separate application. Figure 4.4 shows the primary Allegro CL and LGS interface components. The graphics display window (Scene View) displays the scene objects. The display of classes of objects can be turned on or off from the Legend window to the left of the Scene View Window. The Legend is also used to change the display colors of each class. On the far left is a typical Lisp text window (in yellow, named Toploop by the Allegro CL software) that provides feedback messages from the LGS application to the user. The Toploop window is also the main interface through which CL and LGS commands may be input and executed by the user, such as to query the scene objects, the values of variables, or general debugging settings. The LGS menu commands are contained in the six rightmost pulldown menu options, as noted in the screen capture. The full menu layout of the

LGS Application Menu (Figure 4.5)

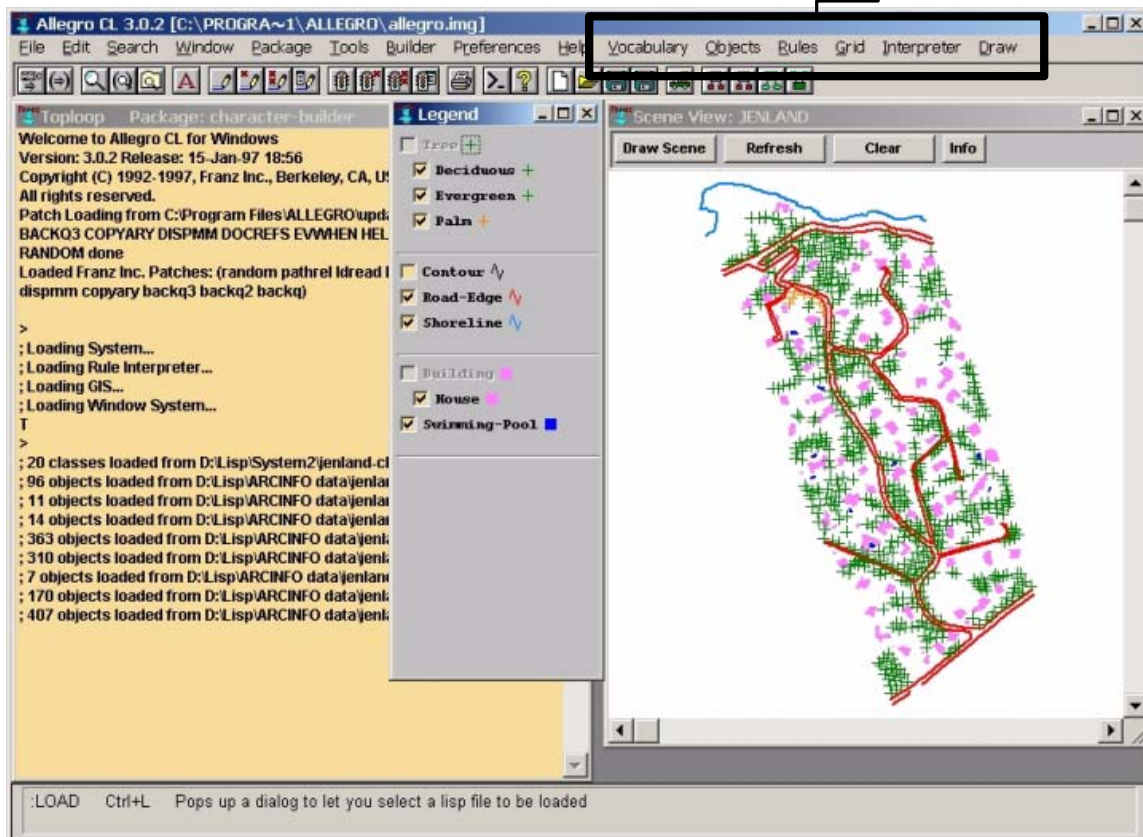


FIGURE 4.4 MAIN INTERFACE OF THE LANDSCAPE GRAMMAR SYSTEM

LGS is shown in Figure 4.5 and relates to the main components of the grammar system. The other menu items and toolbar buttons in Figure 4.4 relate to functions of the Allegro CL software. Despite its simple interface, the power of the LGS application is found within its capabilities for the storage, management and manipulation of vocabulary classes, scene objects and grammar rules that when applied form the ingredients of a landscape scene in the 2D view window.

In the following sections, the structure and functioning of the LGS is presented in detail. The basic discussion framework of Chapter 3, that is the vocabulary, scenes, rules and grammar interpretation process, is followed here to illustrate the connection between the theoretical concepts and the implemented system components.

4.3 The Landscape Class Hierarchy

The vocabulary of a landscape grammar is implemented as a hierarchy of classes relating to landscape objects. As noted earlier, LGS uses CLOS to define classes of objects. There are three levels

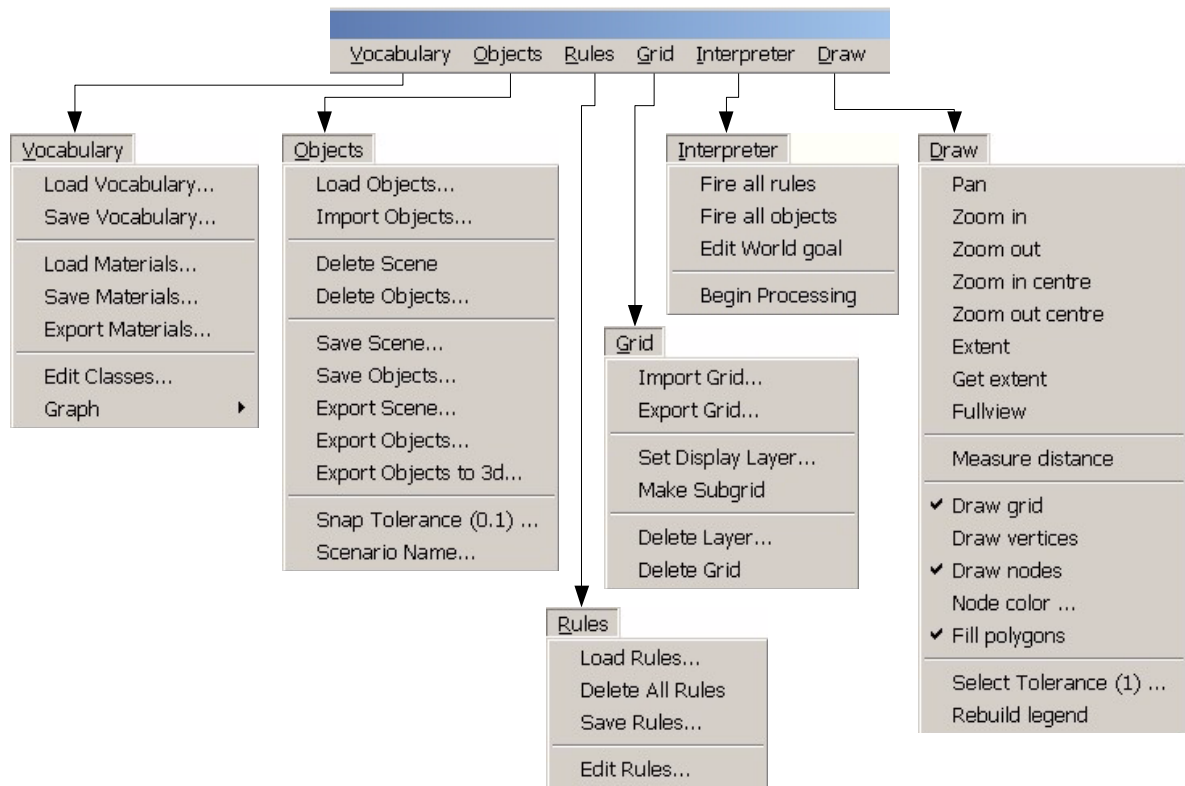


FIGURE 4.5 MENU STRUCTURE OF THE LANDSCAPE GRAMMAR SYSTEM

to the class hierarchy: the Lisp level, the spatial level and the landscape level (Figure 4.6). The Lisp level is internal and of no consequence to understanding the system operations but it should be mentioned that the top of LGS's object-class hierarchy is a subclass of the Lisp 'Standard-Object' and 't' classes. The t class has no superclasses and is a superclass of every Lisp class except itself. The Standard-Object class is a meta-object for most other classes within CL/CLOS (exceptions include Function, Number, Sequence, and Symbol classes). The spatial level contains classes of spatial primitives that are used to represent real-world objects in graphical form. The landscape level contains the user-defined classes representing types of objects found in the subject landscape. The landscape classes constitute the available vocabulary for the user's landscape grammar rules and define the types of objects that can be inserted into a scene. For the conventions of this dissertation, when referring to a specific class, or an instance of a specific class, the class-name is capitalized (e.g. Point, Polygon, Tree). Class-names that are being used as general terms in the text are lower-case. Thus, discussion referring to a 'point' is not necessarily read as an instance of the class Point - it may also be a Vertex, Tree, or an instance of any other subclass of Point.

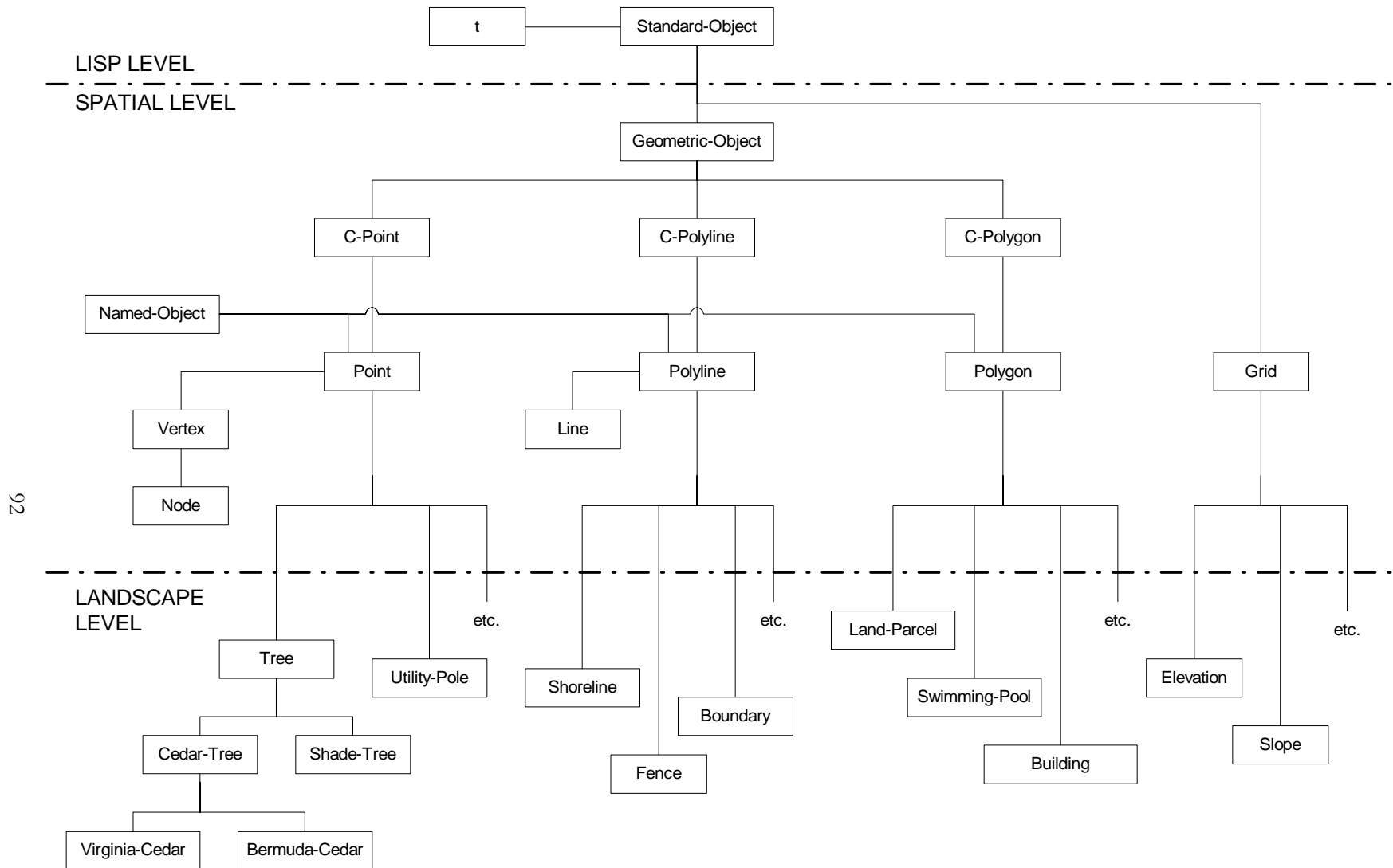


FIGURE 4.6 LANDSCAPE GRAMMAR CLASS HIERARCHY

The spatial level of classes in the LGS is shown in more detail in Figure 4.7, with the attributes, or slots, for each class also shown. At the top of the spatial level is the Geometric-Object class (the Grid class is discussed later). A Geometric-Object is a form of meta-object, that is, instances are not made directly of the Geometric-Object class but every subclass inherits properties from it. For example, every subclass of Geometric-Object inherits the Label slot which is a general-purpose attribute for marking an object for later referral. It is used in the same way that graphic symbols are used in the labelled shape of shape grammars (Section 2.7.2; Stiny, 1980a). The Elevation and Height slots are also inherited and contain numeric values for the elevation at ground level and height of the object above ground level respectively. When an instance is made of any subclass of Geometric-Object, initialization procedures are run for the new object, such as routines to number automatically an object or to check the geometric validity of an object such as ensuring that a line has more than one point. These initialization procedures are written for the Geometric-Object class and inherited by its subclasses.

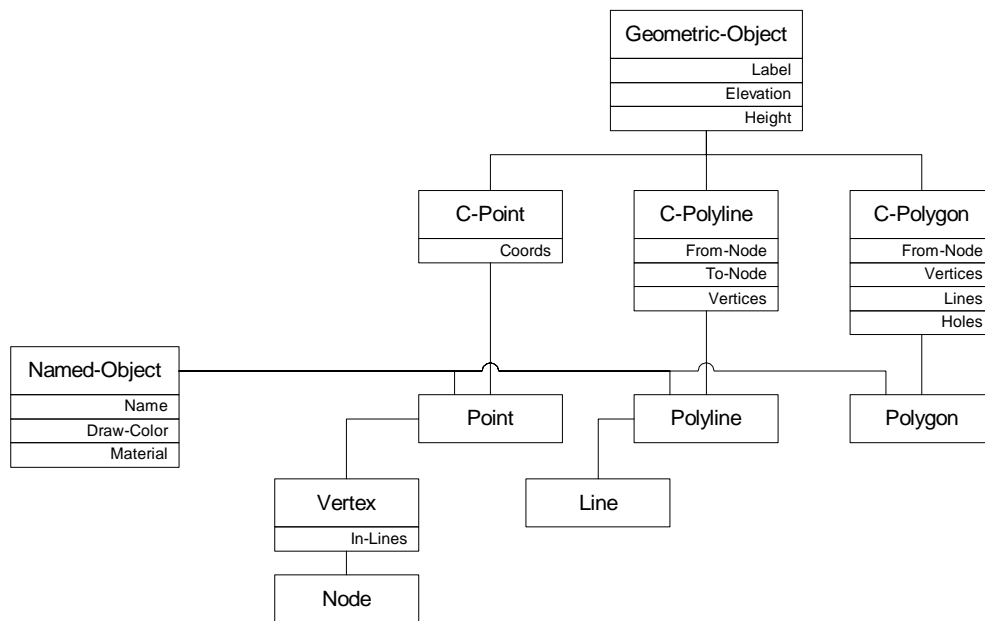


FIGURE 4.7 DETAILS OF THE SPATIAL LEVEL CLASSES

The Geometric-Object class has three main subclasses that form the primitive spatial classes on which the vector part of the GIS is based: C-Point, C-Polyline and C-Polygon. For the purposes of this discussion, these three classes are referred to collectively as Construction or C-Objects (although there is no class named ‘C-Object’). All geometric functions operate on C-Objects. The distinguishing feature of C-Objects is that they are not persistent, that is, they are not recorded on any lists and are not added to the scene database. C-Objects are temporary objects that can be passed from function to function for the purposes of geometric calculations. For example, a function calculating the midpoint of a line returns

a C-Point object, which can then be passed as an argument to another function for further calculations. When a function or set of functions has finished, the C-Objects are automatically deleted, or “garbage-collected”. This feature allows geometric objects to be utilized in the interpreter without cluttering the scene database unnecessarily. The Coords slot of the C-Point class stores a list of two numbers that represent the x and y coordinates of the point. The C-Polyline class has slots that store C-Points, specifically the first and last vertices (From-Node and To-Node slots) and those in between (Vertices slot). The C-Polyline is thus reducible to a list of C-Points. The C-Polygon class also has slots that contain C-Objects. The From-Node and Vertices slots are the same as for a C-Polyline (the To-Node for a closed polygon is the same as the From-Node). The Lines slot holds references to the C-Polylines that comprise the perimeter of the C-Polygon, and the Holes slot contains a list of C-Points for each hole polygon inside an instance of C-Polygon. These references to other objects as components of an object allow topological relationships to be managed between geometric objects.

The Point, Polyline and Polygon classes are direct subclasses of the C-Objects and the parent classes for all user-defined landscape classes. Since instances of the landscape classes are to be recorded in the working-scene they must be persistent unlike, the C-Objects. The Named-Object superclass ensures that instances of its subclasses are not “garbage-collected” by Lisp by assigning the object a name (stored in the Name slot) and recording it in a master list. The Named-Object class also contains a Draw-Colour slot that specifies a color to use when graphically displaying instances of a class in the LGS Scene View window, and a Material slot that refers to a colour definition for rendering later in the GDS 3D software. The Named-Object class also contains the initialization procedures for recording objects in the working-scene database. Therefore, all instances of any subclass of Named-Object are recorded in the working-scene.

There are three further geometric subclasses, namely Vertex, Node, and Line. Vertex is a subclass of Point and represents points that are part of a Polyline. Whereas the Vertices slots of C-Polyline and C-Polygon contain C-Points, the inherited Vertices slot of Polyline and Polygon contain instances of the Vertex class. The Vertex class contains an additional slot, In-Lines, that contains references to the Polylines in which a Vertex object is contained (this is convenient for some geometric functions). The Node class is a subclass of Vertex, representing the vertices on the end of a Polyline or Polygon (its end-points). Likewise, the From-Node and To-Node slots of Polyline and Polygon contain instances of the Node class. The Line class is a subclass of Polyline and is defined as a polyline with only two points (certain geometric functions will run more efficiently if a line is known to have only two points). Although the definition of a group of objects as a single object was included in Chapter 3, the functionality for this could not be implemented in the LGS.

The landscape level of the class hierarchy (Figure 4.6) contains all of the meaningful classes

defined by a user for their subject landscape. The landscape classes are subclasses of either Point, Polyline or Polygon. Users are free to determine the object classes that are relevant to the real world context of their landscape. Thus, the landscape classes might include: Signpost, Tree, Maple-Tree, Birch-Tree, Road-Edge, Sand-Shoreline, Rock-Shoreline, Building-Envelope, Bungalow, Semi-Detached, Cottage or Land-Parcel. Because these landscape classes are subclasses of Point, Polyline and Polygon and thus subclasses of C-Object, all geometric functions operate on landscape classes without any additional programming. In defining landscape classes that suit the needs of their application, users can arrange landscape classes in a hierarchy of superclasses and subclasses, as shown by the Tree classes in Figure 4.6. While the attribute data slots of the geometric classes are fixed, users are free to attach any attribute definitions to their landscape classes. For example, the format for a user's definition of a Tree landscape class may be given as follows:

```
(define-class Tree
  :superclasses (Point)
  :documentation "A point representing the centre of the base of a tree."
  :slots ((draw-colour green)
          (age :type number)
          species))
```

The format is also the syntax for a `define-class` macro function that is executed to create a landscape class. The terms in the above code prefixed with a colon are keywords which are always followed by a value. In order to define a landscape class, the user must specify a superclass of Point, Polyline, Polygon or another landscape class. The documentation string describes the class and is optional. Attribute/slot definitions are also optional and, when specified, may be assigned an optional default value as well as an optional data type. The default value will be stored in the slot for each instance of the class that is created, unless a specific value is given at the time of creation. If a data type for a slot is specified, CL validates any data assigned to that slot. In the above example, the landscape class, Tree, is a direct subclass of Point; is described as “A point representing the centre of the base of a tree”; will be displayed in green; has an attribute definition, age, with no default value but it must be a number; and an attribute definition, species, with no default-value or data type limitations. This program syntax for defining a Tree class relates directly to the equation for an object-type from Chapter 3:

$$OT = \{ \mathcal{Y}, st, AD_{\mathcal{Y}} \cup AD \}, \quad \text{Eqn 3.7}$$

where the list of superclasses (\mathcal{Y}) is provided as (Point), the shape-type (st) is inherited from the geometric class (Point in this case), and the attribute definitions ($AD_{\mathcal{Y}} \cup AD$) are represented as slot definitions for the Tree class (draw-colour, age, species) and any slots inherited from superclasses (Label, Elevation, Height, Coords, Name, Draw-Colour, and Material – ref. Figure 4.7).

When a slot has already been defined for a superclass (as with the Draw-Colour slot previously defined by the Named-Object class), the definition for the subclass supersedes the inherited slot definition including its default value and data type. Hence, in this example, a Tree object would be displayed in green regardless of the Draw-Colour for the Point class. The supersedence of slot definitions is a feature of CLOS and so holds true for any of the slot definitions anywhere in the hierarchy. This allows the attribute definitions of more refined landscape classes (that is, those lower in the hierarchy) to exhibit refined or anomalous default values that override the attributes of a more general and typical definition of such an object.

The class definitions for a landscape character can be written into a text file using the above format and a standard text editor and then loaded into the LGS via the Vocabulary menu shown in Figure 4.5. Because this process can become tedious when making continual improvements to a landscape grammar, a Class Builder tool was developed for creating and editing new landscape classes (Figure 4.8). A CL class grapher utility was modified and incorporated into LGS providing graphical representations of the landscape-class hierarchy for the user (Figure 4.9). A vocabulary of landscape class definitions can be exported from LGS to a text file for future use. While this export utility is essential for recalling the landscape grammar for later use, it has also future possibilities for allowing the transfer of a landscape vocabulary from one region to another (for example, when adopting the landscape style of another place) and the comparison of two or more vocabularies from different regions (this is not currently implemented in LGS).

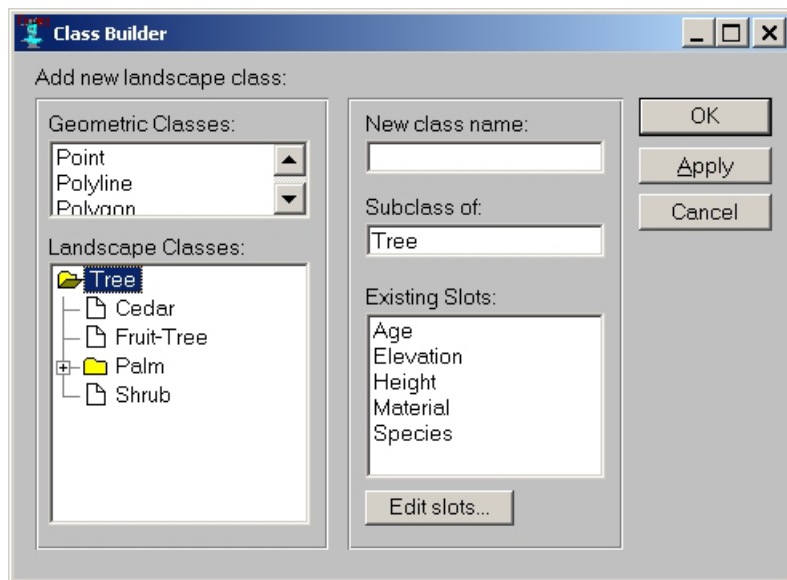


FIGURE 4.8 THE CLASS BUILDER TOOL

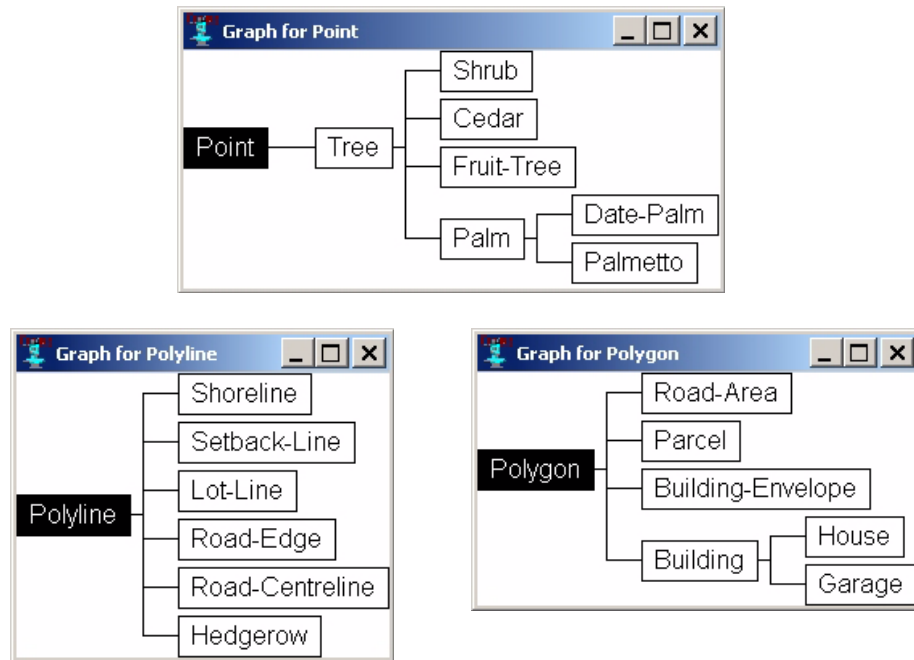


FIGURE 4.9 OUTPUT FROM THE CLASS GRAPHER TOOL FOR POINT, POLYLINE AND POLYGON CLASSES

4.4 Creating Instances of Landscape Classes

Once a landscape class has been defined, instances of the class may be created in LGS. When a landscape class is instantiated, an object is inserted in the working-scene which is the list of all landscape objects. The syntax for creating an instance of a landscape class is given as follows:

```
(new 'Tree '(544145 131105) :age 4 :species 'date-palm).
```

The LGS new function requires a class-name and a geometry. In this case, since Tree is a subclass of Point, the geometry is an (x, y) coordinate pair that defines the location of a specific tree in the scene coordinate space. This coordinate space can be imaginary or can relate to a real-world (geographic) coordinate system for the landscape. The remaining arguments are slot keywords and values in the form :slot-name slot-value. In this case, the Tree instance will have the default draw-colour as assigned in the class definition, as well as an age of 4 and the species “date-palm”. The creation of instances of point landscape classes is the simplest of the geometric classes. The following examples create polyline (Fence) and polygon (Parcel) objects:

```
(new 'Fence
  (list
    (new 'Node '(544450 131270))
    (new 'Vertex '(544375 131250))
```

```

      (new 'Vertex '(544300 131200))
      (new 'Vertex '(544175 131175))
      (new 'Node '(544125 131125)))
    :height 3
    :material 'chain-link)

(new 'Building
  (list
    (new 'Node '(544223 131164))
    (new 'Vertex '(544235 131105))
    (new 'Vertex '(544274 131113))
    (new 'Vertex '(544262 131172)))
  :status 'vacant)

```

The geometry of the polyline is a list of two nodes and a number of vertices. The geometry of the polygon has only one node because it is a closed area. The three examples of newly created landscape objects presented above are displayed in the LGS Scene View shown in Figure 4.10. The `new` command syntax for a landscape object relates directly to the equation 3.13 for an object from Chapter 3:

$$obj = \{OT, G, A\}, \quad \text{Eqn 3.13}$$

in which the object-type (OT) is provided as the class-name, the geometry (G) is provided as a list of coordinates or Nodes and Vertices, and the set of attribute values (A) is provided as the slot values which correspond to the slot definitions for the new object's class and superclasses.

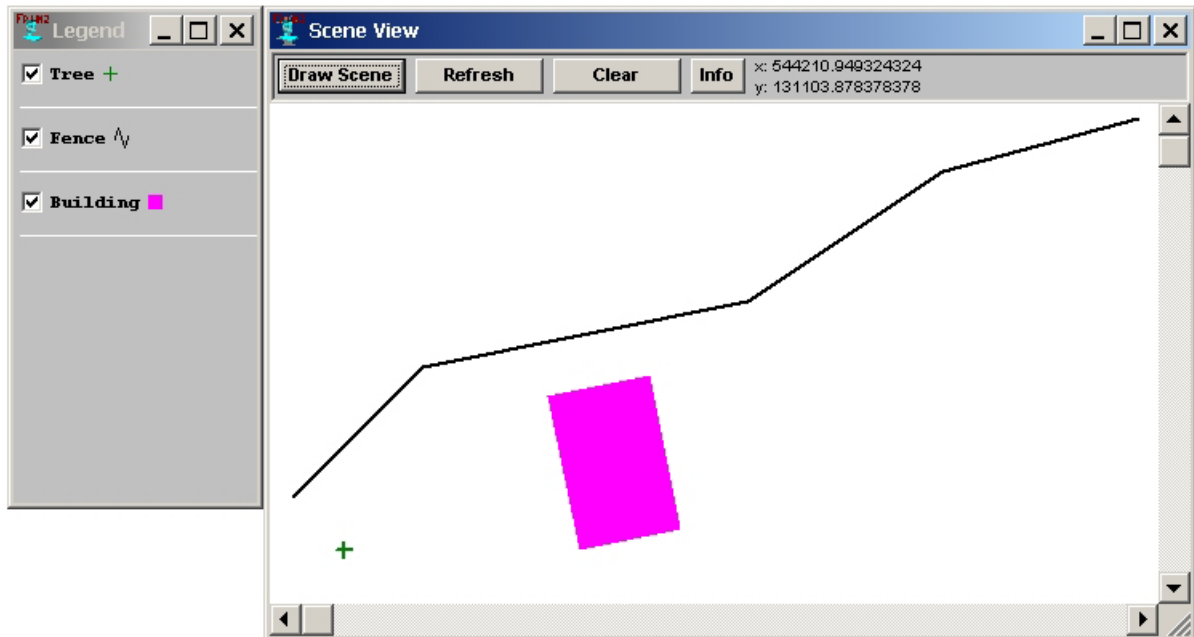


FIGURE 4.10 EXAMPLES OF INSTANCES OF LANDSCAPE CLASSES

In automated landscape generation, it is often more convenient to convert an existing object into an instance of another class than to supply the geometric coordinates explicitly. This is particularly the case when replacing an object of an abstract class (such as “Barrier”) with a physical object (such as “Fence”). In addition, recall that some geometric functions return C-Objects (for example, the midpoint and centroid functions each return a C-Point), which are not themselves instantiated in the scene. These C-Objects can, however, be subsequently converted to an instance of a landscape class, provided that the landscape class is a subclass of the class of the returned C-Object. The LGS `make` function creates a new object of a specified class from another object, retaining its geometry and any slot data that are applicable to the new class. The following syntax creates an instance of the Signpost class from a C-Point returned by a function that calculates the endpoint of a polyline:

```
(make 'Signpost (endpoint ?a))
```

where `?a` is a variable that contains a polyline object.

The landscape objects contained in a given scene may be imported into and exported from the LGS application (see the Objects menu in Figure 4.5). The objects of a scene can be saved to and recalled from an LGS text file format that contains a series of new statements such as the examples given above. Alternatively, objects can be imported into an initial scene from other spatial data formats. The primary external file format is ESRI’s open (non-proprietary) “shapefile” standard which is comprised of at least three separate files, `.shp`, `.dbf`, and `.shx`, which contain respectively geometric, attribute and index data. While the shapefile provides the geometry and attribute data for an LGS object, it does not explicitly provide the landscape class (as required by Eqn 3.13). When importing objects from a shapefile to the working-scene, the user is asked to specify either a landscape class for all objects in the shapefile or the name of a data field in the shapefile that contains the landscape class-name for each imported object. The shapefile format also does not contain topological information, but this is created by LGS as each object is imported from the shapefile. Landscape data may also be imported to LGS from ESRI’s ArcInfo Generate file format (the technical description of which is available in ArcInfo Help files). This text-file format contains geometric data but no attribute data and therefore objects imported from a Generate file must all become instances of a single landscape class in LGS. The geometry of scene objects may also be exported from LGS to the Generate file format for subsequent use in ArcInfo.

The LGS Scene View is a graphical display module for viewing LGS landscape objects in two dimensions. The provision of this tool within the system avoids the need to export a generated landscape scene to an external file for viewing in 2D GIS or CAD software. However, as outlined in Figure 4.2, a process was developed for converting the scene of 2D objects to a 3D representation using GDS software and file formats. The details of this export and conversion process are presented in a later

section following discussion of how the LGS interpreter generates a scene of 2D objects.

The LGS application contains a large library of more than 400 functions written to perform GIS operations (Appendix A). Some functions return numeric or string values, while others return geometric objects (C-Objects) or grid cells. Other functions can be used as predicate (true/false) tests or may return objects and values that are then arguments for predicates. Examples of the operations performed by these functions include: moving, copying, scaling, rotation, buffering, distance calculations, perimeter and area calculations, line intersections, point-in-polygon tests, point-on-line tests, adjacency tests, proximity tests, line generalization, path-finding and calculation of the minimum bounding rectangle and convex hull. The geometric functions are useful as predicate tests in a landscape grammar rule's precondition, for example returning whether an object is within a certain distance of another or returning the objects that fall inside another object. The functions are also useful for creating new landscape objects in the actions of a rule's consequent. Typically, the geometric functions return a C-Object that can be converted into an instance of a landscape class.

The working-scene of objects is stored in a master table that is accessible to the user and to all rules at all times. The scene objects are visualized through a graphical display window (the Scene View) and controlled via the earlier noted Legend window in which classes are listed hierarchically within the legend, classes can be turned on and off and interactively assigned draw-colours (Figure 4.4, Figure 4.10).

The LGS application was extended to represent raster data structures. Grids warranted a separate class at the level of Geometric-Object (Figure 4.6) because they are essentially different from points, lines and polygons, as is demonstrated by the functional separation of raster and vector data modules in GIS software. The class hierarchy needed is not utilized well with grids. It was first attempted to define a Grid class with a slot containing instances of a Grid-Cell class, but given the typically large number of cells in a grid-based landscape representation (often tens of thousands), this data model took much longer for the LGS application to process than the typical representation of a grid as a 2D array of values.

The Grid class from the spatial level of the class hierarchy in Figure 4.6 has several slot definitions, as shown in Figure 4.11. The Num-Rows and Num-Cols slots contain the number of rows and columns in the Grid object; the Cell-Size slot contains the width of one (square) cell, or the 'resolution', of the Grid; the Origin slot contains an (x, y) pair defining the lower-left corner of the Grid; the No-Data-Value slot contains a value which represents the absence of data in a cell; the Cells slot contains a 2D array of values; the Unique-Values slot contains a list of the unique-values in the Cells slot; and the Colour-Ramp slot contains a colour value for each of the values in the Unique-Values slot for display purposes. Because the Cells slot contains one set of values, each instance of the Grid class effectively becomes a separate 'layer' of cell-values representing a different continuous spatial variable.

Using the LGS grid data model, it is possible to accommodate different sizes, origins and resolutions of Grid objects. This feature is useful in cases where the raster landscape data available for a region are maintained at different resolutions. However, since the grid data used in this dissertation have identical Grid parameters, the use of grid data of various parameters has not been tested. Thus, while there may be grid data for elevation and slope for example, it is assumed that the two data sets are registered to the same coordinate space and cell resolution.

Grid
Num-Rows
Num-Cols
Cell-Size
Origin
No-Data-Value
Cells
Unique-Values
Colour-Ramp

FIGURE 4.11 DETAILS OF THE GRID CLASS

Data for continuous spatial variables may be imported to and exported from LGS using ESRI's ArcInfo GRID ASCII format (the technical description of which is available in ArcInfo Help files). A GRID ASCII file is a text file containing header information relating to the first five slots of the Grid class in Figure 4.11, followed by a list of data values for one spatial variable. Figure 4.12 shows the display of a grid of land elevation values (white = highest and black = lowest elevations) in conjunction with a set of vector objects. Grid objects form part of the working-scene and their data are available for reference in the precondition of a landscape grammatical rule. It is also possible for the consequent of a rule, when fired by the interpreter, to alter the values of a set of cells. This allows rules that, for example, excavate, fill or grade the land surface to be used.

This section has shown, by creating instances of landscape classes, how the working-scene in a landscape of interest is populated with objects. The management of these objects is handled in LGS by a custom-programmed object-oriented GIS that accommodates both vector and raster data. This system contains many of the features of a conventional GIS, including visualization of geometric objects with attribute data, many geometric analysis and editing functions and the management of raster data. The creation of a vocabulary of landscape classes and a scene of landscape objects are thus accommodated by the LGS application. The third component of a landscape grammar (Eqn 3.1) is a knowledge-base of landscape rules relating to a regional landscape character. The definition and handling of rules in LGS is presented in the following section.

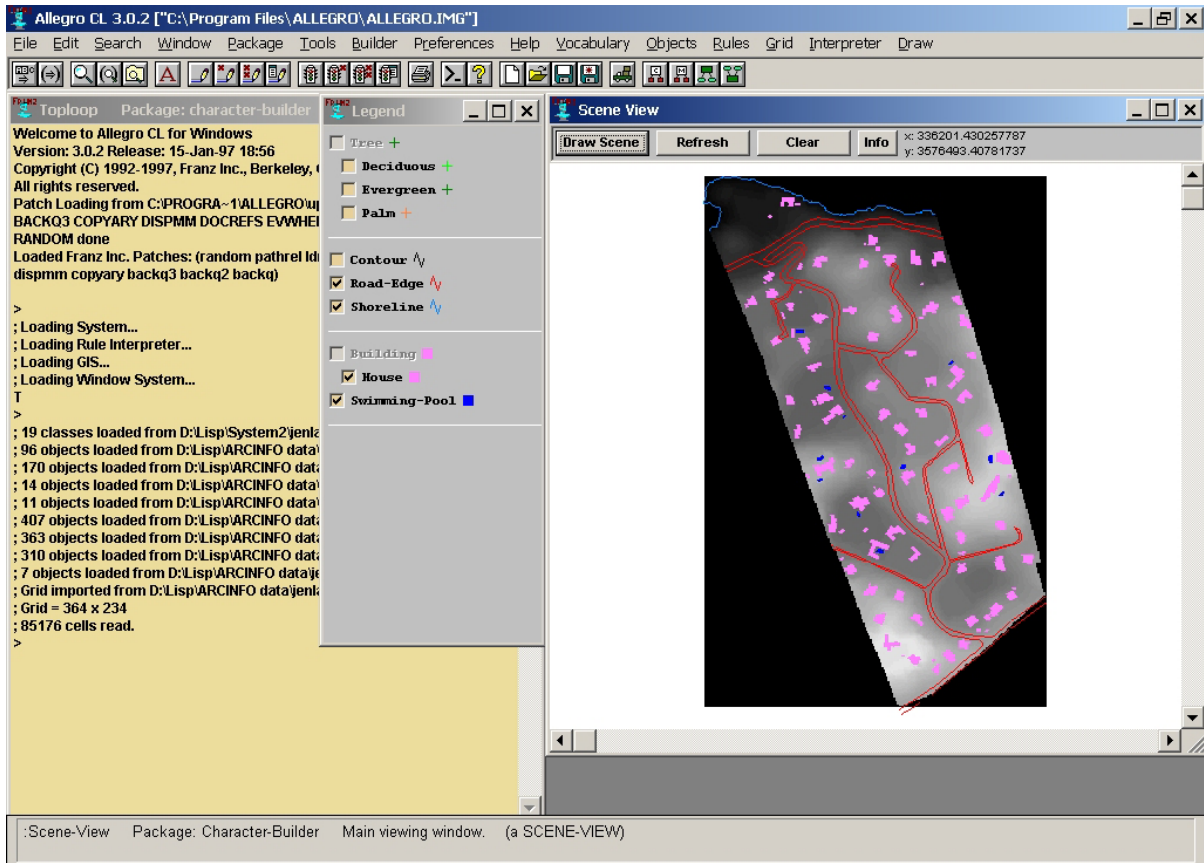


FIGURE 4.12 THE INCORPORATION OF RASTER DATA INTO A SCENE

4.5 Rules and Rulesets

Landscape grammar rules specify the typical relationships between landscape classes. This is achieved by the use of if-then instructions for the construction of a landscape scene. The rules are used by the LGS interpreter (described in the next section) to populate iteratively the working-scene with instances of the landscape classes. Like landscape objects, grammar rules are stored in a rule database in LGS as CLOS objects. Rules are grouped in Rulesets which are also CLOS objects. The Ruleset to which a Rule belongs is specified when the Rule object is created. If the Ruleset does not exist, it is created automatically so there is no special syntax required for the user to create Rulesets. The default 'World' Ruleset contains all other Rulesets and any rules not assigned to a specific Ruleset.

Similar to landscape classes and objects, rules are created using syntax for an LGS define-rule function, such as:

```

(define-rule
  :name build-boundary-wall
  :rulesets garden-ruleset
  :explanation "If a parcel boundary is not already built upon, then build a
              boundary wall"
  :if ((is-a ?a 'Polyline)
       (is-a ?b 'Parcel)
       (is-part-of ?a ?b)
       (is-not-labelled ?a 'built))
  :then ((make 'Wall ?a)
         (set-label ?a 'built)))

```

The above example defines a rule named ‘build-boundary-wall’ that simply states “if a parcel boundary is not already built upon, then build a boundary wall there”. The keywords in the `define-rule` syntax relate to slot definitions in the Rule class. Every LGS Rule object has a name slot containing a user-assigned name for the rule, as well as a Rulesets slot containing the name(s) of the Ruleset(s) to which the rule belongs. Rules can be assigned to more than one Ruleset allowing for re-use of rules in different contexts. The explanation slot provides the intent of the rule and can supply information to a user later reviewing the sequence of rules fired by the interpreter in generating a scene. The two primary slots of a rule object, IF and THEN, refer respectively to the rule’s precondition and consequent. The IF slot contains a list of statements each of which returns true or false and the THEN slot is a list of statements that can be executed in sequence if the rule is selected for firing during the interpretation process. Hence, the above rule may be explained in more rigorous terms as “if an instance of the Polyline class (or by inheritance, any of its subclasses) is part of the perimeter of an instance of the Parcel class and the Polyline instance does not have a ‘built’ label in its Label slot, then convert that line to an instance of the Wall class and insert a ‘built’ label in its Label slot”.

The syntax used in the IF and THEN slots deserves further elaboration. Note first of all that the IF and THEN slots relate directly to the definitions of a precondition and a consequent in Eqns 3.18 and 3.19 of Chapter 3:

$$\text{IF: } \textit{precondition} = \{ \textit{predicate}_1, \textit{predicate}_2, \dots, \textit{predicate}_n \}$$

$$\text{THEN: } \textit{consequent} = (\textit{action}_1, \textit{action}_2, \dots, \textit{action}_n)$$

where each of the items listed in the IF slot is a *predicate* and each of the items listed in the THEN slot is an *action*. Furthermore, each of the predicates listed in the IF slot are made up of a relation and a set of arguments as specified in Eqn 3.21:

$$\textit{predicate} = \phi(\textit{arg}_1, \textit{arg}_2, \dots, \textit{arg}_n)$$

where, from the example, ?a and Polyline are arguments to the relation is-a. The predicates are combined with the logical operator, AND, which is inserted at the start of the IF slot as part of the

initialization procedures for Rule objects in LGS. A landscape grammar predicate makes use of variables that refer to an object or a value. Variables are introduced in the preconditions of LGS Rule objects using the syntax, `?variable-name`. In the above example, the `build-boundary-wall` rule uses two variables, `?a` and `?b`. The rule's consequent can use the variables introduced in the precondition as well as initiate its own variables using a standard Lisp `let` statement.

The relations and other functions used in the LGS rules (Appendix A) are expressed explicitly in Lisp syntax. Currently, users of the system must be familiar with CL although the CL language was extended in LGS with many customized commands to make the writing of landscape rules easier. For example, compare the syntax used here to the simple rules of shape grammars (Figure 2.5). The latter, in their pure form, contain only labelled shapes in the rule's precondition and consequent. The simple, visual rule format is easier to understand but it would take many of these rules to accommodate various configurations of landscapes. A more complex rule syntax, such as that used here in the LGS application, is perhaps more difficult to understand but is more powerful because more situations and actions can be expressed succinctly. By increasing the expressiveness of the grammar rules, the grammar as a whole can be simplified by reducing the number and interdependence of rules.

Like the landscape class definitions, rule definitions can be typed directly into a text file using a text editor and can be loaded from or saved to these files using the LGS Rules menu shown in Figure 4.5. Alternatively, the Rule Editor tool may be used to create and edit rules (Figure 4.13). Using this tool, a user can create and delete Rulesets or select an existing Ruleset to view a list of its constituent rules (top-left window of the figure). Rules can be added to and deleted from a Ruleset or a single rule can be selected from the list to display the code for the rule's precondition and consequent. The code may then be edited and saved back to the Rule database.

The LGS 'Rule' class has subclasses of rules that can have specialized behaviours. The Rule subclasses are not specified by the user but are used internally for efficient rule processing. Currently, there are only two rule subclasses. A Starting-Rule is any rule that contains, in its consequent, a function call that starts a nested iteration of the interpretation process. The One-Variable-Rule is a rule that only uses one variable in its precondition. This type of rule can be handled much more efficiently than multiple-variable rules when searching for scene objects that match the precondition. Both of these rule subclasses are detected automatically by the LGS application and do not need to be specified by the user. When a Rule object is created (such as when loading rule definitions from a file), the rule initialization procedures search the rule's code to count the variable-names in the precondition (thus identifying that the rule is an instance of a One-Variable-Rule) or to find a call to the LGS `interpret` function in the rule's consequent. Other classes of rules beyond Starting-Rules and One-Variable-Rules could be implemented, in a similar manner to the approach to shape grammars by Seeböhm and Wallace (1998).

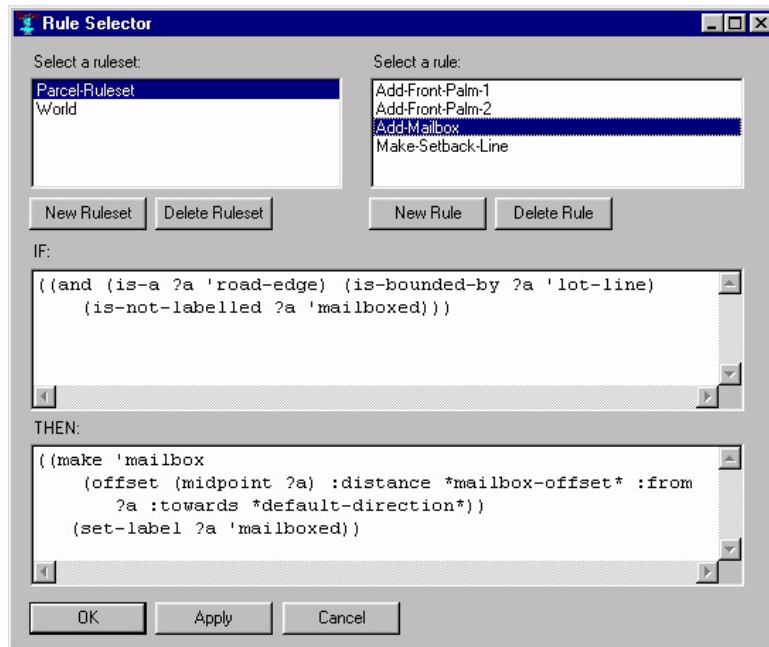


FIGURE 4.13 THE RULE EDITOR TOOL

In their case, various rule-types are used including: “envelope rules” to generate a geometric constraint envelope, “alternate rules” offering alternatives when a rule is not able to be executed and “root rules”, which are the equivalent of Starting-Rules.

In addition to rules and classes, a landscape grammar may also have parameters that store values that are globally applicable across all scenes and are accessible to all rules. Examples of global parameters might include a minimum lot size, climate condition or time of year. Parameters are defined using the LGS `define-parameter` function such as:

```
(define-parameter maximum-building-height 20
  "The maximum height of a building allowed by the planning code")
```

Parameter definitions are loaded from and saved to the same text files as the rule definitions. If parameter values are treated as constants (that is their values are never changed) then references to the parameter-name in a rule can be replaced with the parameter value. The advantage of parameters over constants is that the value of a parameter can be changed ‘on the fly’ during the interpretation process by an action in a rule’s consequent. The color-definitions of materials are also defined using a custom LGS function:

```
(define-material sand :H 148 :L 76 :S 54)
```

where the H, L, and S arguments are respectively the hue, light and saturation elements of a color-definition. While this implementation only makes use of materials as definitions of colour, it would be simple enough to incorporate texture and density characteristics for materials if desirable.

In summary, the implementation of landscape grammar rules in the LGS application makes use of CL syntax, user-defined parameters and LGS functions for spatial and non-spatial calculations and operations on spatial objects. The rules state the conditions under which particular operations should be carried out in order to generate a landscape scene that conforms to the landscape character defined by the grammar. The LGS rule interpreter mechanism is able to carry out the task of applying these rules to the working-scene.

4.6 The Rule Interpreter

As already noted, the process of generating scenes by grammar interpretation requires a vocabulary, rule-base and initial scene (Eqn 3.1). The preceding sections of this Chapter have stated how to define in the LGS application a vocabulary of landscape object classes and sets of grammar rules. Section 4.4 demonstrated how to instantiate the landscape classes creating a visualizable scene of landscape objects. Once an initial scene is available, the process of interpreting the grammar to modify the initial scene can be initiated. The first step in this process is to load a vocabulary of class definitions into LGS from text files. Rules are then loaded, also from text files. Objects are then either loaded from LGS command-files or imported from ESRI shapefiles into the working-scene. Grid data can also be imported into the scene at this time. With a vocabulary, rulesets and a working-scene assembled, the process of grammar interpretation begins.

The interpreter is started from the LGS Interpreter menu as shown in Figure 4.5. The menu contains two switches that modify how the interpreter processes the landscape grammar. When the “Fire all rules” switch is selected, the interpreter processes rules in parallel rather than in serial, that is all matching rules are fired instead of just one. Likewise, when the “Fire all objects” switch is selected, the interpreter processes a rule’s matching objects in parallel, applying each rule to all of its matching objects. These two modes were summarized earlier in Table 2.1 in relation to other types of spatial grammars. In addition to the switches for serial/parallel processing, the interpreter function accepts a ‘goal’ from the user. This feature was implemented as an enhancement to the landscape grammar theory discussed in Chapter 3. The common operation of the interpreter is to continue selecting and firing rules until there is no rule left whose precondition matches the working-scene. The goal is a user-defined predicate function that provides the interpreter with an additional stopping condition. If the goal function evaluates to true, the interpreter stops processing rules regardless of whether there are any that match the working-scene.

In Figure 4.14, part of the system overview from Figure 4.2 is expanded to illustrate the implemented interpretation process in further detail. The flow of program execution here relates to the theoretical process shown in Figure 3.4 of Chapter 3. The interpreter module starts by using the World ruleset and all landscape objects. The Pattern Matcher is a set of thirteen functions that follow the process of identifying matching rules and sets of matching objects for those rules that was defined previously in Figure 3.5 and Figure 3.6. The Pattern Matcher takes each rule in turn and binds objects from the scene database to the variables contained in the code of the rule's precondition. With each bound set of objects and variables (termed an 'environment'), the entire code of the precondition is evaluated returning a true/false value. If true, the set of matching objects is recorded in a 'possible-environments' slot attached to the rule (which is a CLOS object). The Pattern Matcher continues searching for more environments for the rule. Each rule in the ruleset is treated in this manner and any rule that has one or more environment is recorded in a list of matching rules.

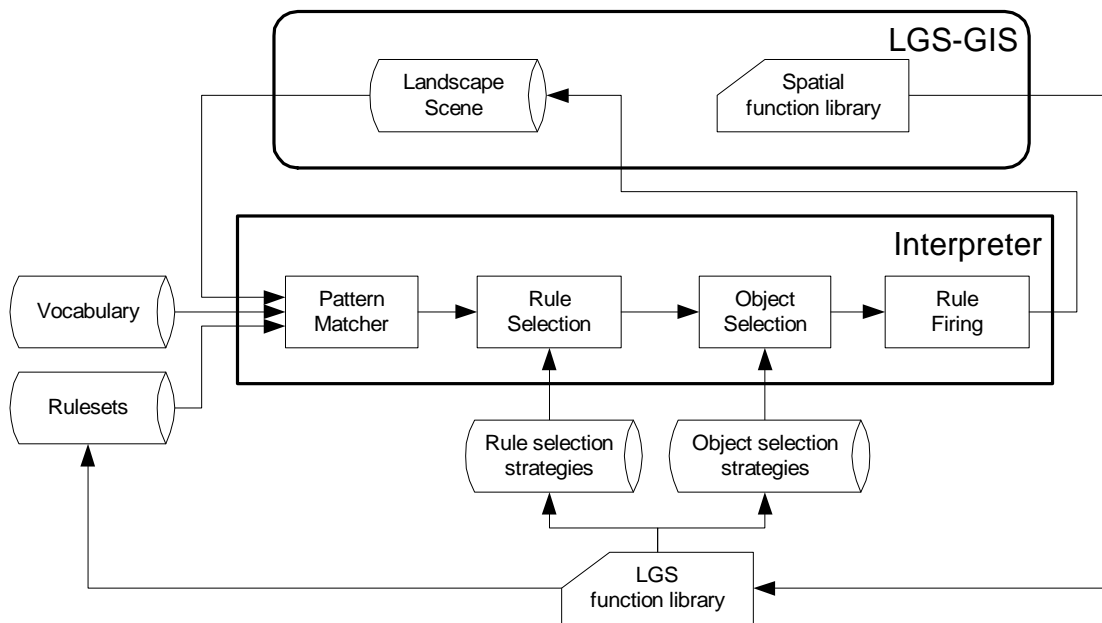


FIGURE 4.14 INTERPRETATION MODULE OF THE LGS

On completion of the Pattern Matcher's search, the interpreter is halted if the list of matching rules is empty. Otherwise, the list of matching rules is then passed to a Rule Selection module that uses rule selection strategies to reduce the list of rules for firing. The set of rule selection strategies is an ordered list of functions written using the available CL and LGS function libraries. The simple strategies identified previously in Section 3.2 are predefined in the LGS application. These include strategies to select: those rules with matching sets of objects that have not been fired already; those rules that have

been fired least recently; or the first (or a random) rule of the set. The other types of strategies discussed in Section 3.2, namely attribute-based, function-based, or spatially-based strategies, are more specific to the nature of the landscape character represented by the grammar. They must therefore be written by the user using CL and LGS functions. Each strategy is a function that accepts as its argument a list of rules and returns a reduced version of that list. The following is a functional definition for the simple strategy of selecting one rule randomly using CL:

```
(defun random-rule (list-of-rules)
  (list (random-element list-of-rules))).
```

This is an illustration of the syntax for the definition of any Lisp function. Briefly, it calls the `defun` function with the name of a new function, followed by a list of its arguments and then by the main body of Lisp code that, in this case, returns a list of one element from the list of rules. While it is not intended to provide a comprehensive summary in Lisp programming here, it should be stated that this is the vehicle through which rule selection strategies are defined. For parallel processing, these strategies are skipped altogether as all rules will be fired. In serial rule processing, the list of matching rules is reduced to a list of one rule. The result of the rule selection module is passed to the Object Selection module which is implemented in exactly the same way as rule selection, except that the list contains objects rather than rules. The object selection strategies are defined as Lisp functions making use of LGS functions where desired. Again, in the parallel rule application mode, the object selection strategies are ignored while in serial application the list of possible-environments is reduced to one set of landscape objects.

The Rule Firing module replaces the variable-names in the rule's consequent with landscape objects according to the selected environment and then executes the consequent's program code. This execution results in changes to the objects in working-scene database. The environment that was used in the rule application is recorded in a 'previously-fired-envts' slot of the rule object, as well as in a history of the rules fired. The LGS application allows the visualization of the modified working-scene by refreshing the display in the Scene View. Before the next iteration of the interpreter is started, LGS checks to determine whether the fired rule has signalled a halt (by setting the global variable `*system-halt*` to true) or if a goal has been achieved (e.g. a maximum number of objects has been reached in the working-scene). If neither has occurred then the Rule Stepper dialog is displayed stating information about the current state of the interpreter (Figure 4.15). The information includes the name(s) of the rule(s) fired, the number of iterations the interpreter has performed, the level of interpretation which is greater than zero if a Starting-Rule has initiated a nested session of the interpreter, and the number of objects present in the working-scene. The Rule Stepper provides the opportunity for explicit user control as the user can pause after each iteration, let LGS continue processing rules uninterrupted until no more

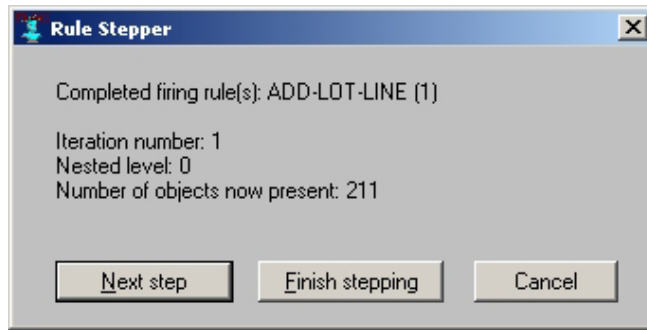


FIGURE 4.15 THE LGS RULE STEPPER DIALOG

rules are applicable (or a goal is reached or a halt signalled), or cancel the interpretation process completely.

The preceding paragraphs describe one iteration of the interpreter. After a rule is fired, the vocabulary, rules and modified working-scene are supplied to the interpreter module and the process of pattern matching, rule and object selection begins again. On completion of the interpretation process, the working-scene has reached its final state. A report is issued in the Lisp text window listing the number of rules fired and the number of objects in the final working-scene for each landscape class. A history of the selected rules and objects that were fired at each iteration is also written to a log file for later examination. The final 2D working-scene is displayed in the Scene View window and may be inspected using standard windows-based zoom and pan functions in addition to the display controls in the Legend window. The entire scene or particular classes of objects may be exported to LGS text files or an external file format for the GDS software for 3D visualization. After a full interpretation, the working-scene is deleted, the initial site objects re-loaded and the interpretation process begun again in order to generate another scene that displays the character defined in the landscape grammar.

4.7 A Scene Generation Example

This section illustrates the use of the landscape grammar system by providing an example of a simple scene generation. A vocabulary of nine class definitions was loaded into LGS, as well as seven parameters and eight rules. The definitions of the classes, parameters and rules in the CL syntax are as follows:

```
;;; -----
;;; Landscape Classes
;;;
(define-class Mailbox
  :superclasses (Point))
```



```

(define-class Tree
  :superclasses (Point)
  :documentation "A point representing the centre of the base of a tree."
  :slots ((draw-color green))
         (age :type number)
         species))

(define-class Palm
  :superclasses (Tree))

(define-class Shrub
  :superclasses (Tree))

(define-class Setback-Line
  :superclasses (Polyline))

(define-class Lot-Line
  :documentation "A line representing an edge of a land parcel."
  :superclasses (Polyline)
  :slots ((setback 10)
         (draw-color orange)))

(define-class Road-Line
  :superclasses (Polyline))

(define-class Road-Edge
  :documentation "A line representing the edge of a road."
  :superclasses (Road-Line)
  :slots ((setback 20)
         (draw-color red)))

(define-class Parcel
  :documentation "A polygon representing the closed boundary of a land parcel."
  :superclasses (Polygon)
  :slots ((draw-color #S(RED 255 GREEN 236 BLUE 187))))

;;; -----
;;; Parameters
;;;

(define-parameter *lot-frontage* 40)
(define-parameter *lot-depth* 60)
(define-parameter *front-palm-offset* 4)
(define-parameter *mailbox-offset* 2)
(define-parameter *default-direction* :right)
(define-parameter *road-setback* 20)
(define-parameter *lot-line-setback* 10)

;;; -----
;;; The World Ruleset
;;;

(define-rule
  :NAME add-lot-line
  :RULESETS world
  :IF ((is-a ?a 'Road-Edge)
       (is-not-labelled ?a 'lot-closed)
       (> (object-length ?a) *lot-frontage*))
  :THEN ((make 'Lot-Line
              (make-perpendicular
               :to ?a
               :of-length *lot-depth*
               :at-the-point (distance-along-line *lot-frontage* ?a)
               :towards *default-direction*)))
  :EXPLANATION "If a road-edge is longer than the lot-frontage, then add a lot line
               perpendicular to the road-edge, splitting it at the lot-frontage
               distance.")

```

```

(define-rule
  :NAME close-lot-lines
  :RULESETS world
  :IF ((is-a ?a 'Road-Edge)
        (is-bounded-by ?a 'Lot-Line)
        (is-not-labelled ?a 'lot-closed))
  :THEN ((let* ((side-lot-lines (bounding-lines ?a 'Lot-Line))
                (closing-lot-line
                 (new 'Lot-Line
                     (list
                      (endpoint :of (first side-lot-lines) :away-from ?a)
                      (endpoint :of (second side-lot-lines) :away-from ?a))))))
          (new 'Parcel
              (path-of-vertices
               :along-the-lines (list* ?a closing-lot-line side-lot-lines)
               :lines (list* ?a closing-lot-line side-lot-lines)))
          (set-label ?a 'lot-closed)))
  :EXPLANATION "If a section of a road-edge is enclosed by two lot-lines and a parcel
                has not yet been created there, add a third lot-line that joins the
                other two and make a parcel.")

(define-rule
  :NAME start-parcel-ruleset
  :RULESETS world
  :IF ((is-a ?a 'Parcel)
        (is-not-labelled ?a 'setbacked))
  :THEN ((interpreter
          :rules (rules (get-ruleset 'parcel-ruleset))
          :objects (cons ?a (lines ?a))
          (set-label ?a 'setbacked)
          (dolist (line (lines ?a))
            (remove-label line 'setbacked)))
          :EXPLANATION "If a parcel has not had the setback lines laid out, then start a
                        nested interpretation with the Parcel ruleset and a new working-scene
                        that is limited to the parcel and the lines of its perimeter.")

;;; -----
;;; The Parcel-Ruleset
;;;

(define-rule
  :NAME add-mailbox
  :RULESETS parcel-ruleset
  :IF ((is-a ?a 'Road-Edge)
        (is-bounded-by ?a 'Lot-Line)
        (is-not-labelled ?a 'mailboxed))
  :THEN ((make 'Mailbox (offset (midpoint ?a)
                                :distance *mailbox-offset*
                                :from ?a
                                :towards *default-direction*))
          (set-label ?a 'mailboxed))
  :EXPLANATION "If a road-edge is bounded by two lot-lines, then it is part of a lot.
                Add a shrub at the midpoint of the lot-front, just inside the lot.")

(define-rule
  :NAME add-hedgerow
  :RULESETS parcel-ruleset
  :IF ((is-a ?a 'Lot-Line)
        (not (is-bounded-by ?a 'Lot-Line))
        (is-not-labelled ?a 'hedgerow-done))
  :THEN ((make 'shrub (insert-points-along-line ?a 8 2))
          (set-label ?a 'hedgerow-done))
  :EXPLANATION "If a lot-line is not the back lot-line, insert shrubs along it.")

```

```

(define-rule
  :NAME add-front-palm-1
  :RULESETS parcel-ruleset
  :IF ((is-a ?a 'Road-Edge)
       (is-bounded-by ?a 'Lot-Line)
       (is-not-labelled ?a 'treed))
  :THEN ((make 'Palm (offset (distance-along-line 6 ?a)
                            :distance *front-palm-offset*
                            :from ?a
                            :towards *default-direction*)))
  :EXPLANATION "If a road-edge is bounded by two lot-lines, then it is part of a lot.
                Add a palm-tree inside the lot near the beginning of the lot-front.")

(define-rule
  :NAME add-front-palm-2
  :RULESETS parcel-ruleset
  :IF ((is-a ?a 'Road-Edge)
       (is-bounded-by ?a 'Lot-Line)
       (is-not-labelled ?a 'treed))
  :THEN ((make 'Palm (offset (distance-along-line (- (object-length ?a) 6) ?a)
                            :distance *front-palm-offset*
                            :from ?a
                            :towards *default-direction*)))
        (set-label ?a 'treed))
  :EXPLANATION "If a road-edge is bounded by two lot-lines, then it is part of a lot.
                Add a palm-tree inside the lot near the beginning of the lot-front.
                Once this palm-tree is placed, the road-edge is fully treed.")

(define-rule
  :NAME make-setback-line
  :RULESETS parcel-ruleset
  :IF ((is-a ?a 'Polyline)
       (is-not-a ?a 'Setback-Line)
       (is-not-labelled ?a 'setbacked)
       (is-a ?b 'Parcel))
  :THEN ((make 'Setback-Line
              (buffer ?a :distance (setback ?a) :towards (centroid ?b)))
        (set-label ?a 'setbacked)))
  :EXPLANATION "For all lines of the parcel perimeter, insert a setback-line whose
                position is offset towards the centre of the parcel by the setback
                distance associated with that line.")

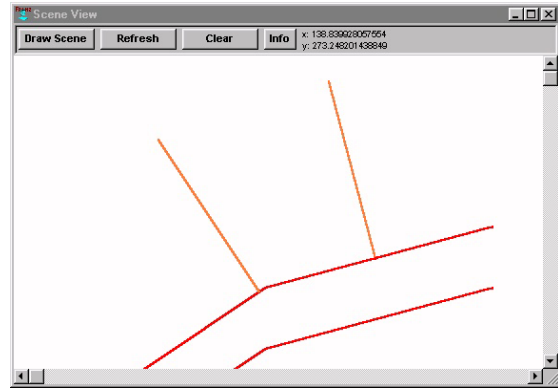
```

Three of the rules belong to the default ‘World’ Ruleset, while the other five belong to a ‘Parcel-Rules’ Ruleset that was designed to operate within a single parcel. Figure 4.16 shows the application of the above rules creating a grammatically generated scene. The initial input to the working-scene in this case comprises two edges of a road (red) that are made instances of the Road-Edge class. The interpreter begins in the World ruleset with the full working-scene (all objects, that is the two Road-Edges). On the first iteration of the interpreter, the add-lot-line rule is the only rule of the three in the World ruleset that matches objects in the working-scene. The add-lot-line rule is thus selected and fired. The result of the firing of the rule, as shown in Figure 4.16 (i), is the extension of a lot-line (orange) perpendicular to the road-edge to a distance specified by the *lot-depth* parameter. On the second iteration, in Figure 4.16 (ii), the same rule is selected and fired again inserting a second lot-line into the working-scene at the distance from the first lot-line specified by the *lot-frontage* parameter.

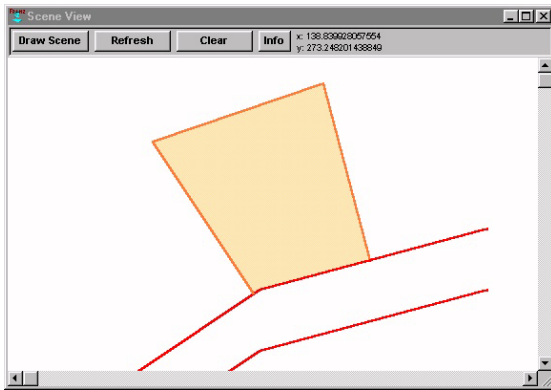
On the third iteration, both the add-lot-line rule and the close-lot-lines rule are applicable to the scene. A rule selection strategy that selects rules that have been fired least recently



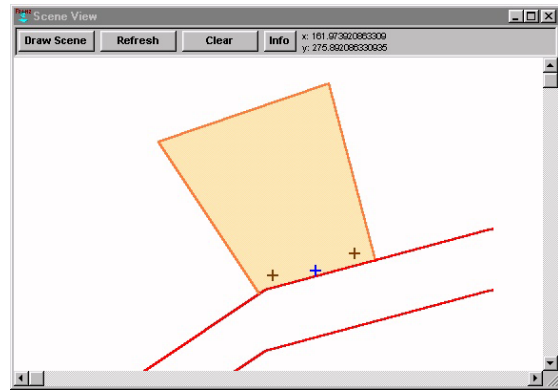
(i) Step 1 (add-lot-line)



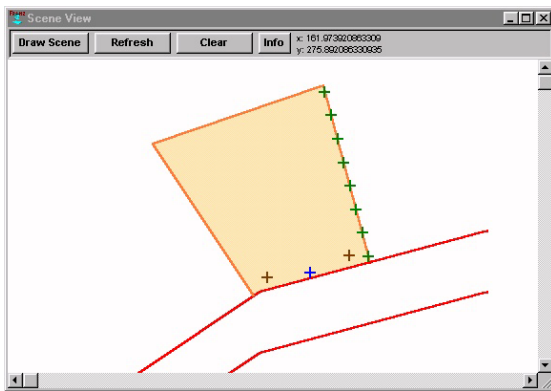
(ii) Step 2 (add-lot-line)



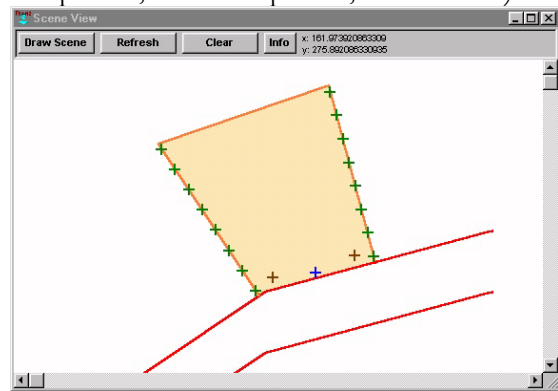
(iii) Step 3 (close-lot-lines)



(iv) Steps 4, 5, 6, 7 (start-parcel-ruleset, add-front-palm-1, add-front-palm-2, add-mailbox)

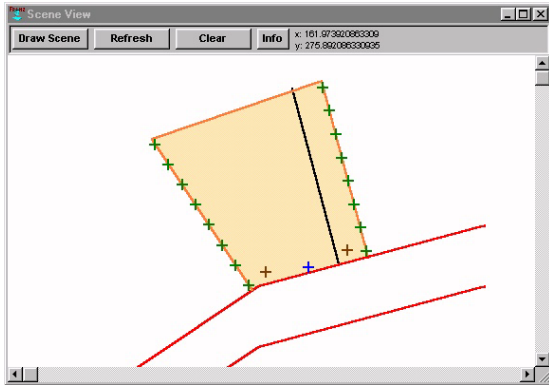


(v) Step 8 (add-hedgerow)

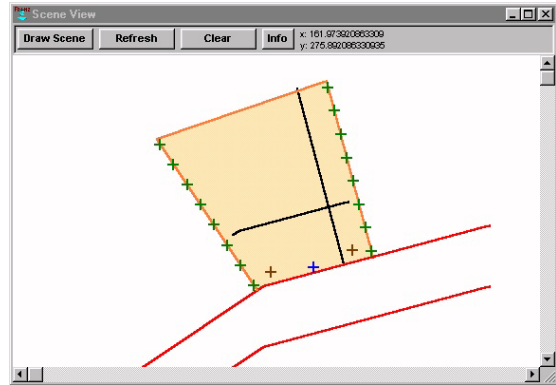


(vi) Step 9 (add-hedgerow)

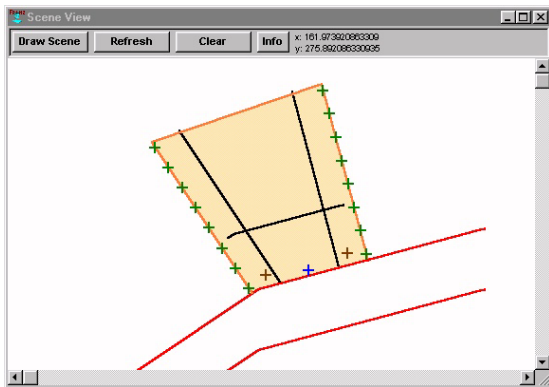
FIGURE 4.16 ITERATIVE MODIFICATION OF A SCENE



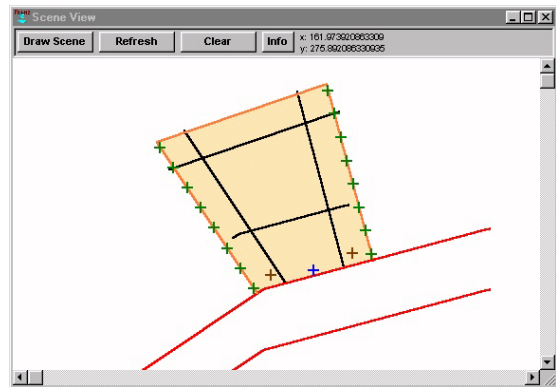
(vii) Step 10 (make-setback-line)



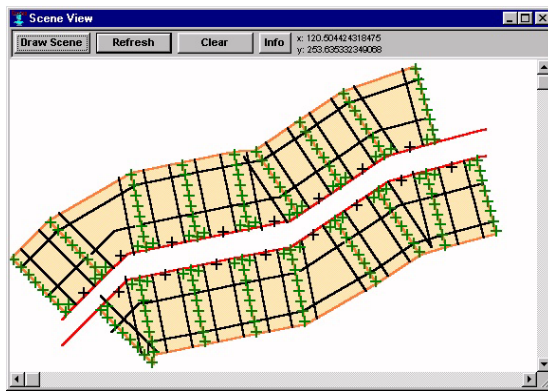
(viii) Step 11 (make-setback-line)



(ix) Step 12 (make-setback-line)



(x) Step 13 (make-setback-line)



(xi) Step 193

FIGURE 4.16 CONTINUED

reduces this list of matching rules to the `close-lot-lines` rule. The firing of this rule (Figure 4.16 (iii)) closes the lot by adding a Lot-Line between the end-nodes of the other two Lot-Lines, and creates a new Parcel polygon object (yellow).

The fourth iteration fires the `start-parcel-ruleset` rule that begins a nested interpreter process. The nested iterations are geographically confined to the created Parcel and the lines that make up its perimeter, and are functionally confined to the rules of the Parcel-Ruleset. Figure 4.16 (iv) shows the result of the next three iterations, firing the `add-front-palm-1`, `add-front-palm-2`, and `add-mailbox` rules of the Parcel-Ruleset. These rules insert Palm-Tree point objects (brown) in the two front corners of the lot and a Mailbox point object near the centre of the lot frontage. The `add-hedgerow` rule is fired twice (Figure 4.16 (v) and (vi)) inserting a series of Shrub point objects (green) along the each Lot-Line that is joined to the Road-Edge.

The next figures illustrate the calculation of ‘setback’ lines that define the envelope within which a development may occur on a parcel. In Figure 4.16 (vii), (ix) and (x), a Setback-Line object is inserted at a distance of 10 metres from a Lot-Line object, while in Figure 4.16 (viii) the distance from the Road-Edge object is 20 metres. In each of the four iterations, the same rule, `make-setback-line`, is being fired but uses different distance values that are stored in Setback slots defined in the Lot-Line and Road-Edge vocabulary classes.

After Step 13, the nested interpreter process finds no more applicable rules and exits from the nested interpretation. Control returns back to the consequent of the `start-parcel-ruleset` rule in the outer process using the full scene of objects and the World Ruleset. The remainder of that consequent labels the Parcel object as having been “setbacked” and removes labels that are no longer needed from the lines in the Parcel’s perimeter. While this example has focused on one parcel for illustrative purposes, the interpreter actually continues processing objects and rules until it finds no more rules that are applicable to the working-scene. The final result after 193 iterations is shown in Figure 4.16 (xi). The construction of this scene took approximately 15 seconds to generate, without using the Rule Stepper, on a PIII computer with an 800 MHz processor.

This example demonstrates the generation of a simple landscape scene from a landscape grammar in the LGS application. This landscape grammar implementation is also demonstrations of the ability of the system to emulate other types of spatial grammars, as presented in Chapter 2.

4.8 Export of the Scene for 3D Visualization

While the LGS Scene View provides a simple 2D graphic display capability, the visual impact of the scene is ideally represented for planning and other real world purposes in 3D to enable the viewer to

visualize the landscape as it is (if the scene has been created to develop an existing landscape) or as it could be. In response to this, the scene contents can be transferred to GDS software for rendering and visualization in 3D. As noted, the procedures used in this dissertation make use of the GDS Solid Modeller module for creating 3D objects as well as the Scene Viewing System (SVS) module to render a scene of 3D objects and allow the observer to move around the simulation model. The landscape scene generated by the LGS is transferred to GDS using the GDS THINGS file format (Informatix, 1993). Earlier research performed similar operations by transferring ESRI data to GDS and creating 3D objects by executing programs written in GDS-BASIC language (Mayall, 1993). The new approach used in this dissertation is to generate directly from the CL/LGS environment either 3D GDS data (THINGS) files or to provide command scripts that, when run in GDS, perform the operations to create, store and visualize 3D objects (Figure 4.17). This approach was used in order to contain the procedures to the LGS application as much as possible and because GDS-BASIC is very restrictive compared to the Lisp language.

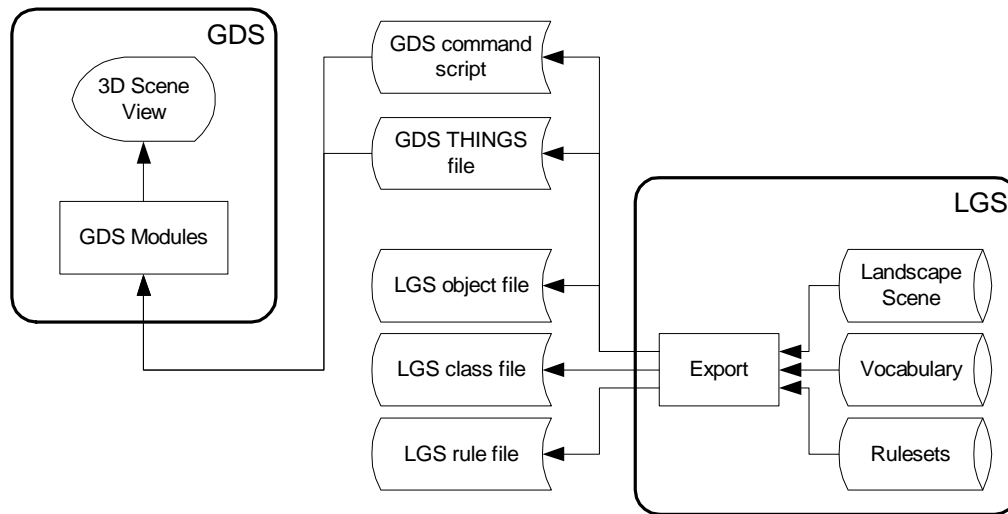


FIGURE 4.17 EXPORT OF LANDSCAPE GRAMMAR AND SCENE

The export module of the LGS application uses the 2D coordinate data of the landscape objects in conjunction with their attribute data to provide a 3D interpretation of those objects in the form of a GDS THINGS file or command script (Table 4.2). Point objects are interpreted as insertion points for the base of individual 3D representations of those objects. The Elevation slot-value of the Point object determines the base z-coordinate of the 3D object. The 3D representation of the object must pre-exist in a GDS data file (“codex”) and LGS exports a GDS command script that recalls the 3D object from the codex and inserts it at the appropriate locations. The object may be a 3D solid (that is enclosing a volume of space) or it may be a vertical planar object that faces the observer. Because the former can

add significant complexity to the 3D rendering routines in GDS, the latter is desirable when a large number of points are included in the scene. Polyline objects are interpreted as vertical planar 3D objects, such as a fence. The Elevation slot-value of each vertex of the polyline determines the z-coordinate of that vertex for the 3D object, while the Height slot-value determines the distance over which the polyline is extruded in the z-axis. Alternatively, a polyline can be used to represent a path over which a vertical shape is ‘swept’ to create a 3D solid object (Table 4.2).

Polygon objects are interpreted as solid objects as when a square is extruded into a cube. The Elevation slot-value of the polygon is used as the z-coordinate for each vertex in the base of the 3D solid, while the Height slot-value determines the distance over which the polygon is extruded in the z-axis. Using this basic attribute information pertaining to the third dimension, the LGS Polyline and Polygon objects can be exported to a GDS THINGS file. Since the THINGS file format carries no attribute data or object-naming conventions, each class of landscape object is exported to its own file. Hence, the converted 3D scene in GDS is a visualization of the scene geometry with no attribute information.

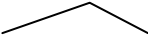
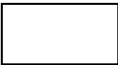


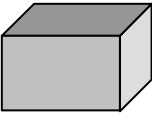
	Point	Polyline	Polygon
2D objects	+		
3D objects			
	Inserted solid	Extruded planes	Extruded solid

TABLE 4.2 RELATION OF 2D TO 3D OBJECTS

The GDS software uses ‘materials’ to define the colour of each face of an object. GDS materials can also be used to define the surface reflectance and mass of an object but only colour was used in this dissertation. All landscape objects in LGS have a Material slot that can be used to store the name of a GDS material that is later assigned to that object in the 3D renderings (the name of a GDS material is a maximum of 8 characters in length). Each material name is associated with a color definition (as a Hue-Light-Saturation value) and all of these associations are stored in a central table. As each LGS object is exported, the name of the material of each face is saved with the object in the THINGS file. In addition, the color definitions of materials can be exported to a GDS command script to recreate those colors in the GDS software.

While Table 4.2 illustrates the translation of the geometric object classes to 3D objects, the terrain on which those objects reside must also be represented in the 3D simulation. The LGS application represents terrain as a grid of elevation values, but in a solid modelling environment such as GDS, terrain is represented as a very large solid object with an undulating surface of triangular facets. Such a solid is usually generated from points, each with a z-coordinate value, or a triangulated irregular network (TIN) in which each node of the network has a z-coordinate value.

In the use of LGS in this dissertation, a TIN is converted to a lattice or grid of values using ESRI's ArcInfo GIS software (Figure 4.18). The grid is exported to an ArcInfo GENERATE file and imported into LGS as a Grid object in the scene. The grid values are used to calculate the elevation values of the geometric landscape objects which are subsequently exported to GDS as described above. To create the terrain solid in GDS, the original ArcInfo TIN is recalled, exported to the GENERATE (NET) format, and then converted to the THINGS file format using a conversion program written for the LGS. Although unimplemented here, an alternate method would be to develop an LGS export function that creates a terrain solid in a THINGS file by allowing the centre of each cell in the LGS elevation grid to be a point in a lattice, thus creating a network of triangular facets that connect to each other at the centre points of the cells.

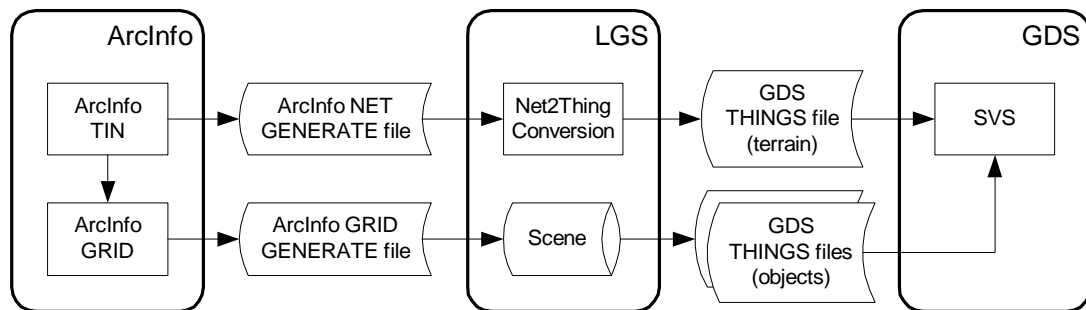


FIGURE 4.18 USE OF ELEVATION DATA

The conversion of LGS landscape objects to 3D GDS objects can be enhanced by scripting sequences of commands from the GDS Solid Modeller module. These commands create new solid objects by using Boolean operations (union, intersection or difference, for example) on two or more 3D solids. If the LGS application were implemented within a 3D environment, that is the landscape objects were represented as 3D solid objects in LGS, then the landscape grammar rules could use Boolean solid operations directly in the scene. In this implementation, where the 3D component is externalized, the solid operations must be scripted for later execution in GDS. Because these operations are not tightly integrated with LGS, they cannot be flexibly applied to individual objects by individual rules. The

Boolean solid operations are therefore enclosed as export routines. Two types of scripts are generated by the LGS. One script cuts the shape of a polygon object into the terrain surface. The primary application of this technique is for representing roads in the landscape. The polygonal landscape objects are extruded above the maximum elevation of the landscape and the terrain solid is removed from that extrusion. The resulting solid is then lowered by 10cm and removed from the terrain solid. The final result is the terrain solid with indentations in the locations of the LGS polygonal objects. When the LGS objects are road polygons, the roads appear to be cut into the surface of the terrain. Mayall (1993) describes this process in more detail with illustrations.

The other script is used to create sloping faces on the top of an extruded polygon. The intention of this routine is to create hipped roofs on top of solid objects that represent buildings. When polygonal objects representing building footprints are exported from LGS to a THINGS file, a roofless building is simply extruded from zero elevation to a vertical distance of the value of its Elevation slot plus the value of its Height slot. In addition, if an LGS object has a 'Roof-Material' slot then the slot's value is used as the GDS material (colour) of the top face of the extruded polygon. In order to make a hipped roof for a polygonal object, a command script is generated that first extrudes the polygon an additional vertical distance that is specified in the object's 'Roof-Height' slot. Then, for each edge of the polygon (or wall of a building if the polygon represents a building footprint), a solid object is defined that cuts away a portion of the extruded polygon such that a sloping face is left. The shape of this 'cutting' object is such that it resembles a 'winged' cuboid that lies nearly adjacent to the wall.

Figure 4.19 provides elevations and a perspective view of a cutting object, as well as a section and plan to show the relation of the cutters to a 3D building object. Angle a is half of the horizontal angle between two edges of the polygon's perimeter. Angle b is calculated from the desired slope of the roof as specified in the object's 'Roof-Slope' slot in LGS. The shape of each 3D cutting object is

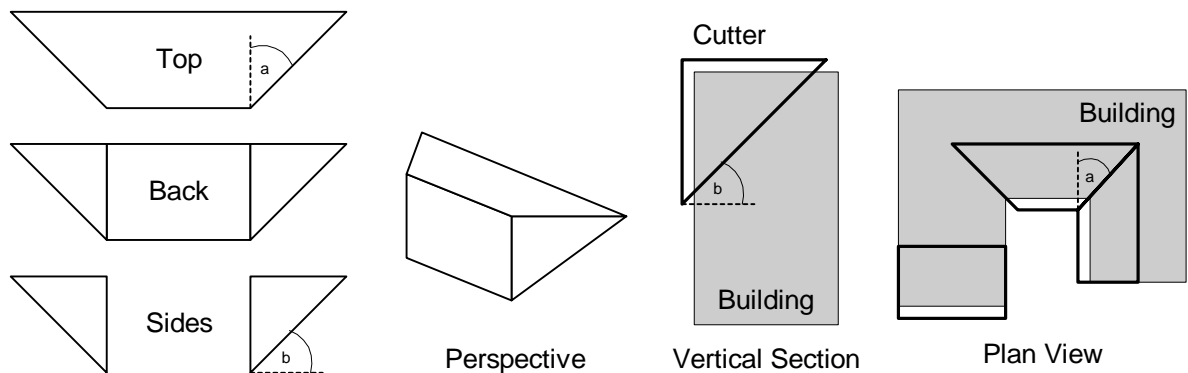


FIGURE 4.19 ROOF CUTTING OBJECTS

calculated in LGS and exported to a THINGS file. A GDS command script is also exported that uses the GDS Solid Modeller to remove each cutter from the extruded building solid. Thus, while the Boolean solid operations are detached from the LGS application, they are influenced by the Roof-Height and Roof-Slope parameters in the exportation of the 3D cutting objects and the values of these parameters can be set according to landscape grammar rules.

In summary, the 2D scene of landscape objects that is grammatically generated by the LGS can be exported to THINGS files and command scripts used to allow the visualization of the scene in 3D in GDS software. The command scripts create solid objects that are stored in a GDS codex file. The objects from the codex and from the various THINGS files are input into the GDS SVS module where the scene is assembled and rendered in 3D space. The observer can move around the scene by changing the observer's eye-point. When vertical planar objects are used in GDS as 3D representations of LGS point objects, their effectiveness is degraded when the observer moves around the scene in SVS making the 'flatness' of these objects apparent. To alleviate this, a program was written to rotate all such objects to face the observer each time the observation point is moved.

The 3D visualization of the grammatically generated landscape scene represents the last step in the process of examining the spatial consequences of the landscape character definition that is encapsulated in a landscape grammar. The implementation involves the writing of class, parameter, and rule definitions, as well as rule and object selection strategies, using the CL programming language in conjunction with the functions of the LGS application. These definitions are utilized by the landscape grammar interpreter in conjunction with a GIS that stores a spatial database of landscape objects. As specified by the landscape grammar theory of Chapter 3, the interpreter iteratively applies the landscape grammar rules to an initial scene resulting ultimately in a final set of landscape objects that can be viewed graphically as a map in the LGS application. The final scene can then be exported as data files with command scripts that when run in the GDS 3D software allow the visualization of the grammatically generated scene as a 3D landscape simulation.

The use of a programming language and externalization of the 3D modelling adds a degree of difficulty to the operation of the system. It is not likely that a planning agency interested in modelling landscape character would be able to use the system in its currently implemented form. However, it can serve the implementation objectives of providing a research vehicle with which to demonstrate landscape grammar theory and explore the issues of implementing a spatial grammar interpreter for landscapes. The LGS implementation is based on the theory of Chapter 3 and the spatial grammars presented in Chapter 2. To illustrate the relation of this implementation to other spatial grammar mechanisms, demonstrations of the ability of the system to emulate other types of spatial grammars are presented in Appendix B.

4.9 Summary

The implementation of a landscape grammar interpreter presented in this chapter is facilitated through the development of an object-oriented, GIS-based and rule-based computing environment that is based upon the theoretical concepts presented in Chapter 3. The LGS application, developed in Common Lisp, allows a landscape vocabulary to be defined as hierarchical landscape classes, and a scene to be populated with landscape objects that are instances of those classes. The user's knowledge of how classes of objects are related in a landscape character can be represented as landscape grammar rules using CL and LGS functions. The LGS interpreter mechanism is then able to apply these rules to an initial scene of objects, modifying the scene at each iteration and culminating in a final scene that adheres to the user's ideas about landscape character as they are represented in the rules. The final scene can be visualized in two dimensions or exported to GDS CAD software for 3D modelling and visualization. Having now presented the implementation of the landscape grammar theory as a computer-based landscape grammar interpreter, the discussion now turns to the application of landscape grammars to the context of a planning agency, focusing specifically on how planners might construct a grammar for their region and then use it in particular planning applications.

Chapter 5

Landscape Grammars and Planning

The theory of landscape grammars presented in Chapter 3 detailed how a landscape's character can be modelled by extending the language of landscape metaphor, specifically using grammatical structures and mechanisms to articulate landscape concepts. An implementation of this theory as a computer-based application using object-oriented GIS and knowledge-based programming was described in Chapter 4. The current chapter relates the discussion of landscape grammars to the specific context of visual resource planning (VRP) introduced in Chapter 2. First, the general purposes to which landscape grammars may be put in VRP are discussed, relating the natural and cultural elements of a landscape grammar to the regulatory forces imposed by a planning agency. Methods suitable for a planning agency to construct a regional landscape grammar are then presented. Subsequently, the discussion turns to the integration of a landscape grammar into the planning process, and finally to the implications of landscape grammar use for approaches to planning practice.

5.1 Landscape Grammars in Planning

VRP was defined in Chapter 2 as the practice of working towards the achievement or maintenance of a desirable character for a region's visual resources. For the purposes of this dissertation, planning practice is defined generally as the devising, writing, and enforcement of policies and regulations to ensure the orderly development of land (Faludi, 1973; Davidoff & Reiner, 1973). The achievement or maintenance of a desirable visual landscape character is subsumed within this definition of planning practice and is the focus of 'planning' for the current discussion of landscape grammar application. The domain of interest is the activities of public sector planners, primarily at the local or regional level.

Landscape grammars can serve two main purposes for planning practice, thus defined: (i) a repository of local knowledge of a region's landscape character, and (ii) a means for encoding and exploring the potential consequential impacts of proposed planning policies and regulations on landscape character. The first purpose intends for planners to use landscape grammars as a vehicle for representing knowledge about the visual and spatial nature of objects within a regional landscape. In this context, a landscape grammar may be considered an ontology of the local visual landscape character. On the one hand, this 'landscape knowledge' pertains to observations of a region's physical objects of both natural

and cultural origin. Natural objects and patterns are products of animal or vegetation species adaptation, while cultural objects arise through extended human occupation. Statements pertaining to the types of trees that grow in a region, their relative abundances, the species that are seen in residential gardens, or used for roadside planting are examples of landscape knowledge that refer to the vegetation of a region. Knowledge of the width and linearity of roads, the occurrence of sidewalks, and the placement of benches or other objects in public areas touch on some of the cultural elements within a regional landscape. Perhaps most significantly, the planning of development must include knowledge of the nature of built structures that occur within a landscape, such as the typical scale and dimensions of buildings, the nature of their roofs, the use of architectural details such as dormers, window shutters, stairs, or finials, and ancillary structures such as walls, fences and sheds. On the other hand, in addition to these physical elements, planners must also preserve the integrity of more abstract and intangible spatial phenomena such as land parcels, outdoor amenity space, sightlines, viewsheds, watersheds, habitats and migration (for animal and bird species) and transportation routes. As discussed in Section 3.1.1, such abstract spatial concepts can be used as constructs for understanding the order or structure of a landscape.

Landscape grammar formalisms, as presented in Chapter 3, provide planners with a means of encoding landscape knowledge as sets of landscape class definitions and logical statements. The grammar interpreter mechanism discussed in Chapter 3 allows the knowledge contained in the grammar structure to be applied to a specific set of existing objects in a landscape. The LGS computer application presented in the previous chapter demonstrates the translation of those formalisms into an executable form. Hence, landscape grammars allow planners to articulate their landscape knowledge and then examine its validity by producing landscape simulations that may or may not reflect a desired landscape character. By evaluating landscape simulations generated from the grammar, based on their resemblance to the real landscape of the region, planners have a basis for detecting errors in their landscape knowledge. While more details on this process are provided in the next section, it is worth noting here that this capability to scrutinize one's ideas is valuable in itself, since planners are expected to hold such landscape knowledge but do not currently have the means to articulate and refine it.

It is possible for planners to develop a landscape grammar that is entirely superficial or syntactic, that is, without any reference to, or even investigation of, the reasons why objects are arranged the way they are in a regional landscape. It is more likely, however, that the knowledge on which a grammar is based incorporates, in addition to the inventory of object classes, some assumed understanding of the forces that shape the landscape. This might involve an acknowledgement of the deep tradition of an architectural style, the recognition of the growing conditions for certain species of trees, or the adoption of standards to make structures function appropriately. By incorporating some of this 'meaning' into the

landscape character, the landscape grammar takes on semantic content. These meanings are likely to be represented explicitly in the grammar as abstract spatial constructs, or non-terminal object classes, in a landscape vocabulary, as well as implicitly in the relationships defined in the spatial rules. In the landscape grammar implementation presented in the previous chapter, the explanation slot of rule objects allows for the semantic meaning of a rule to be explicitly described.

Planners use this syntactic and semantic landscape knowledge to formulate policies and regulations that are then applied, for example, to landscape shaping activities such as subdivision and development control. The intent of these protocols is to maintain a landscape character by influencing the form of changes to the physical landscape before they occur. Such policies refer to the natural elements of the landscape as well as its cultural elements. Planning regulations can also incorporate abstract spatial entities such as the concepts of setback lines, building envelopes, and development ceilings, as well as local and global parameters such as minima and maxima for parcel areas, site coverages and plot ratios. Planning regulations also typically state relationships between objects in the landscape, such as “buildings may occur no less than 100 metres from the shoreline” or “building setbacks of at least ten feet from lot lines”. In grammatical terms, such planning regulations offer new vocabulary as well as parameters and rules that augment, and may or may not be sensitive to, those already derived from the ecological and cultural norms of the region. In knowledge-based terms, they represent planners’ knowledge of how the landscape should be modified in order to maintain the desired landscape character.

When planning policies and regulations are enforced, they become forces that shape the landscape and thus become part of the landscape grammar and character for that region. They propose new non-terminal spatial classes as well as new spatial rules, while also providing a semantic explanation of why changes to the landscape are occurring in the manner that they are, because they must do so according to planning legislation. The introduction of new planning regulations in a community eventually influences the landscape character of the region by inducing recurrent changes across the landscape as new development occurs. The landscape grammar theory and implementation presented thus far in this dissertation provide a means for expressing these regulatory influences in the same grammatical form as our knowledge about natural and cultural features of the landscape. When proposed regulations are encoded as classes and rules and used as part of an existing landscape grammar formulation, changes to a landscape character caused by the regulations can then be anticipated. By generating simulated scenes from the modified grammar, the potential effects of the regulations can be visualized in advance of the implementation and permanence on the landscape. Thus, the use of landscape grammars in planning is extended from the representation of ‘landscape knowledge’ to the second purpose as stated earlier, namely, a means for encoding and exploring the consequences of actual and proposed planning policies and regulations.

In the illustration of a simple landscape grammar from Section 4.7, such regulatory elements of the grammar can be identified in the `Setback-Line` class, the `setback` slots of other landscape classes, and the `make-setback-line` rule that calculates where the line should occur. It should be noted that the setback distances in this case might not necessarily be derived from planning regulations, but instead from observations of cultural norms in local development practices. The distance values in the `setback` slots could be modified to generate a simulated scene that relates this one aspect of a zoning by-law in the study area. Because they commonly operate at an abstract and semantic level in a grammar, the representations of planning regulations as classes and rules have structural influence over the generation of a grammatical landscape scene, just as the regulations themselves tend to provide the structural constraints within which the larger objects and patterns of a landscape (such as roads, parcels and building envelopes) are permitted to exist. Because of this structural role, regulatory rules should be processed early in the grammar interpretation process.

Difficulties with adopting grammars in planning arise when temporal aspects of landscape character are considered. As new spatial objects and rules proliferate through a region, the visual landscape character of that region as a whole may be significantly modified. A regional landscape character is dynamic, albeit generally changing at a slow enough rate for planning to remain a viable and worthwhile activity. In this context, Figure 5.1 illustrates the possible evolution of a landscape from a completely natural environment to an increasingly developed one, implying the changing nature of the landscape's vocabulary and grammatical rules in their inclusion of ecological, cultural and regulatory

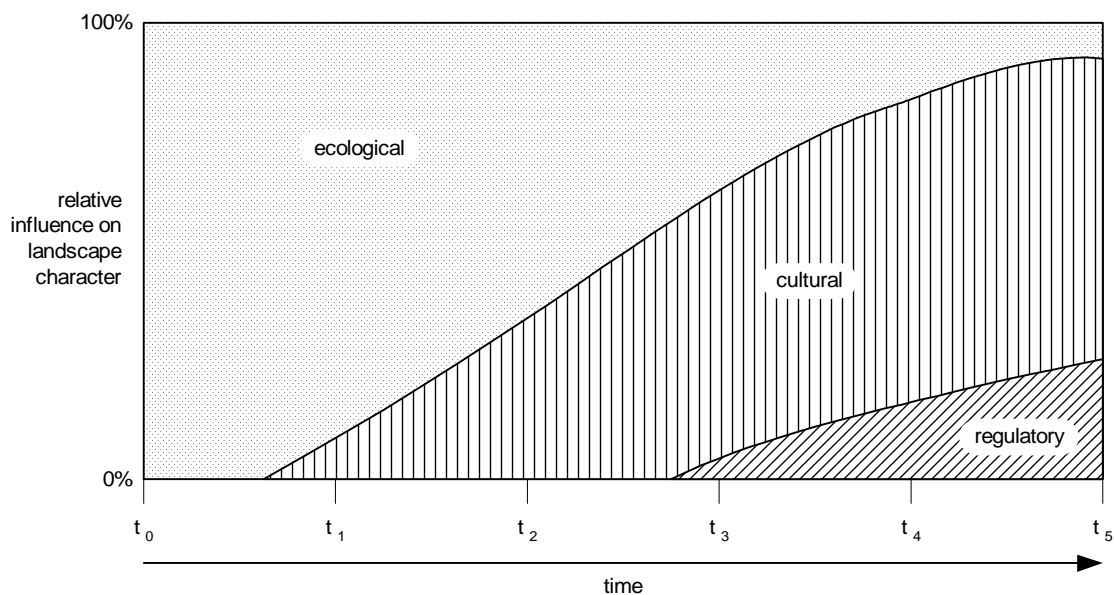


FIGURE 5.1 LANDSCAPE EVOLUTION AND LANDSCAPE GRAMMAR ELEMENTS

object-types and patterns. In pre-settlement times, the landscape's character, and thus its grammar, are wholly influenced by ecological (as opposed to cultural or regulatory) objects and spatial patterns (t_0 in Figure 5.1). If the landscape in question were currently developed, it may be possible to model its previously ecological landscape character, given the proper historical and geographic data. The grammar would use only ecologically based classes and rules and would be similar to a wholly natural area existing today. In Figure 5.1, t_0 is set at an arbitrary point in time preceding human occupation but the time scale could be expanded greatly leftwards to represent more realistically the extended time scale of natural landscape evolution.

Human settlement introduces new vocabulary and new spatial patterns into the landscape. Even after initial human habitation (t_1 , t_2 in Figure 5.1), there may be a period of low-density development (such as an agrarian village), in which there is little or no regulation to control development. During this time, cultural or human-made objects and patterns could be introduced but would still be heavily influenced by the patterns of the surrounding ecosystems. A grammar for the landscape character of this period would involve cultural and ecological object classes and rules, with the former becoming increasingly dominant as development intensifies. As human settlement becomes more organized (say from a village into a town), community regulations governing the form of development would likely arise (t_3 in Figure 5.1). The regulatory aspects of the landscape's character then would play an increasingly influential role in the formulation of landscape grammars governing the visual resource (assuming planning activities are increasingly valued in the governing of the community). This would become increasingly likely if the landscape in question were to develop into a major urban centre, represented at the rightmost part (t_5) of Figure 5.1.

The ecological-cultural and cultural-regulatory boundaries in Figure 5.1 are to some extent arbitrary. The shapes of these two boundaries could vary with different circumstances in the landscape. For instance, in the rapid development of a natural area, the ecological-cultural boundary would be much steeper as cultural objects are introduced *en masse*. Similarly, if a regional or local regulatory plan were to be introduced with greatly increased regulation of design elements, then the cultural-regulatory boundary would rise steeply at the commencement date of the plan (although it may be flattened somewhat if the regulations are enforced gradually). A forested landscape that becomes subjected to a forest management plan might not be influenced by cultural factors at all, showing only an ecological-regulatory boundary in its evolutionary diagram.

To incorporate a time sequence into a landscape grammar for planning requires the grammarian to investigate the history of the region to determine which influences have been present, and then, evaluate their impact on the landscape. The built architecture, for example, of a particular landscape scene may be composed of objects from different time periods. Consequently, grammars can be

constructed more easily for landscapes that are developed quickly, since the development rules are more likely to be consistent and potentially less complex. When building a landscape grammar for a particular region, the staging of natural, cultural and regulatory events must be determined in order to assign the grammar some temporal validity.

Figure 5.1 illustrates the progression of types of classes and rules over time from an entirely natural grammar to the introduction of cultural objects and patterns and later the introduction of regulatory influences as suggested above. It may be appreciated easily that the introduction of a cultural grammar into a region will likely affect the existing natural grammar, insofar as human settlement is seen as a disruption to natural ecosystems. Similarly, the introduction of landscape-based regulations (such as wetland preservation) may affect, positively or negatively, the existing natural and cultural grammar of a landscape. Regulatory rules may change the environment such that certain natural/cultural objects or patterns consequently flourish, diminish or disappear altogether. Since the impacts of planning regulations can alter the patterns of natural and cultural objects, regulatory grammar rules may cause cultural or ecological rules to be rewritten relative to earlier versions of the same. It may also be that new regulatory elements have no effect on landscapes. Planning regulations can be considered to have a ‘probability of adoption’, that is, the probability that development or conservation incentives will be utilized by the public or that controls will be heeded and/or enforced.

In summary, it can be seen that the development of landscape grammars for planning serves to articulate knowledge about the character of a regional landscape in terms of its natural, cultural and regulatory components. The natural and cultural components may be sufficient for describing the landscape character of a place for the purposes of refining planners’ understanding of its landscape. They also underscore the importance of the temporal scale in the development of a landscape’s character. The introduction of regulatory objects and rules to a landscape grammar provide the ability to represent past, current and proposed planning policies and regulations in relation to our knowledge of the natural and cultural patterns of the landscape. These are the primary roles for landscape grammars to play in VRP. The following section proposes methods relevant to the first purpose noted earlier in this section, that is, building a landscape grammar for a regional landscape character. Section 5.3 then focuses on how to incorporate the use of a landscape grammar, once constructed, into planning practice.

5.2 Methods for Landscape Grammar Construction in Planning

The methods used to construct a landscape grammar(s) for a particular region will depend to some extent on the nature of the landscape in question (for example, natural, rural, countryside, town, or urban), the availability of information, the availability of local participants to construct and evaluate the

grammars and the uses to which the grammar(s) is put. The implications of these considerations are discussed further in the next section. At this point, the following discussion outlines a general approach to constructing a landscape grammar, assuming a settled landscape and sufficient information and participants. The methods are applied to a study area in Bermuda in subsequent chapters in the dissertation. The general process is outlined in Figure 5.2. First, specific sites within the region must be identified for study. A further scoping exercise identifies from the chosen sites the types of landscape objects and spatial patterns that are visually significant or meaningful to the current or desired landscape character. Using the metaphor that is central to this dissertation, these comprise the landscape’s language elements and in identifying these elements, the landscape’s legibility is articulated. This involves assessing how those objects and patterns that are present contribute to the landscape’s surface structure as well as the meaning behind its deeper structure.

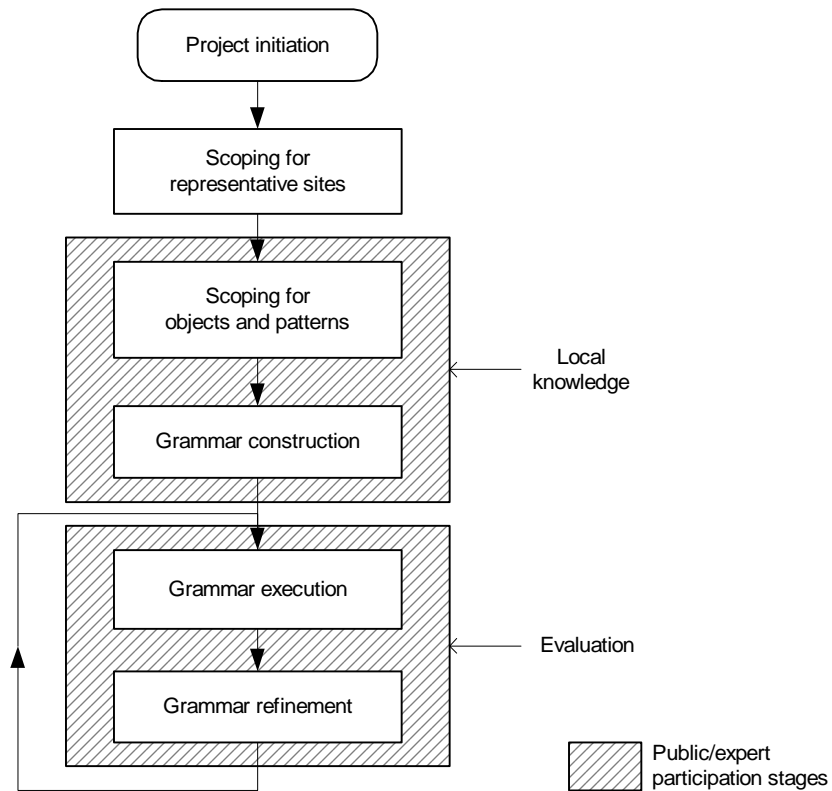


FIGURE 5.2 THE LANDSCAPE GRAMMAR DEVELOPMENT PROCESS

The next task in the process is to translate these elements into a vocabulary of classes and sets of rules using a landscape grammar system such as the LGS implementation, thus constructing a preliminary landscape grammar for the region. The grammar can then be applied to a site to generate landscape

scenes that embody the character defined by the grammar, as was illustrated in Chapter 4. Finally, the scenes and hence the language/grammar relationship are assessed to determine how well they embody the desired landscape character, and a continual process of grammar refinement begins. Input from various participants can be sought in the scoping and scene evaluation stages especially to identify the deeper meanings of spatial patterns. The stages of this process for landscape grammar development are now described in further detail.

5.2.1 Scoping

The different scenes encountered in a regional landscape will display that region's landscape character to varying degrees. That is, some scenes will be better examples of the landscape character than others both in surficial and deep terms. If we perceive the landscape character as a grammar, then the scenes of a region may be seen as more or less grammatical of the region than other scenes. Therefore, in the two scoping stages of Figure 5.2, the first aim in analyzing a landscape character is to distinguish those landscape scenes that embody the character from those that do not, and then to examine the visual and spatial structure of the desired scenes. In a linguistic analogy, the first step is to select representative sentences or phrases from which to construct a vocabulary and grammar rules to govern the language. The identification of distinctive landscape scenes is also made necessary when the study area is so large that it is not feasible to survey it comprehensively or that there are high variations in visual character within it.

A methodological issue arises here in that sites may be selected because they represent the regional landscape character well, but the whole point of studying those sites is to arrive eventually at a definition of that same landscape character. Hence, the selection criteria for the first step depend upon the findings of the second step in the process. The only obvious way to overcome this problem is that the grammarian must have some *a priori* knowledge of the desired landscape character before the scoping process begins. Reliance on the grammarian's subjective interpretation does not seem unreasonable for this purpose if it is required that he/she should be highly familiar already with the regional landscape character being studied. Even in this case, however, the grammarian may select a site, begin scoping for object-types and patterns, and then find that the site bears some atypical characteristics and that more representative sites in the region may have been overlooked. The implication of this aspect of site identification is that the scoping process is not necessarily linear or simple, which is also true of the scoping process for objects and patterns.

The criteria used to select particular sites within a region will thus depend upon the perceived nature of the landscape character under study. That is, the sites should contain a proliferation of the objects and patterns that are perceived to comprise the desired landscape character as a whole, whatever

those objects and patterns might be for the region at hand. The selection of sites may be further curtailed by the purposes to which the grammar will be used, such as demonstrating the effects of a particular regulation on the landscape prior to enforcement or implementation.

The purpose of the second scoping stage of the landscape grammar development process is to identify those objects and patterns, or language elements, of the landscape's character that will be included in the landscape grammar itself. There may be many types of objects in the region that exhibit spatial patterns in some form and to some extent in landscape scenes. Accounting for all possible objects and patterns not only jeopardizes the feasibility of the grammar development process, but it may also be unnecessary for representing the region's landscape character. In linguistics, the non-redundancy of language (that multiple words are generally not created for the same concept) is relative to the experience of the local culture. Arctic cultures identify fifteen different types of what those in the 'south' refer to simply as "snow", because of their deeper experience with snow in their environment (Woodbury, 1999). In modelling landscape character, non-redundancy is also a useful principle in that differentiations between types of objects in a landscape grammar need only be as detailed as is meaningful for the landscape character. Where pine trees in a region are few in number and have little traditional significance locally, it is not beneficial to differentiate between various kinds of pine trees in a landscape grammar. Some objects and patterns will therefore contribute to a landscape character more significantly than others. There may be 'signature' visual characteristics that are commonly referred to in descriptions of a region's landscape character. In this context, Seebohm (2000) contends that the work of landscape artists, especially Impressionists, may offer insights into those visual features that capture the essence of a landscape's character.

The level of abstraction in the grammar model will also depend on the level of detail that can technically be represented by the visualization technology used by the grammar interpreter system. For example, because the GDS software used in the LGS application does not use texture mapping or scanned images, the representation of vegetation in its 3D models is limited, and therefore detailed differentiation of tree-types in an LGS grammar will not be visually discernible in the outputs. In a system not affected by this limitation, the grammar could make use of subtle or detailed differentiations between types of objects, such as between different kinds of palm trees for example. In summary, the primary objective of this scoping stage is to identify those objects and patterns that are significant language elements in the desired landscape character. An incremental approach to grammar development is useful here, determining the more frequently occurring object types and rules before including those with lesser frequencies.

The identification of these key objects and spatial patterns may be accomplished by a variety of methods as shown in Figure 5.3. Despite its representation as a simple flowchart here, the scoping stage

is likely to be an iterative process. The identification of objects and patterns using one method can lead the landscape grammarian to return to previous methods to re-examine them (as represented by dashed arrows in Figure 5.3). Some methods may be omitted depending on the availability of information and participants or the particular uses for which the landscape grammar is designed. In addition, the methods need not occur in the order presented in Figure 5.3, although the reasons for the presented order are referred to in the paragraphs below.

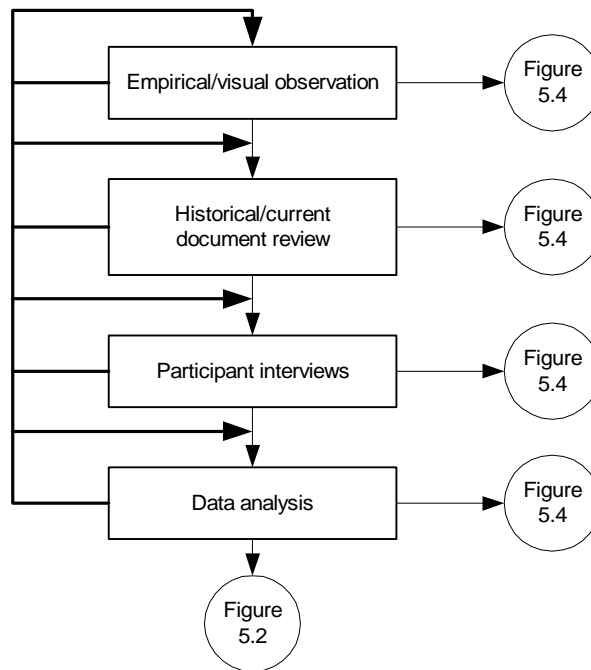


FIGURE 5.3 SCOPING IN LANDSCAPE GRAMMAR DEVELOPMENT

Empirical field observation entails the grammarian spending time on the site gaining practical visual experience and familiarity with its landscape objects. The tasks to be performed using this method, and each of the other methods in Figure 5.3, are shown in Figure 5.4. For each task, the objective is to identify the types of objects and their typical attributes (which become the grammar’s landscape vocabulary), as well as the spatial and visual patterns that these objects exhibit in the landscape scenes (which are later translated into the grammar’s rules). As already emphasized, effort should be concentrated on those objects and patterns that seem to contribute the most to the visual character of the area under scrutiny. The identification of object-types and patterns as being of natural, cultural or regulatory origin helps to modularize the landscape knowledge for use in the grammar. These details form a visual inventory of the site. A hard copy or digital map of the site can be annotated with names of the objects observed. Aerial photography allows partial visualization of those parts of the landscape that

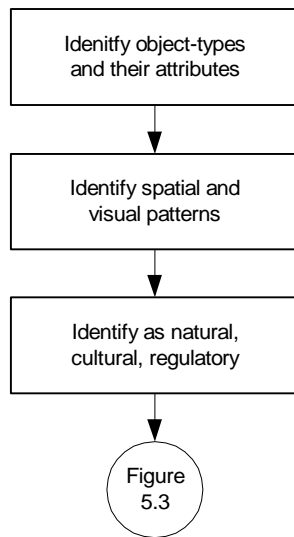


FIGURE 5.4 IDENTIFICATION OF GRAMMAR COMPONENTS

are not accessible by foot. Using sketches to record the observed spatial patterns and relationships between the objects helps to develop further the grammarian’s familiarity with the landscape character. Throughout the survey, the observations may be both qualitative and quantitative. Particular details may include the observed frequency with which a type of object occurs, the physical characteristics it typically exhibits and the locations in which instances of the object-type generally occur relative to other types of objects. While it is the specific objects in the landscape that are observed, the objective of the visual inventory for landscape grammars is to make *generalized* statements about these objects, their attributes and locations that are inferred from the on-site observations. The grammarian not only reads the landscape for its syntax, but also, if possible, for a deeper understanding of the landscape’s semantic content (perhaps, linguistically speaking, ‘reading between the lines’).

Another stage for building a landscape knowledge-base for the sites under study is the review of historical and current documents relating to the sites (Figure 5.3). For natural areas, this may include past environmental surveys that can explain ecological structures and dynamics. For developed areas, archival descriptions and photographs of the site, past and current planning legislation, regulations and guidelines, records of subdivision and development permissions and any neighbourhood agreements can inform the grammarian about underlying structural rules that have guided development on the sites. In places where there are distinctive architectural styles there may be treatises on the architectural heritage of the area. A development history of an area can reveal whether, for example, all parts of the site were developed simultaneously, incrementally outwards from a central building, inwards from its periphery, or in no specific spatial order. If the site’s development was incremental, there may be different sets of planning regulations that controlled development at different periods in time. Document review is probably the

most valuable source of information for writing regulatory grammar rules, although the impact of documented policies and regulations on the physical landscape will also depend on how they are interpreted and enforced on building projects. One can reasonably expect regulatory relationships to exhibit less variability across a landscape than natural and cultural ones, if planning regulations are applied consistently over the entire area.

Field observations provide the grammarian with, at the very least, a preliminary idea of the objects and patterns that contribute to a landscape's character. Planning and other documentation offer insight into the forces that shape development. The knowledge gained from these two tasks can be used as a basis of discussion with other people who are familiar with local landscapes. Participant interviews provide the grammarian with alternate ideas and opinions pertaining to the significant elements of the local landscape character (Figure 5.3). This allows a broader knowledge-base from which to build the landscape grammar. The subjective assessment of a landscape's aesthetics is often seen in landscape studies in terms of expert/professional versus social/public opinions (Smardon et al., 1986).

The use of design and planning professionals in landscape studies is advantageous because such individuals have had to develop skills of 'reading' and 'writing' the local landscape in various aspects of their professional work. This is especially the case if the region possesses a heritage of design practice that is sensitive to traditional landscape appearances. Hence, the role of professional participants is to provide insight into two aspects of the landscape: first, the contribution of various landscape components to the overall visual character, which is based on the professionals' experience in designing and reviewing development proposals that maintain the local landscape character; and second, the decisions that shape the development of a site such as the reasoning of the designers and planners involved with the sites, which reveals semantic meaning for the chosen landscapes and understanding of their structural patterns. The opinions of such landscape 'experts' can differ from those of the general public, and the use of non-professional, or general public, participants in the process of scoping for objects and patterns is in accord with established planning principles regarding public participation. Such interviews, however, do require of the participants a high level of familiarity with the local landscape character regardless of their professional background.

Table 5.1 displays a matrix with the classes of professional status and familiarity with the local landscape, as well as examples of interview groups that can be attributed to the extremes of these classes. It is suggested here that for the purposes of identifying the particular objects and patterns that contribute to a landscape's character and explaining the structural influences on the local landscapes, individuals with high landscape expertise and high familiarity with the local region (that is, "expert insiders") are the most useful. The general public, or "non-expert insiders", are more valuable for the evaluation of landscape scenes later in the grammar development process, providing a measure of public acceptance of

	Professional Status	
Familiarity	<i>Expert</i>	<i>Non-expert</i>
<i>Insider</i>	Local planners	Local citizens
<i>Outsider</i>	Foreign architects	Most tourists

TABLE 5.1 TYPOLOGY OF GROUPS FOR INPUT ON LANDSCAPE GRAMMARS
Source: Bourassa (1991)

the general landscape character. An argument can be made that “outsiders” can also provide insights about the landscape that are perhaps too obvious for locals to see. While they are perhaps in a good position to assess the uniqueness of a place’s surficial character compared to other places, since they are primarily from places other than the locality, outsiders may not however be in a good position to articulate details of the more subtle structural characteristics of a place.

The fourth method for studying the objects and spatial patterns that contribute to defining or shaping a visual landscape character is the use of spatial and non-spatial analytical tools with landscape data from the site(s) under study (Figure 5.3). Such data may be obtained from available digital GIS databases generated from aerial photography, or collected from previous surveys or the grammarian’s field observations (using, for example, digital site photographs). Landscape data can be used at this stage to verify the ideas garnered from field observations and participant interviews. Note that spatial analysis in the scoping stage is not necessarily performed in a landscape grammar system such as the LGS application from Chapter 4, although that system contains functions for spatial analysis. Instead, conventional GIS software which is specifically designed for the spatial analysis of existing landscape data can be used. It would be unreasonable to place a general limit on the type of analytical operations and tools that are appropriate for the scoping stage of landscape grammar development. The landscape grammarian should be free to utilize any analytical techniques that are appropriate to the available landscape data. It is beyond the scope of this dissertation to provide a comprehensive review of all landscape analysis methods that could be used in the scoping stage of landscape grammar development. Likewise, it would be inappropriate to suggest here that there is some standard process that applies to the analysis of all landscapes. It should be stated, however, that the objective of the data analyses undertaken at this stage is to infer generalizations about or relationships between the types of objects found in the landscape.

To this end, descriptive non-spatial statistical measures may be used to characterize the attribute values of a set of objects. Descriptive measures of central tendency or dispersion can be used, such as the mean number of bedrooms in a set of houses, the proportions of sheds that are made of wood, stone,

or aluminum, the range of length values for lot frontages, or the variance of the heights of trees in the study area. In addition to non-spatial statistics, similar measures can be used for describing a set of objects spatially (Silk, 1979; O'Rourke, 1993). Coordinate values for the average x-coordinate and average y-coordinate for a set of points, for example, gives the mean centre location. The geometric centroid of a set of points is an alternate measure. The standard distance deviation, as well as various other techniques of spatial autocorrelation and nearest neighbour analysis, is a measure of the dispersion of points around the mean centre point, and can be used to describe, for instance, the degree to which the locations of trees of a particular species are clustered spatially. The minimum bounding box and convex hull for a set of objects represent descriptions of the minimum area covered by those objects.

Other statistics look at the relationship between two different variables or types of objects. Correlation coefficients are a measure of the propensity for collocation on two variables, such as the incidence of specific disease and the presence of environmental causal agents. Spatial auto-correlation measures the relationship between object locations using distance-based variables. For example, the locations of pool-houses are positively auto-correlated with the locations of swimming pools (as the former are found near the latter), while the locations of tourist accommodations are negatively auto-correlated with the locations of waste landfills (as the latter are not attractive for tourists). In addition, the predicate relations discussed in Chapter 3 (for example, inside, outside, adjacent to, above, below, intersects, left/right of) define relationships between two or more objects that result in a true/false statement rather than a coefficient.

Other spatial analyses define new spatial data from existing data. For example, line generalization techniques can be used to simplify or "smooth out" the complexity of a linear object. Applied to landscape data, this might be used to generalize a highly erratic shoreline into a simpler line that suggests the general orientation of the shore. Terrain analysis can be used to generate the slope and aspect (facing compass direction) values of each unit of the land terrain from elevation data. Buffering operations create a polygon whose perimeter is a constant or variable distance from an object which can be used with landscape data to infer, for example, the area affected by traffic noise around a road, or the area around a fuel storage facility that must be kept clear of other objects. A web of Thiessen or Voronoi polygons can be calculated around a set of points to infer areas of influence such as the trade areas of a set of businesses (Boots & South, 1997; Okabe et al., 2000). Polygonal overlay operations that determine the areas of coincidence for two or more polygon objects might be used to infer the areas of woodland that typically intersect a land parcel or development zone.

There are many more methods available to analyze landscape data and new landscape metrics and analyses are continually under development. Any of these analysis techniques may reveal to the landscape grammarian new insights into the patterns of the landscapes under study. These patterns are

later expressed as a landscape grammar in the construction stage. As illustrated in Figure 5.3, the tasks of field observation, document review, participant interviews and data analysis may be repeated until there is an acceptable level of 'landscape knowledge' for the study site both on its surficial syntax of objects and on their deeper meaning for residents. The second scoping stage of landscape grammar development can thus entail each of these methods to identify and investigate the types of objects and patterns in a landscape, in order to understand the composition of its landscape character. Once such an understanding is attained, it must be translated into a vocabulary of landscape classes and a set of grammar rules in order to construct the landscape grammar.

5.2.2 Grammar Construction

Once an inventory of objects and patterns has been examined and a general characterization of the landscape has been made, these generalizations must then be formalized into an executable landscape grammar comprised of class definitions and spatial rules. As stated in previous chapters, the landscape vocabulary consists of a hierarchy of landscape class definitions, each with a shape-type and set of attribute definitions.

It was noted in Section 3.1.1 that there are several ways to define a class hierarchy since some attributes may be used as the basis for differentiating between landscape classes. The categorization of landscape objects into classes is thus subjective for the grammarian and different hierarchies may be suitable depending on how they are referred to in the rules. The hierarchical organization will also depend to some extent on the level of abstraction that is desirable or achievable in the landscape visualizations. If there is a high level of abstraction in the landscape models generated by the grammar interpreter then the class hierarchy can define broad classes of objects.

The selection of types of objects for inclusion in a vocabulary can be aided by the use of frequency measurements for potential object-types, if such measurements are collected in the scoping stage. Those types of objects that occur most frequently are good candidates for becoming a landscape class in the hierarchy. Similarly, observed value ranges for attributes of those objects can be used to express the default value for an attribute definition of a landscape class (for example, the height of a tree is calculated to be a random number between say 5 and 20 metres).

The construction of grammar rules is based on the identified relationships between classes of objects and/or their attribute values as they are found to occur in the landscape. This process can be difficult to establish, since many of the generalizations about landscape features can include statements that are intuitively obvious for humans but which need to be articulated further for use in a computer-based processing environment. For example, the statement "[if] a tree is near a house ..." can be restated as a spatial grammar formalism as:

near(tree,house),

and then into an executable form using LGS syntax:

```
(and (is-a ?a `Tree) (is-a ?b `House) (< (distance ?a ?b) 5)).
```

The formalization of this statement required the interpretation that “near” for trees and houses means separated by a distance of less than five metres. The formalization of observations into rules can also entail the translation of the observations first into expressions of geometric primitives before the subsequent expressions are translated into program code. For instance, while human interpretation might easily infer the general orientation of a winding shoreline, an executable rule statement may need to calculate a directional trend from the geometry of the shoreline object before calculating its orientation. Hence, the process of articulating landscape generalizations into an executable form may well involve several revisions to the vocabulary classes and grammar rules.

The frequencies with which some classes of objects are observed to occur in the physical landscape in the scoping stage can be used in the construction of rules. The frequencies of objects observed in the field become the probabilities with which the relevant classes of landscape objects are instantiated in the working scene simulation. For example, the observation that 40% of all garden walls in the landscape are capped can be used in an LGS grammar rule as follows:

```
IF: (is-a ?a `Garden-Wall)
THEN: (set-label ?a (one-of `(Capped Uncapped) `(0.40 0.60))),
```

where the `one-of` function selects an item from the first ordered list with the associated probabilities stated in the second ordered list.

In the construction stage of landscape grammar development, the articulation of humanly intuitive observations into executable forms is perhaps the most significant challenge. It requires creativity and insight from the grammarian in order to design a landscape grammar that successfully generates a desired character of scenes upon execution.

5.2.3 Grammar Execution

Once assembled a landscape grammar is applied to a site simulation in order to produce a set of landscape scenes. These scenes represent some of the possible spatial consequences of the ideas embodied in the grammar. Since the grammar is seen to represent planners’ and/or others’ knowledge of the local landscape character and its ‘language’, the generated scenes represent the application of that knowledge to a specific site. By looking at these simulated scenes, those and other planners can visualize possible results of applying their knowledge. If the grammar represents regulations to be coded in a plan, then some of the possible landscape consequences of enforcing those regulations are able to be

visualized. The role of visualization is significant here because it is the means through which errors or flaws in the grammar may be most easily detected. It should also be noted that the purpose of a landscape grammar is to visualize a landscape 'style' and this is achieved through the visual manifestation of the grammar's formal language, that is, the potentially infinite set of generated landscape scenes. Thus, in order to appreciate the grammar's effectiveness several generated scenes should be visualized and evaluated for their efficacy or relevance relative to the desired landscape character.

Section 3.3 of Chapter 3 discussed the problems associated with the infinite number of scenes that could be generated. One practical approach to the issue is to allow the grammarian to bias the elements of the scene generation in order to create worst-case scenarios. This is achieved by modifying rules to reduce the use of stochastic statements. For example, to achieve a maximum-development scenario, the rule statement:

```
IF: (is-a ?a 'Building)
THEN: (set (height ?a) (a-number-between 20 30))
```

can be replaced with:

```
IF: (is-a ?a 'Building)
THEN: (set (height ?a) 30)
```

where ?a represents a Building object. Another approach is to decide upon criteria for culling the set of generated landscape scenes, thereby eliminating unacceptable or high redundancy scenes. The grammar interpreter would perform the application of such criteria after the firing of rules is complete. Such a mechanism could be built as an enhancement to the LGS application presented in Chapter 4. In executing a landscape grammar, planners might also try to sample the set of possible scenes stochastically, although it is not clear how such sampling might be achieved with a grammar that contains many stochastic statements and an infinite number of scenes. Despite the inability to generate a comprehensive collection or representative sample of landscape scenes, the exploration of a limited set of landscapes is a valuable activity because it provides further exploration of the consequences of land-related ideas than would otherwise be available to planners using non-automated landscape simulation methods.

5.2.4 Grammar Evaluation and Refinement

The scenes generated by the execution of the landscape grammar interpreter are then evaluated for consistency with the desired character. The purpose of the evaluation stage is to identify how the grammar vocabulary and rules need to be refined in order to achieve the desired character. While the grammarian can perform the evaluation alone, there is also a role for further participants at this stage. The involvement of participatory groups at this stage serves to make the knowledge stored in the

landscape grammar more publicly based, and, from the perspective of the accepted participatory planning model, more valid. When a participatory approach is used, the visualization of the grammar's outcomes as a 2D or 3D landscape simulation is advantageous because it facilitates communication and discussion among participants.

Regardless of the number of evaluators, the evaluation and refinement stage involves the identification of flaws in the grammar as evidenced in the simulated landscape scenes. Apart from the legitimate inherent inaccuracies in the landscape grammar (that is, the grammar does not accurately represent the landscape character), a number of other potential sources of flaws are possible. One has already been mentioned in the grammar construction stage, namely the difficulty of translating ideas into geometric terms for use in a processing system. In everyday activities, we often do not need to articulate landscape generalizations in detail on the assumption that we share a common landscape experience with others. The geometric articulation of those notions can thus prove difficult. If it is assumed that the translation to a grammar will require simplifications of the real-world objects, then the grammar will be innately imperfect as is the case for any other model of reality. This will also be the case if an incremental approach to grammar construction is adopted. By focusing on the most frequently occurring types of landscape objects, others will be omitted necessarily and it may become evident in the generated landscape scenes that some of the omissions were in fact significant to the visual landscape character.

The level of abstraction used in the landscape grammar will also influence the evaluation of the scenes generated from it. Viewed from distances that are too close for a given level of landscape abstraction, the generated scenes, and by implication the grammar, will always seem inaccurate as it will reveal the (perhaps deliberate) lack of realism in visual details. Hence, scenes generated by grammars that represent buildings as massing models should be viewed from a greater distance than for those generated by grammars that include architectural details intended to be viewed in close proximity. Given that a landscape can be represented with varying degrees of generalization, it follows that the visual landscape character of a place can be represented by simple or complex grammars depending on the desired level of abstraction. The point of significance for the evaluation and refinement of landscape grammars is that the grammar should only be evaluated with consideration to the level of abstraction with which it was designed. Of course, the incremental approach to grammar construction allows that flaws due to abstraction can be subsequently refined with vocabulary classes and rules that inject more detail into the landscape scenes.

It is inevitable that the dynamic nature of landscapes and landscape character will eventually render a landscape grammar out-of-date. As landscape patterns change over time, the character of regions evolve, just as languages evolve with the introduction of new words and phrase forms. Hence, the landscape grammar defined for a place must evolve with the place's character if it is to reflect that

character accurately. For planning purposes, this is probably not much of a concern since it is often the case that, ignoring natural catastrophes, landscape character change takes place on a larger time scale than planning activities (five to ten years).

A participatory approach to grammar evaluation and refinement undoubtedly opens debate on grammar elements and introduces the complications of attaining consensus. Perceived flaws in the landscape grammar can arise purely from disagreement among participants on the nature of the landscape character. Even if consensus on a landscape character has been attained, there may still be disagreement on the degrees to which the generated scenes reflect that character. If multiple participants are to be used in the evaluation of grammatically generated landscape scenes then it is advisable that the same participants be involved in the scoping stage to establish the nature of the landscape character that is to be modelled. On the positive side, the involvement of such groups serves to verify previous field observations that have been set down in the grammar. The main requirement of participants is thus a substantial familiarity with local landscape settings (Table 5.1).

The consolidation of multiple opinions is by no means trivial and is a commonly encountered difficulty in planning activities. The construction of a shared knowledge-base using landscape grammars is subject to these same difficulties. The involvement of multiple participants invites the notion of multiple conceptualizations of reality. This is a key concept in postmodernist planning which asserts that knowledge is imperfect and subjective. While it is not the aim of this dissertation to address the rationalist/non-rationalist debate in planning theory, it is obvious that the landscape grammar concept, in its aim to encode a central knowledge-base, is rationalist in nature. However, it is also tempered to some degree from pure rationalist thought, by the acknowledgement here of an imperfect knowledge-base. It is assumed that some consensual, or at least majority, opinions pertaining to some topic may be reached. This assumption underlies many planning activities that deal with the notion of 'public good'.

Once certain flaws or gaps in the grammar are discerned and their causes ascertained, the grammar can be refined and executed again to see the new results. It is at this point that the process enters a continuous cycle of grammar execution, evaluation and refinement. The perpetuity of this cycle is based on the assertion that the landscape grammar will always be imperfect, that is, it will never completely account for all features and patterns observed in a landscape. The implications of this acknowledgement of grammar imperfection are that the process of landscape grammar development, once initiated, must be continuous, and the grammar may never be complete. It becomes an evolving repository of 'landscape knowledge', defining what *can* be said about the local landscape character. Because the development of the knowledge-base is a learning process for the planning agency, the utility of landscape grammars lies as much in the construction process as in the grammar that is developed. The assumption of imperfect knowledge changes the utility of a grammar from that of attaining knowledge, to

that of improving knowledge. To the extent that the improvement of planning knowledge is a worthwhile pursuit, the development of a landscape grammar is worthy of planners' consideration.

5.3 Use of Grammar Visualizations in Planning

Although a constructed regional landscape grammar is likely to be kept under continual refinement, the landscape knowledge that is attained through grammar development is relatively pointless if it is not applied at some stage. While this dissertation does not go so far as to institute landscape grammar modelling within a planning department, this section identifies several situations in planning activities that may benefit from the use of a landscape grammar knowledge-base. Each has different requirements regarding the type of knowledge that the grammar must contain.

The obvious application of a landscape grammar is to simulate the development of a vacant area of land according to the ecological, cultural or regulatory principles encoded in the grammar. The pertinent data for the site are input into the landscape grammar interpreter as an initial scene, and the execution of the grammar iteratively inserts new objects into the scene, thereby creating several simulations of the site in a developed state. Because the landscape grammar data and simulations are easily handled as digital entities in a computing environment, the same process can be used on an already developed site to explore what the site might have looked like had it been developed under different conditions. Data relating to any landscape objects currently on the site are removed from the initial scene in order to make it a 'vacant site' again. The exploration of various 'what-if' development scenarios is achieved by changing the parameters or rules in the landscape grammar and executing it various times on the initial scene.

In other uses of landscape grammars, the initial scene need not be vacant before a grammar is applied to it. The alteration of an existing development can be simulated using a grammar that is designed to account for, work around or modify the existing objects of a landscape scene. For example, a landscape grammar can be designed to model infill development in which new structures are inserted in any remaining land between existing structures. Alternatively, a more detailed landscape grammar might use rules to replicate the incremental addition of rooms to existing buildings over time (Brand, 1994). Because of their different functions, grammars that modify existing landscapes do not need to include all of the objects and rules of grammars that operate on vacant land. These two applications may, however, be considered as two grammars that can be incorporated into one landscape grammar which behaves differently depending on whether its working scene is vacant or occupied.

Another interesting possible application is the transfer of a landscape grammar constructed for one locality into another locality. This represents the application of knowledge derived from one

community to the physical site of another community. If the site characteristics of the new community are significantly different from those of the grammar's locality, the grammar's rules will most likely not apply, although some subsets of rules may be applicable to restricted parts of the new site. While the transfer of a landscape grammar to a new locality is only theoretical, it may serve as a means for knowledge-sharing between planning agencies. Such transfer may also be considered undesirable in that it may be seen to perpetuate the homogenization of landscapes in different places, thus exacerbating "placelessness". In this vein, the transfer of design standards has been used routinely to achieve efficiency in the reproduction of types of mass-developed suburban residential neighbourhoods in North American communities.

The applications suggested above focus on the relatively quick development of a landscape. Other applications are not so directed and specialized, but rather operate with fewer rules and more iterations representing a longer time period for landscape change. For example, rather than planning the potential insertion of new objects into a landscape for a specific development project, grammar structures can be used to simulate the effects of ongoing landscape management practices over time. Forestry management practices, along with knowledge of tree growth, can be encoded into a set of rules and applied iteratively over a large scene to simulate the gradual effects of such practices on a forested landscape. If sufficient data are available, historical landscape grammars can be constructed based on known or theoretical spatial relationships between the types of landscape objects that are known to have existed at that time. Such historical landscape grammars could apply to early forms of human settlement based on knowledge of living practices at that time. Pre-settlement natural landscapes can be generated from a grammar that encodes knowledge pertaining to the ecological behaviours of natural landscape objects. To extend the application even further, geologic forces can be modelled as rules to simulate the incremental changes to landforms over a long period of time.

While the focus of landscape grammars in this dissertation is on generative grammars, an analytical application of a landscape grammar in planning is to assess whether singular development proposals conform to a landscape character definition. This is equivalent to determining whether a design is producible from the grammar, which requires that the components of the design conform to the classes and rules in the grammar. If they do conform, then the desired visual character (as embodied by the grammar) has been achieved. Theoretically, this determination would involve searching the solution space of the grammar until the subject design is found, but given the infinite number of possible designs this is not a feasible approach. Another approach is to run the landscape grammar backwards to see if it could iteratively deconstruct the proposed design into the pre-development conditions of the original site. While this might be possible in some of the simple rewriting grammars mentioned in Chapter 2, the complexity of operations in landscape grammars renders impractical the assessment of individual

development proposals. Hence, a special analytical landscape grammar would have to be built that is designed to deconstruct a landscape scene relative to a vocabulary and spatial rules.

The use of landscape grammars is thus relevant to different kinds of landscapes as well as planning situations. This section has outlined several areas of planning activities to which landscape grammars, and the scenes generated by them, are applicable. It can be seen that the uses to which a landscape grammar is put influence the content and form of the grammar. The following section proposes that, conversely, the use of landscape grammars in planning potentially influences the nature of planning as it is practiced.

5.4 Implications for Planning

The utilization of a landscape grammar in planning activities has some implications for approaches to planning. Some of these relate to advances in computer-based visualization capabilities and others to the role of knowledge in the planning process.

The ability to simulate and visualize specific landscapes is within the realm of planners using currently available GIS and CAD technology. One of the advantages of computer-based modelling and visualization techniques is that, through the more efficient production of landscape models, they can allow the consideration of more development scenarios than is otherwise possible in a reasonable time frame. Buildings or other structures may be constructed, inserted and removed from a digital landscape simulation more easily than by non-computational means. The implication of this for planning is that more scenario-based activities are feasible than otherwise. Extending this capability, the 3D visualization of landscape grammars allows even further capabilities in the generation of visual landscape simulations. In addition, because landscape grammars construct scenes according to generalized definitions of landscape entities and their spatial relationships, they can more easily allow an activity that has generally not been available to planning agencies, that is, the experimental examination of landscape character definitions.

It is conceivable that advanced form-modelling capabilities in the planning field could contribute to focusing a greater emphasis on landscape planning, at the expense of the more traditional domain of land use planning. The advanced 3D modelling software currently available is capable of displaying highly realistic simulations of landscapes. The allure and use of such technology may lead planners to devise regulations that focus more on form, because they are now able to assess built form in a more 'sophisticated' manner. Such a shift in emphasis would, however, be fallacious. Planners are no better equipped to do landscape planning with a photo-realistic virtual reality landscape model than with a basic wireframe representation or perhaps even a 2D sketch. In order to plan for visual landscape form,

knowledge of the character and composition of the local landscape is essential. Realistic landscape models that are not based on such knowledge may be no more than sophisticated facades. Hence, a shift may be required in planning from relying solely on the use of 2D and 3D spatial models to the incorporation of the representation and testing of local knowledge.

One consequence of the landscape grammar approach is the consideration of the potential landscape impacts of planning regulations. In acknowledging the impacts of plan-writing on the local landscape, planners are forced to recognize themselves as agents of change and to visualize the potential impacts of a plan in the same manner as those of singular development proposals. This perspective has not been feasible with conventional landscape modelling technologies because their adoption of space- and entity-based (grid- and vector-based) landscape representations does not accommodate the modelling of generalities such as those embodied in most planning regulations. The use of GIS and CAD in planning is highly amenable to the inventorying of objects in a landscape, but planning regulations more commonly refer to generalizations rather than individual objects. In order to treat a plan as a source of landscape change and to then be able to visualize its impacts, the landscape representation technology must allow for the generalizations about objects in addition to the recording of specific instances of the objects. This perspective is allowed by the incorporation of classes and rules in a grammatical representation of landscape character.

Finally, the landscape grammar approach explicitly assumes one of the critiques of the rationalist planning model, namely, incomplete knowledge of landscape phenomena. This is incorporated into the grammar model by the decision to use a particular level of abstraction and by the continual evaluation and refinement of the grammar over time. The grammar approach also affirms the immense magnitude of outcomes that are possible. It recognizes that a landscape character must be embodied by more than one scene within a region and consequently that the visualization of a landscape grammar requires the generation of more than one simulation. Because these scenes will portray the modelled character to greater or lesser extents, some latitude must be allowed in their evaluation.

These acknowledgements of imperfection and multiple outcomes force planners using the grammar formalism into a process of outcome exploration, instead of outcome prediction. They also encourage the incorporation of a role for learning in the planning process. The 2D or 3D landscape model that may typically be relied on as a predicted future is exploded to let the planner articulate the influences both at the syntactic level with spatial relationships and at the semantic level with deeper meanings of such patterns. The articulation of these factors forces deeper probing into the planner's understanding of the landscape and a process of learning that is made obvious by the acknowledgement of grammar imperfection. The landscape grammar can be made to be more complete, incorporate finer landscape details and less abstraction, or account for a larger array of anomalies and thus become

applicable to a wider range of landscapes. The essence of this process is the ongoing improvement of the planners' knowledge of the local landscape character. The grammatical approach to planning adopts a process of knowledge improvement rather than the use of regulatory actions based on the assumption of complete knowledge.

5.5 Summary

The theoretical constructs for landscape grammars must be implemented not only as a computational tool but also in the context of the day-to-day work of planning agencies. Methods for the construction of a landscape grammar have been presented including various methods for identifying the components of a landscape character, articulating them as an executable grammar, and then evaluating the grammar's outputs. Several planning applications of landscape grammars were suggested, as well as potential implications for the planning profession of incorporating grammar use into planning practice.

Now that the application of landscape grammars to a planning context has been discussed, attention turns in the following chapters to the application of landscape grammars to a specific case study. The landscape grammar concepts and LGS implementation are applied to the island of Bermuda to generate 2D and 3D Bermudian residential landscapes for sample study areas. The following chapter presents the general Bermuda landscape characteristics, an overview of Bermuda's planning environment, and two specific residential study areas. Chapter 7 then applies the landscape grammar system to the study areas first to duplicate existing scenes and then to simulate new development.

Chapter 6

The Bermuda Visual Landscape

“What are the peculiarities which make Bermudian scenery not merely beautiful but *different* from other beautiful scenery and worth crossing half the world to see because of that difference? What are the peculiarities which, if found in the South Sea Islands or the Mediterranean, would at once remind a discriminating traveler of Bermuda, and lead him to say that at last he had found another spot in the world almost matching the kind of charm he had found in Bermuda.”

Frederick Law Olmsted (1936, 89-90)

The implementation of a landscape grammar interpreter such as that discussed in Chapter 4 demonstrates that landscape grammar concepts can be accomplished in principle and in contrived examples. Methods are available for putting landscape grammars to use as shown in Chapter 5. The application of landscape grammars to a real-world context will demonstrate the applicability of the theory proposed in this dissertation to real-world planning needs and visual landscape management. The application of the implementation to a particular region also helps to identify potential modifications to grammatical structure and mechanisms, as well as the strengths and weaknesses of a grammatical approach to landscape planning.

Accordingly, this chapter introduces the study area in which the grammar theory is applied, namely the island of Bermuda. Following a description of Bermuda’s visual resources, the value of and threats to these resources are identified. The vulnerability of Bermuda’s land resource has led to careful land and development planning on the island. Bermuda’s regulatory planning environment, with particular reference to its focus on visual landscape character, is described. While the island’s landscape can be seen to have relatively smaller areas with different characters, the application of landscape grammars for the purposes of this dissertation is focused on the Bermudian residential landscape. In particular, a residential area is selected for grammatical modelling. Geographic data available from the island and used in the modelling process are also discussed.

6.1 The Landscape of Bermuda

Bermuda comprises a small group of islands in the western Atlantic Ocean, approximately 670

miles east of Cape Hatteras, North Carolina, United States, which is the nearest point of land. While not geographically in the Caribbean, this British colony is culturally, socially, and environmentally similar to that region. The islands are located on the top and to the southern end of the Bermuda sea mount, an inactive volcanic pedestal capped with limestone and reefs reaching approximately 10 miles in diameter. Coral reefs are found to the north of the islands and less than a mile off the south shore. There are seven main islands and over 200 small islands, forming a chain approximately 35 km long and 2.5 km wide. The main islands are clustered together and connected by bridges, and are referred to by residents as the Island of Bermuda (Figure 6.1). The island's total land area is approximately 54 km² (5,392 hectares), approximately 6% of which comprises developable land from former foreign military bases that closed in 1996. The remaining land is primarily built up and quite densely occupied.

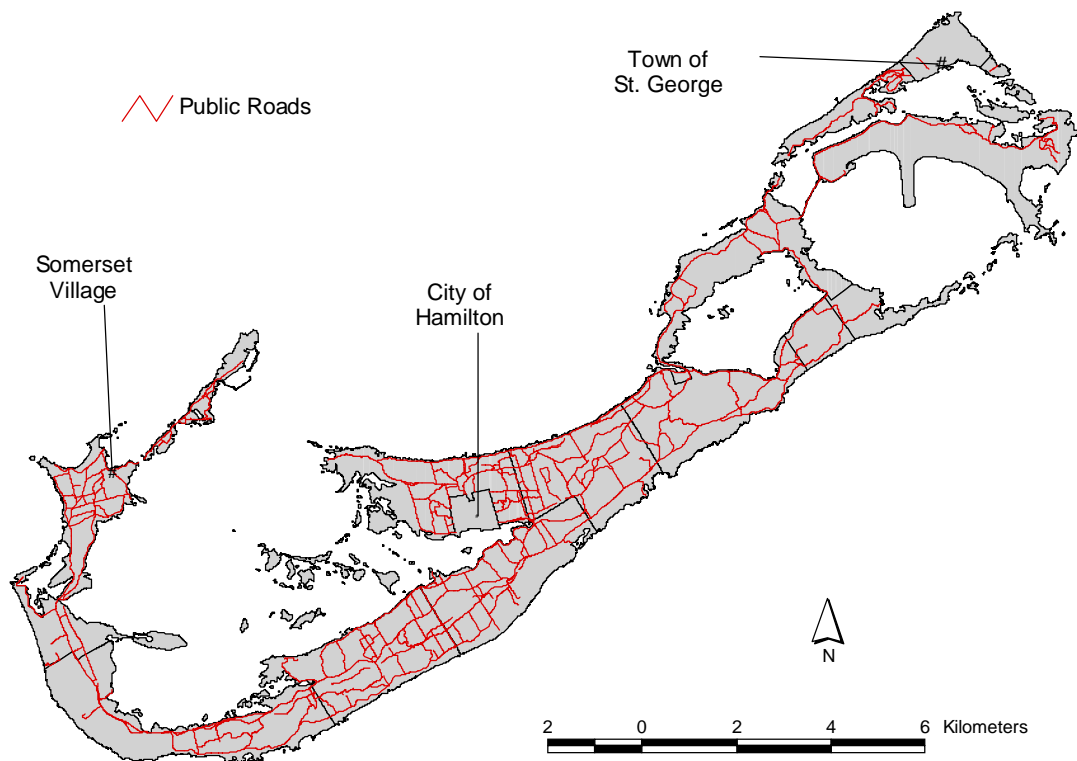


FIGURE 6.1 MAIN COMMERCIAL CENTRES AND ROAD NETWORK OF BERMUDA

6.1.1 Natural Features

The island is volcanic in origin topped with a marine limestone cap and, as noted above, surrounded by coral reefs. The topography of Bermuda is the result of the accumulation of dunes, and

the evolution of karst features. The land surface mainly consists of rolling hills, the highest of which is 79 metres above sea level. The shoreline is mainly rocky, with beaches found primarily on the south-western coast and intermittently elsewhere. Due to the island's small size and the porosity of the limestone base, there are no rivers or freshwater lakes and the soils are generally shallow (Hayward & Rowlinson, 1981). The average depth of soil throughout Bermuda is only about 15 cm, ranging from almost nothing on elevated areas to over a metre in lowland tracts. The climate is sub-tropical and frost-free, as temperatures rarely rise above 32°C in summer or fall below 16°C in winter. The relatively high humidity (an annual average of 79%) is tempered by frequent on-shore breezes. The warm climate is attributable to the island's location on the edge of the Gulf Stream (which also allows Bermuda's coral reefs to survive). As a result, the vegetative land cover of Bermuda is similar to those of more southerly latitudes of the mainland United States.

Despite the scarcity of soil on the island, the warm temperatures, high humidity and well-distributed rainfall allow a lush growth of varied vegetation, including several varieties of tropical and subtropical trees, wherever there is enough topsoil. Because there is no extended dry season, the landscape is always green and will usually return to dense forest if left undisturbed (Hayward & Rowlinson, 1981). Plants from tropical and temperate climates grow and thrive in Bermuda. It is estimated that there are nearly 1,500 species of trees and plants on the island (Phillips-Watlington, 1996; Sterrer, 1998). Before settlement, Bermuda's forests were dense with cedars, palms and other native and endemic species. During the past 400 years of colonization, many more species have been introduced, now contributing an estimated 85-90% of plant species. Bermuda's current woodlands are mostly comprised of evergreens, with some semi-deciduous species and exhibit a wide range of foliage and color. Much of the original flora has now been replaced by agricultural crops, pasture, gardens and lawns. The majority of the island is considered developed. There are few large natural expanses and many of those are recreation fields. The remaining significant natural areas are typically protected wooded hillsides, long narrow strips of rocky shoreline, the dune areas of the beaches of the south shore, some ponds and mangrove swamps. Figure 6.2 shows a mosaic of aerial photographs of Bermuda. The level of development is readily apparent from the photos, once it is understood that Bermuda's roofs are painted white.

There is currently no classification map that delineates the natural habitats of Bermuda, although there are two main references that attempt to identify them. Thomas (1998) identified ten main habitat types: Beach/Dune, Rocky Coastal, Upland Coastal Hillside, Upland Hillside, Upland Valley, Peat Marsh, Fresh/Brackish Pond, Upland Limestone Karst, Salt-Marsh or Salt/Brackish Pond, and Tidal Mangrove Swamp. This classification omits any settled areas of the landscape, as it is biased towards natural areas, especially freshwater and marine environments. The botanical field guide by Phillips-Watlington (1996)

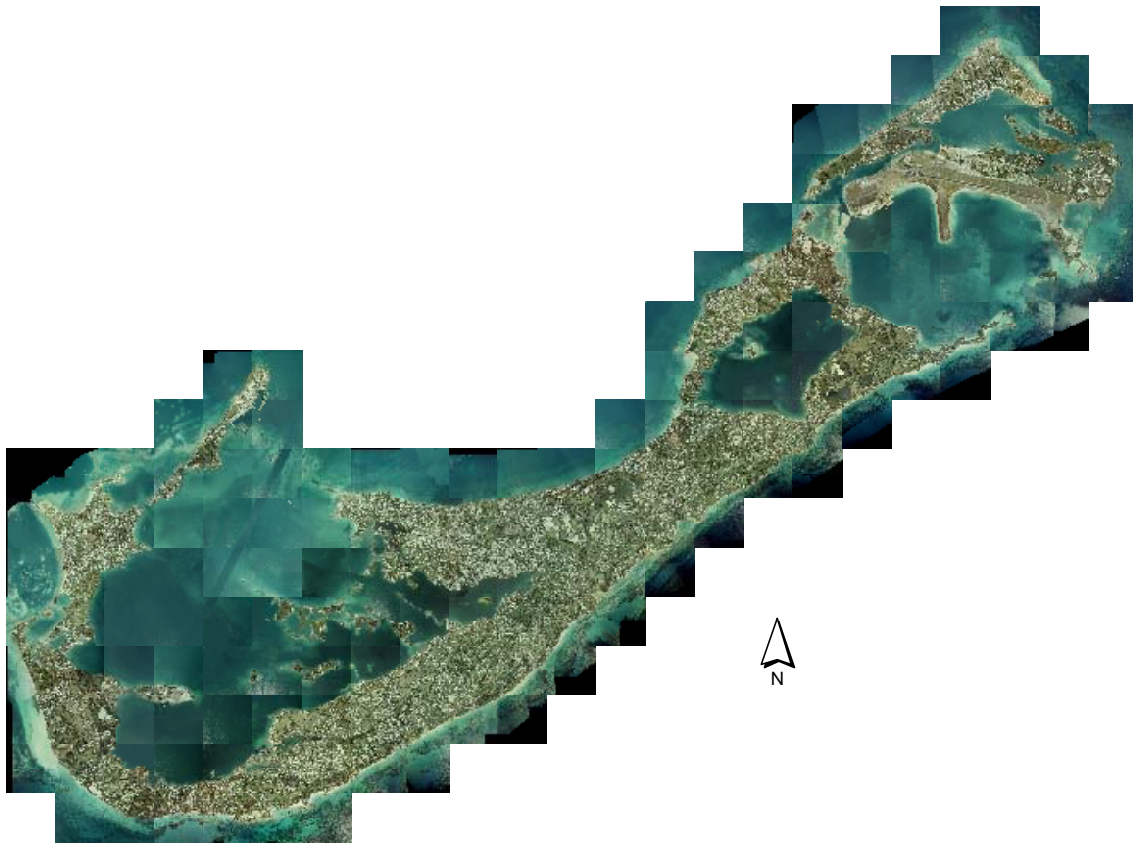


FIGURE 6.2 AERIAL PHOTOMOSAIC OF BERMUDA

also mentions some habitats, likely from the notes of long-time Bermuda Government Conservation Officer, Dr. David Wingate. This list of thirteen habitat types is more inclusive of human intervention: Shallow Sandy Bays, Brackish Marsh, Coastal Hillside/Rocks, Mangrove Swamps/Tidal Lagoons, Beach/Dune, Upland Hillside, Upland Valley, Freshwater Wetland/Ponds/Marshes, Arable Land/Meadows, Gardens, Limestone Sinks, Hedgerow/Wayside/Rocky Outcrops, Islands. In both references, the habitat descriptions are based on the species of vegetation found there, with little or no reference to physical or spatial characteristics that might serve as a basis for the spatial classification of the island's land mass. From 1997–2000, the Bermuda Biodiversity Project performed terrestrial vegetation surveys at 1200 sites across the island, but the data have not yet been fully analyzed (Bermuda Biodiversity Project, 2001, pers. comm.).

6.1.2 Cultural Features

Bermuda's current socioeconomic situation is characterized by high population density with high levels of wealth. The current resident population is estimated at 60,000 persons, yielding a population

density of over 1,110 persons per square kilometre (11.1 persons/hectare). Although Bermuda has a traditionally tourism-based economy, the international finance industry became the dominant contributor to the economy in the 1990s. This change in the economy demanded an influx of high net-worth professionals and is attributed with increases in the foreign workforce and the currently high demand for housing. The 1998 mean employment income was US\$41,322 (Bermuda Department of Statistics, 1999).

The island's high level of development in recent decades is commonly attributed to these high levels of population and wealth, as well as the finite limitations of the land resource. Of the island's 4,884 hectares of land (before military base closures), 3,084 (63%) are zoned in main development areas (Figure 6.3; Table 6.1). There are 597 km of paved highways, four sea ports and one airfield serving the island. The City of Hamilton, Town of St. George and Somerset Village are the significant business and retail centres, with minor commercial intersections scattered about the island (Figure 6.1). Roads are characterized as narrow and winding, typically with two lanes (approximately 8 metres) or less. The three primary roads, North Shore Road, Middle Road and South Road, together span the length of the island, with hundreds of secondary and tertiary roads branching from them.

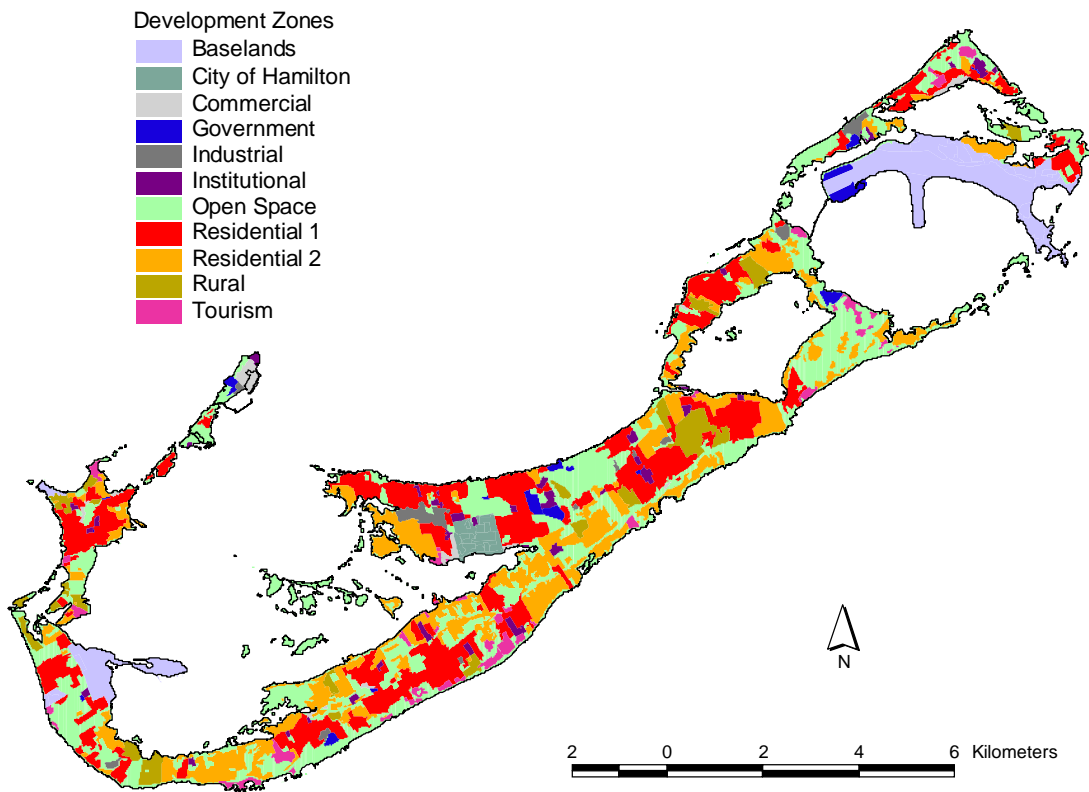


FIGURE 6.3 DEVELOPMENT ZONES FROM THE BERMUDA PLAN 1992

While Bermuda has no comprehensive record of land use, the development zonings of the land are often employed as an indicator of use. The island is divided into mutually exclusive development zones and conservation zones are overlaid on specific areas. Table 6.1 shows the composition of the island in terms of its development and conservation zoning. The most significant uses include open space and residential (Residential 1, Residential 2, and Rural zones), with the latter comprising 43% of the total land area. Conservation areas account for 2,337 hectares (43%) of the land mass, of which 1,816 hectares (78%) occur within designated Open Space areas. The spatial polarization of land for development and land for conservation is illustrated by the significant proportion of conservation zones situated within Open Space areas. The conservation areas situated within the other development zones are fragmented pockets of agricultural land and woodland.

Development Zones	Area (hectares)	% of total land mass
Open Space	1,800	33%
Residential 1	1,271	24%
Residential 2	1,036	19%
Rural	254	5%
Tourism	143	3%
Institutional	101	2%
Government	83	2%
Industry	72	1%
Commercial	53	1%
City of Hamilton	71	1%
Former Military Baselands	508	9%
<i>Total</i>	<i>5,392</i>	<i>100%</i>

Conservation Zones	Area (hectares)	% of total	% of total land mass	Area in Open Space	Area in other dev. zones
Green Space	659	28%	12%	535	124
Woodland Reserve	399	17%	7%	268	131
Recreation	340	15%	6%	295	45
National Parks	330	14%	6%	330	0
Agricultural Land	282	12%	5%	143	139
Nature Reserve	177	8%	3%	175	2
Woodland	150	6%	3%	68	82
<i>Total</i>	<i>2,337</i>	<i>100%</i>	<i>43%</i>	<i>1,814</i>	<i>523</i>

TABLE 6.1 ZONING COMPOSITION OF BERMUDA UNDER THE BERMUDA PLAN 1992

It is difficult to characterize the parcelation of the island, since there is no central land parcel registry in Bermuda. The first subdivision of the island is exhibited in the Savage Map of 1650-1660, which shows the land divided into long narrow parcels extending from the north to south shores. While

these parcels were successively subdivided over the next centuries, the shapes of fences, hedges, walls, paths and roads in some parts of the island, as shown in current topographic mapping, still delineate parts of that original subdivision (Figure 6.4). Such lines also help to estimate current parcel boundaries. It is estimated by the Bermuda Government that there are now approximately 18,000-20,000 parcels of land (Bermuda Ministry of Works & Engineering, 2000, pers. comm.). Estimates from local planners suggest that in high density areas, the sizes of most residential lots fall within the range 0.1 – 0.2 hectares ($\frac{1}{4}$ - $\frac{1}{2}$ acre), while in lower density areas the range is more typically 0.3 – 0.6 hectares ($\frac{3}{4}$ – $1\frac{1}{2}$ acres) (Bermuda Department of Planning, 2000, pers. comm.).

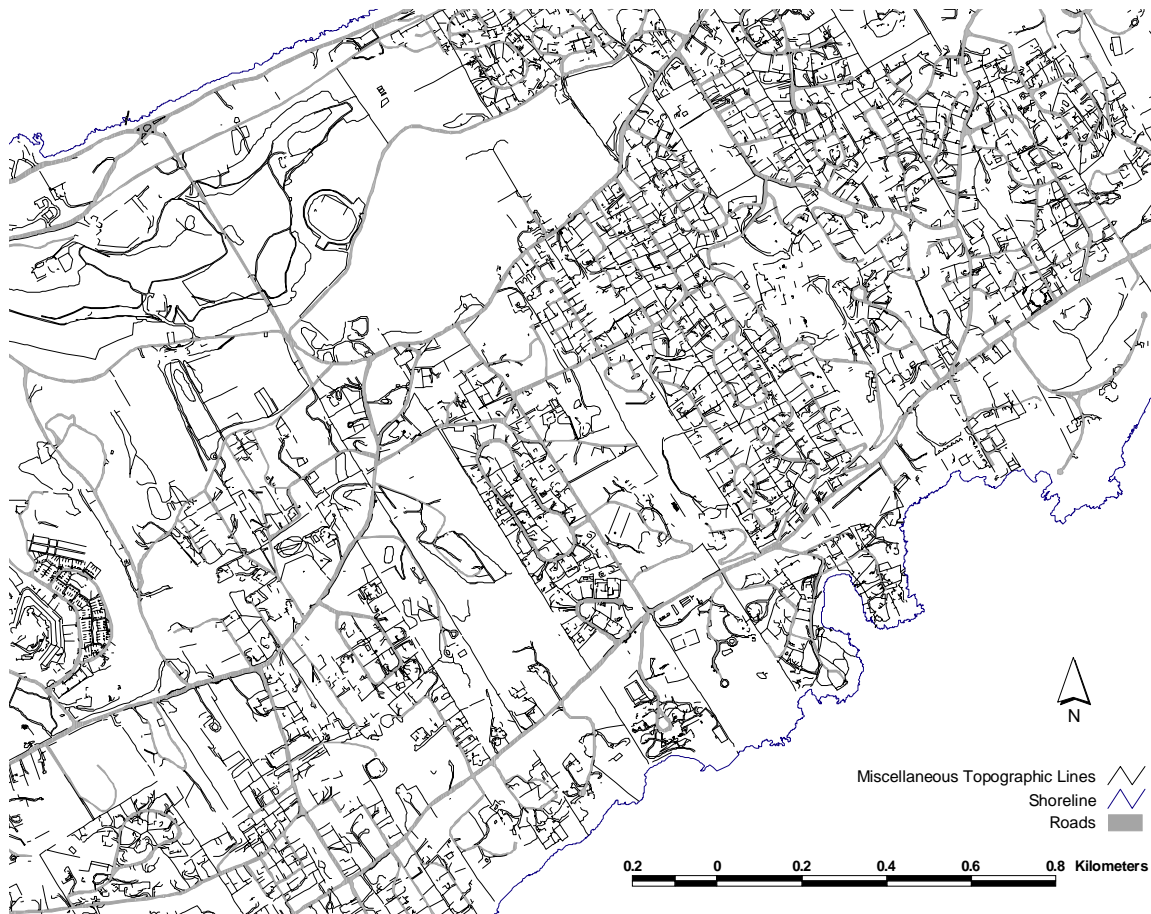


FIGURE 6.4 “MISCELLANEOUS LINES” FROM BERMUDA TOPOGRAPHIC MAPPING

A core feature of Bermuda’s landscape character is small-scale development. Outside of the City of Hamilton, the largest buildings are the hotels and institutional buildings. Commercial buildings that occur outside the main centres are small enough to be comparable in size to medium or large homes. The majority of buildings in the Bermuda landscape, however, are residential. Of the island’s 28,450

buildings, 80% occur in either Residential 1, Residential 2 or Rural areas. Table 6.2 shows that a majority of the housing stock is in the form of one- or two-unit cottages. It is an increasingly common trend for Bermudian homes to incorporate a one-bedroom or studio apartment. As of August 2000, there were 27,584 dwelling units in Bermuda (Bermuda Land Valuation Office, 2000), creating a housing density of 5.6 dwelling units/hectare (excluding the former military bases).

Dwelling Unit Type	Number (1991)	% of Total (1991)
Single family cottage	6,764	31%
Two-unit cottage	7,952	36%
Three-unit dwelling	3,236	15%
Apartment dwelling	3,655	16%
Other	454	2%
<i>Total</i>	<i>22,061</i>	<i>100%</i>

TABLE 6.2 COMPOSITION OF DWELLING UNITS BY TYPE
(Source: Bermuda Census Office, 1992)

Bermuda has a vernacular style of architecture which is widely applied to residential and even commercial structures. It is comparable to colonial architectural style but modified due to local resource conditions. Traditionally built of local limestone (referred to locally as “Bermuda stone”), homes are now constructed of concrete block and plastered. Roofs are constructed on a timber frame using overlapping rows of limestone tiles (“Bermuda slate”), which are plastered and whitewashed. The building façades are adorned with pastel colours, shuttered windows, and wide external chimneys. Less frequent, but typically Bermudian, features include ‘welcoming arm’ staircases, plaster ‘eyebrows’ above the windows and scalloped gable-ends. Many of these and other traditional features have been in existence since stone buildings first became common in Bermuda in the mid-seventeenth century. Treatises of traditional Bermuda architecture are provided by Raine (1989), Humphreys (1923) and the Bermuda National Trust (1995, 1998, 2000).

Despite the island’s high density, the visual character of the landscape still retains what is referred to as a ‘rural’ experience. This is primarily due to the small scale of site development and the proliferation of landscaping and natural vegetation growth. While Bermudian gardens may typically feature structures such as low stone walls, wooden pergolas, Chinese moongates, and mock butteries, the most dominant garden feature is plentiful vegetation, including large shade-trees, small fruit trees, boundary hedges, and bushes fringing houses and walls. Large garden trees often include Royal Poinciana, Pride of India, and various species of Ficus for shade, while Casuarina, Bermuda Cedar, Norfolk Island Pine, Frangipani and several varieties of palm are more decorative. Tall fruit trees include

loquat and avocado, while banana, papaya and various citrus species comprise the smaller fruit trees. There are many types of hedges used, including Hibiscus, Oleander, Match-Me-If-You-Can, Natal Plum, Pittosporum, and Surinam Cherry. There are various species of vines, ferns and smaller plants that are not considered in this dissertation.

The description of Bermuda's landscape character is augmented by stating what is not present: plazas or shopping malls with large parking areas; large or illuminated signs; traffic lights (except in the City of Hamilton); mass-developed neighbourhoods with identical floor plans; series of lots with open grass lawns; wooden buildings; and municipally maintained residential sidewalks.

6.2 The Pertinence of Landscape Planning to Bermuda

Bermuda's visual landscape character has both cultural and economic significance for the island's community. Because it has a strong historical presence, the local landscape character has served for several centuries as a symbol of cultural identity. The local architectural style is particularly treasured because of its local origins and continuity through centuries of development. In 1984, when asked to identify the best aspects of living in Bermuda, residents ranked the natural beauty of the island as second only to the friendliness of its people (Gurr, 1984, as cited in Bermuda Department of Planning, 1990). Hence the value placed on local landscape character by members of the community reinforces the cultural need for careful consideration of visual landscape character.

Like Bermudians, tourists also favour Bermuda's high landscape quality. Visitor surveys show consistently that most tourists make favourable comments about the island's high aesthetic quality and unspoiled environment (Bermuda Department of Tourism, 1996-1999). Bermuda's economy has for decades been founded upon a successful tourism industry, which has been credited for the high standard of living enjoyed by Bermudians. Since the tourism industry is greatly dependent on pressured visual resources and since the local economy is likewise dependent on tourism, there is a critical economic need to maintain the quality of the island's visual landscape resources.

The visual resource, particularly the aspect of a rural landscape character and unspoiled natural areas, is under constant pressure for change from a growing population and economy. In a 1990 survey of public attitudes toward visual quality in Bermuda, residents emphasized the existence of these pressures in their replies to how the island's appearance had changed (Figure 6.5). Because of the cultural and economic values placed on the island's landscape character, the increasing pressures on a finite land area necessitate careful planning measures. It is this consideration that emphasizes the relevance of visual landscape planning to Bermuda.

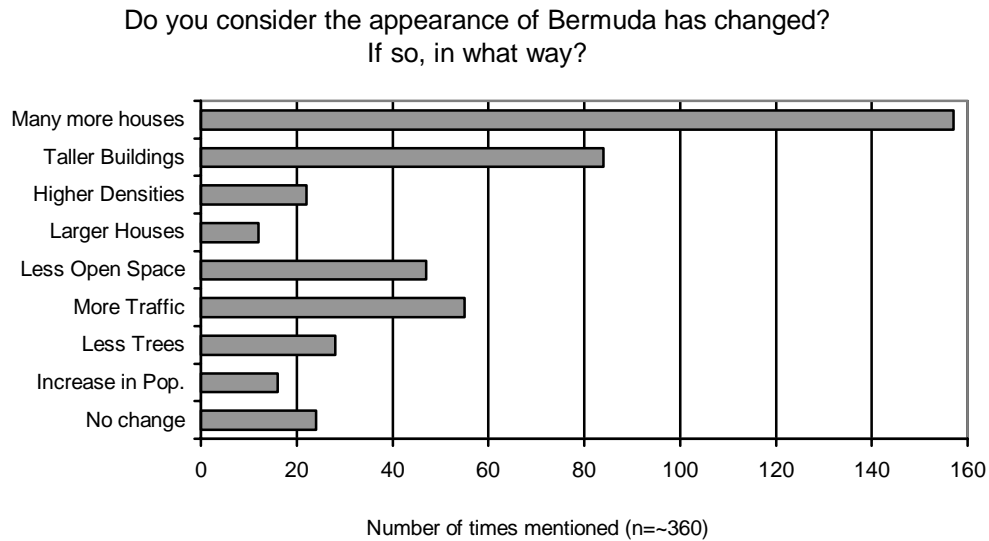


FIGURE 6.5 RESIDENTS' IDENTIFICATION OF CHANGES IN BERMUDA'S APPEARANCE
(Source: Bermuda Department of Planning, 1990)

6.3 The Planning Environment of Bermuda

6.3.1 Bermuda's Planning History

Since Bermuda's physical development space has always been limited, resource management activities have been necessary since the initial settlement in the early seventeenth century. Among the island's first Parliamentary acts in 1620, was an act "against the killing of young tortoises", which is widely quoted as the world's earliest recorded conservation legislation. Apart from legislation to enact specific public works projects, other early acts called for residents to mark parcel boundaries with trees and hedges, the construction of a church in each parish, reforestation measures and the protection of areas of common land (Raine, 1989). Further measures to protect natural resources followed in the subsequent centuries. The first example of town planning in Bermuda was the 1790 commission to consolidate land for what was to become the City of Hamilton (Rowlinson, 2000).

Modern statutory planning emerged in its present form in the mid-twentieth century (Figure 6.6). The first comprehensive building standards were introduced as the Building and Land Development (Control) Rules, 1948. The Building Authority that was established also had jurisdiction over certain planning issues, such as determining whether a proposed building "would cause undue detriment to the amenities of the neighbourhood or to the beauty of prospects or views" (Rowlinson, 2000). In 1962, a Government-commissioned report entitled "The Next 20 Years" by H. Thornley Dyer, became the first

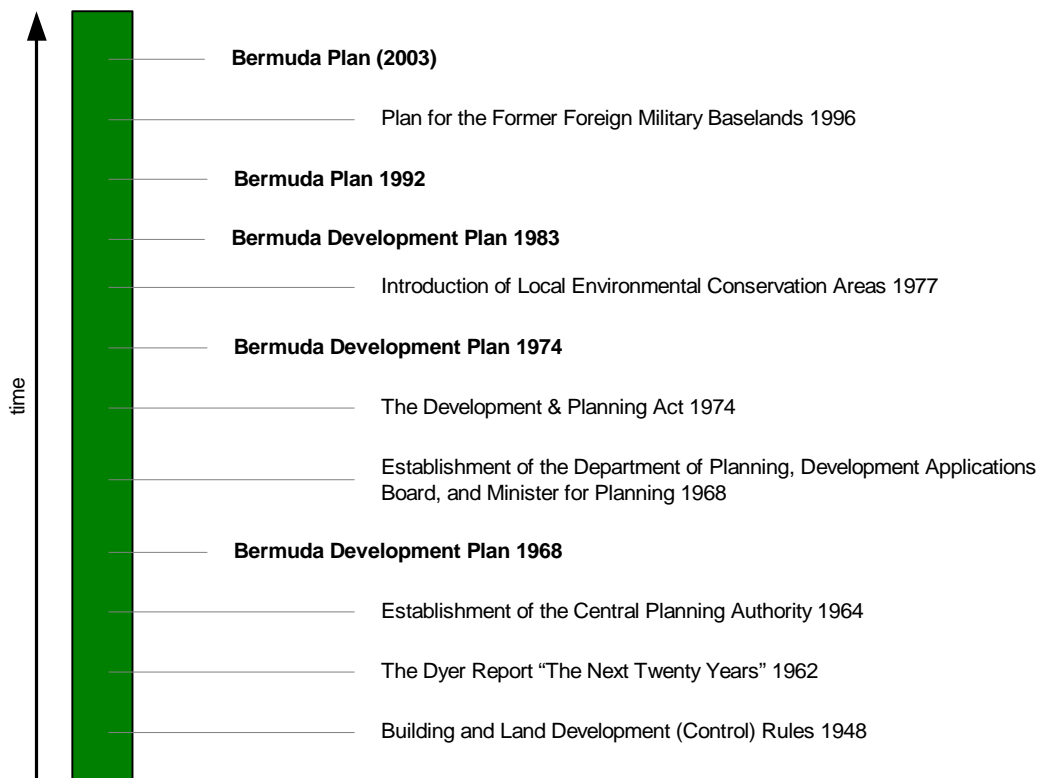


FIGURE 6.6 SIGNIFICANT RECENT EVENTS IN BERMUDA’S PLANNING HISTORY

directive to future development across the island, leading to the formation of a Central Planning Authority in 1964, the Development and Planning Act 1965 and the basis for Bermuda’s Development Plan 1968 (Rowlinson, 2000). As a result of the adoption of the Bermuda Constitution of 1968, the Central Planning Authority became the Planning Department, serving a Development Applications Board that determined planning applications and the Government’s Member for Planning responsible for policy development. These bodies still exist as the current planning administrative structure.

The first Bermuda Development Plan of 1968 was compiled quickly without comprehensive survey and analysis (Rowlinson, 2000). It used a form of zoning, identifying areas which allowed respectively 3 houses per acre and 1 house per acre. The Bermuda Development Plan 1974 was the first proper plan involving extensive surveys, analysis, forecasts and public participation. It used a “broad-brush” zoning technique, only identifying areas for high and low density, along with cottage colonies and adopted a focus on the visual quality of the Bermuda landscape. Local “Environmental Conservation Areas” (ECAs) were later developed, following public outcry over the huge growth in development from 1974 – 1982. The ECAs proved ineffective in controlling the spread of development into adjoining natural areas.

Following public pressure to undertake comprehensive survey of the dwindling open spaces and agricultural land, the Bermuda Development Plan of 1983 provided the first effort to delineate comprehensively the island's environmental resources. In addition to the detailed types of conservation areas it produced, the plan provided more types of development zones, segmenting residential areas into High Density, Medium Density, Garden District, and Rural zonings. The importance of conservation areas and the pressure for release of further developable land were central influences on the current Bermuda Plan (1992), which is discussed in the next section.

6.3.2 The Current Planning Environment

The current Bermuda Department of Planning falls under the responsibility of the Ministry and Minister of the Environment. The Department's essential responsibilities are the control of development, the maintenance of construction standards and the protection of the island's natural and community resources through the development of policies and plans. Accordingly, the Department is divided into three sections: Development Control, Building Control and Forward Planning. The policies and procedures under which the Department operates are specified in the Bermuda Plan 1992 (in conjunction with the City of Hamilton Plan 2000 and the Plan for the Former Military Bases 1996) and the Bermuda Building Codes of 1998.

The central tenet of all of Bermuda's plans since 1968 has been the control, rather than the promotion, of development (with a few exceptions, such as the encouragement of residential development in the City of Hamilton to reduce traffic congestion elsewhere on the island). Increasingly, the meaning of 'control' has evolved from the maintenance of orderly development, to the identification of the most acceptable means for restricting further development. Not surprisingly, to address this goal, the Bermuda Plan of 1992 adopted a central philosophy of "sustainability" and "sustainable development". Since the latest Government was elected in 1998, the guiding principle of "sustainable development" has become a doctrine and has been applied to other domains including the tourism, international business and immigration.

The Forward Planning section's overall approach to planning is a combination of deliberative rationalism and communicative practice (as defined in Alexander, 1998). Draft plans have typically been formulated by forward planners, through a process of survey and analysis, followed by in-house deliberation on the initial options. The ideas are presented to organizations and the general public for comment and the draft plan is revised in consideration of such feedback. As the island has become more developed and planning conflicts more intense, planning methods have more recently involved communicative action earlier in the process.

The products of the planning process, namely the Plans, reflect a combination of British and

North American approaches to planning. The use of zoning was adopted from the practice of North America, while examples from British town planning can be found in Bermudian legislation, such as the Development and Planning Act's Section 34 and Zoning Order Agreements (voluntary agreements between land-owners and planning authorities), Tree Preservation Orders, and the listing of historic buildings, among others. However, Bermudian plans are distinct from many of their overseas counterparts in their strict requirements on details of built form and visual appearance. This is undoubtedly a result of the scale at which planning is performed and the increasing pressure on limited land resources.

The Bermuda Development Plan 1983 effectively introduced what may be considered "status quo" planning in Bermuda. At that time, natural and developed areas were inventoried and delineated. Since then, pressure for further development has become more intense and environmental resources are afforded increasingly more protection. As a result, the Bermuda Plan 1992, while reclassifying residential zonings, offered few major modifications to zoning boundaries. The name of the 1992 Plan intentionally dropped the word 'development', in order to reflect the increasing importance of conservation.

The next Bermuda Plan is likely to come into effect in 2003. An increasing spatial polarization between developed and natural land is expected to become more pronounced, producing more rigid and more precisely placed zoning boundaries. The intensification of developed land is the most likely approach to future development, which will place strain on local concepts of the traditional Bermuda landscape character.

6.3.3 Landscape Planning in Bermuda

Planning activities regarding landscape character can be seen in the early development of the current Bermuda Plan 1992, from the study of visual quality to the definition of a "Bermuda Image" and then subsequently in the details of development control standards in the 1992 Planning Statement. In 1990, the Department conducted a survey to address specifically "public attitudes to visual quality in Bermuda" (Bermuda Department of Planning, 1990). The objectives of the survey were to "(i) identify the principal visual elements which contribute to aesthetic quality in Bermuda; (ii) isolate those factors which detract from aesthetic quality in Bermuda; and (iii) provide guidance in formulating planning policy aimed at protecting, enhancing and improving visual and environmental quality" (Bermuda Department of Planning, 1990). The survey results identified not only the least and most attractive locations on the island, but also features of the landscape considered to be particularly important to Bermuda's visual quality.

In addition to environmental management, the 1992 Plan also further emphasized visual amenity and its implications for tourism and the quality of life for residents. "Bermuda 2000: Facing the Future"

(Bermuda Department of Planning, 1991) introduced the policies and proposals of the 1992 Plan. Of the eighty-eight policies in that document, at least forty contain implications directed towards visual amenity, typically using phrases such as “well-designed in harmony with the physical characteristics of the site”, “display[ing] a character which is in sympathy with the island’s architectural traditions” and “programmes to improve the appearance, amenities and character of particular residential neighbourhoods” (Bermuda Department of Planning, 1991). Many policies refer to “high quality environments”, “visual importance”, “landscaping” and “architectural interest”. Several others have indirect positive implications for visual quality.

While the Planning Department has not attempted to define Bermuda’s landscape character in a grammatical manner, an articulated concept, termed the “Bermuda Image”, has been central to past and present Plans. The Bermuda Image is a statement of the visually appealing elements that make up the island's unique character. The concept has been assigned particular importance within the regulatory framework, as evidenced in the directions to the Development Applications Board (DAB):

“The Board shall apply the details of planning provisions and other relevant provisions of the Statement in a manner which -

...

(c) ensures that all development is sensitive to, and compatible with, the Bermuda Image.”

(Section 3.1, Bermuda Department of Planning, 1992)

The definition of the Bermuda Image has incorporated considerably more detail since it was first introduced in the 1974 Plan, when it was included as a glossary item (Figure 6.7). The 1983 Plan moved it into the main body of the document and itemized its elements in a list. Under this plan, the Development Control section found itself under criticism from the architectural community, alleging that the requested revisions to their designs were based on subjective and inconsistent interpretations of the Plan’s vague, but pivotal, definition of the Bermuda landscape character. In response, the results of the 1990 visual quality survey provided a basis for a more detailed definition of the Bermuda Image in the Bermuda Plan 1992.

The Bermuda Image concept lends credence to other planning policies and regulations. Woodlands, open space, agricultural land, mangroves, shorelines, and small islands are all under different forms of protection in the Bermuda Plan and all are valued as parts of Bermuda’s visual resources. Local architecture, including that of large, commercial buildings in the City of Hamilton, has to retain those qualities and features which are deemed to characterize ‘traditional’ Bermudian styles. There are thus numerous planning initiatives to maintain the visual quality of Bermuda in terms of natural and man-made elements, with special reference to the traditional landscape character, the definition of which has

‘The Bermuda Image’ means the visual appearance of Bermuda resulting from the unique combination of profuse sub-tropical vegetation, colourful houses with their white rooftops and distinctive architectural character, vividly coloured waters, pink beaches, winding narrow roads and lanes, stone walls, hedges, rock cuttings, and generally diminutive scale.

(Bermuda Department of Planning, 1974)

“the visual appearance of Bermuda resulting from the combined presence of factors including, but not limited to –

- (a) plentiful sub-tropical vegetation
- (b) hedges, stone walls, rock cuttings and narrow, winding roads;
- (c) houses with white roofs and of otherwise distinctive architecture;
- (d) a generally diminutive scale;”

(Bermuda Department of Planning, 1983)

“The Bermuda Image” means the appearance of Bermuda resulting from a harmonious mix of natural features and man-made elements which produce a visual quality and a character of development which are distinctively Bermudian, and which includes –

- (a) a scale of building which is compatible with the landform and which sits comfortably in its setting;
- (b) the balance and proportions of the traditional building form as exemplified in sturdy residential structures with white pitched roofs, and features and embellishments which distinguish local architecture;
- (c) plentiful lush and colourful sub-tropical vegetation;
- (d) gently rolling hillsides and dense vegetation which effectively blend to screen development and to maintain the illusion of open space and a natural appearance;
- (e) Bermuda stone walls, weathered rock cuts, hedging and planting alongside roads; and
- (f) natural coves, bays, the rocky coastline and islands, with views and glimpses of vividly coloured waters and the ocean.

(Bermuda Department of Planning, 1992)

FIGURE 6.7 THE EVOLUTION OF THE DEFINITION OF THE “BERMUDA IMAGE”

been elaborated over time.

Consideration of the Bermuda Image in development control is performed in relation to the “details of planning” and the current development standards as defined in the 1992 Planning Statement. The former are defined in Section 3 of the Planning Statement (see Appendix C) and include all of the relevant factors which the DAB may consider in evaluating a development proposal. In addition to this list of details, development standards exist for each of the development zones, principally defining the maxima and minima for different quantitative factors. Table 6.3 and Table 6.4 show the development standards for Residential 1 and Residential 2 zones, as these are important in shaping a basic grammar for expression of the Bermuda Image in terms of lot layout and building location.

	Detached House	Attached House	Apartment House
<i>Maximum density:</i>			
Houses per acre	6	12	N/A
Units per acre	N/A	12	20
Minimum lot size	6,000 sq. ft.	3,500 sq. ft.	6,000 sq. ft.
Maximum site coverage	35%	35%	35%
<i>Minimum setbacks from:</i>			
Public road/Railway Trail	25 feet	25 feet	25 feet
Estate road	20 feet	20 feet	20 feet
Lot line	10 feet	Discretion of the Board	10 feet
Maximum height	2 storeys	2 storeys	5 storeys

TABLE 6.3 REGULATIONS FOR DEVELOPMENT IN RESIDENTIAL 1 DEVELOPMENT ZONE.
Source: The Bermuda Plan 1992 Planning Statement.

	Detached House	Attached House	Apartment House
<i>Maximum density:</i>			
Houses per acre	2	3	N/A
Units per acre	N/A	3	6
Minimum lot size	18,000 sq. ft.	12,000 sq. ft.	18,000 sq. ft.
Maximum site coverage	20%	20%	20%
<i>Minimum setbacks from:</i>			
Public road/Railway Trail	30 feet	30 feet	30 feet
Estate road	25 feet	25 feet	25 feet
Lot line	15 feet	Discretion of the Board	15 feet
Maximum height	2 storeys	2 storeys	2 storeys

TABLE 6.4 REGULATIONS FOR DEVELOPMENT IN RESIDENTIAL 2 DEVELOPMENT ZONE
Source: The Bermuda Plan 1992 Planning Statement.

6.3.4 Adoption of Landscape Planning Technologies

During the 1990's, considerable funds were invested in the development of GIS technologies and databases within the Government of Bermuda. In 1998, topographic map data for the entire island, digitized at a scale of 1:2500, became available for use. In the fall of 2000, detailed digital orthophotography, at a spatial resolution of 20cm, also became available. Other digitized map resources now include the zoning maps of the Bermuda Plan 1992, geo-referenced civic addresses, census district boundaries, postal code districts, and water wells. Of all the agencies in the Government corporate structure, the Department of Planning, particularly its Forward Planning section, is expected to make the most use of these products as well as generating its own derivative resources according to planning needs.

The integration of landscape planning technologies with existing Department activities has accelerated considerably in recent years. In October 1999, the Department launched a custom-built, centralized database system, termed the Bermuda Environmental Management Information System (BEMIS). It represented the first integration of GIS technology with business systems in Bermuda. Currently, virtually all activities of the Development and Building Control sections utilize BEMIS.

The development of GIS technology in the Bermuda Government has not yet reached a mature state. Data products now exist, as discussed above. The Planning Department's BEMIS is being followed by two other GIS-enabled business application projects from other departments. However, the use of GIS for applications other than simple queries is limited to small, ad hoc efforts in the Forward Planning section. No major achievements have been made in spatial or surface analysis, or models for transportation, population or ecosystem health. The use of 3D computer-based visual simulations is rare and usually conducted by the architect for major land development projects at the request of the DAB.

Table 6.5 lists the spatial data sets that are currently in digital format in Bermuda. The data are categorized by thematic content and information is provided regarding the spatial object type, scale of digitization, data maintenance, the original database and/or authority responsible for the data. Each data set covers the entire island. The Topographic Map Database is published in ArcInfo export format and the Bermuda Zoning Database in the public domain ESRI shapefile format. In addition, the aerial photographs are stored in GeoTIFF and GeoJPEG format, census and Government parcel boundaries as ArcInfo coverages and addresses and wells in shapefile format. For the purposes of this dissertation, all map data were converted to shapefiles and the aerial photography were used as GeoJPEG files.

6.4 Suitability of the Bermuda Residential Landscape for Landscape Grammar Use

The preceding sections have established that Bermuda is a community in which limited land resources and significant growth have led to strict planning measures and that those measures have included considerable references to the traditional landscape character of the island. As a result, Bermuda provides a suitable case for the analysis of landscape character and visual resource planning. It has also been shown that Bermuda's planning authorities have begun to address the definition of their local landscape character. With increasing development pressures, the Department has had to examine more closely the Bermuda Image concept. The 1992 Bermuda Plan illustrates how recent efforts have brought more detail to the concept's definition. This interest in the details of landscape quality/character definition provides a favourable base from which to investigate the landscape syntax of the area and, from this, to develop a Bermudian landscape grammar. The strict planning environment is also beneficial for the study of landscape grammars. Since Bermuda does not engage in the practice of considering

Theme	Items	Format	Scale	Maintenance Dates (last)	Source
Buildings	Building footprints	Polygon	1:2500	Updated weekly (May 2000)	TMD (W&E)
Roads	Road edges	Polygon	1:2500	Updated weekly (May 2000)	TMD (W&E)
	Road centrelines	Line	1:2500	Updated weekly (May 2000)	TMD (W&E)
Cultivation	Cultivated areas	Polygon	1:2500	Every five years (1994)	TMD (W&E)
	Agricultural conservation areas	Polygon	1:5000	Every five years (1994)	BZD (DP)
Topography	Contour lines	Line	1:2500	Every five years (1994)	TMD (W&E)
Shoreline	Shoreline	Line	1:2500	As required (1994)	TMD (W&E)
Water features	Swimming pools, ponds, canals	Polygon	1:2500	As required (1994)	TMD (W&E)
Vegetative areas	Mangroves, marshes, woodlands, scrub	Polygon	1:2500	Every five years (1994)	TMD (W&E)
	Unsurveyed vegetation symbols	Point	1:2500	Every five years (1994)	TMD (W&E)
Survey control	Surveying benchmarks	Point	1:2500	As required (1994)	TMD (W&E)
Administrative units	Parish, city and town boundaries	Polygon	1:2500	As required (1994)	TMD (W&E)
Land forms	Cliffs, slopes, rock, sand	Polygon	1:2500	Every five years (1994)	TMD (W&E)
Miscellaneous edges	Fences, walls, hedges, driveways, patios, kerbs	Line	1:2500	Every five years (1994)	TMD (W&E)
Zoning and conservation features	Development zones	Polygon	1:5000	As required (1996)	BZD (DP)
	Conservation zones	Polygon	1:5000	As required (1992)	BZD (DP)
	Protection areas (Historic, Cave, Water)	Polygon	1:5000	As required (1992)	BZD (DP)
	Railway Trail	Line	1:5000	As required (1992)	BZD (DP)
	Tribe Roads	Line	1:5000	As required (1992)	BZD (DP)
Aerial ortho-photography	Terrestrial aerial photographs	Tile	161 tiles at 20cm pixel resolution	Every five years (April 1997)	W&E
Census districts	(without census data)	Polygon	1:2500	As required (1998)	W&E
Legal boundaries	Government-owned land parcels	Polygon	1:2500	As required (June 2000)	W&E
Addresses	Address points	Point	1:2500	Updated weekly (August 2000)	W&E
Hydrology	Wells	Point	1:2500	Not maintained (January 2000)	MoE Hydro, DP
	Estimated groundwater lenses	Polygon	1:5000	As required (1992)	BZD (DP)

Sources: TMD – Topographic Map Database; BZD – Bermuda Zoning Database; W&E – Ministry of Works & Engineering, Lands, Buildings, & Surveys Division; DP – Department of Planning; MoE Hydro – Ministry of Environment, Hydrogeology Section

TABLE 6.5 SPATIAL DATA AVAILABILITY FOR BERMUDA

changes to the zoning of land during the life of a plan (through amendments), the regulatory influences on the Bermuda landscape are more controlled and easier to account for in a landscape grammar study than in North American communities which allow such zoning changes.

It is important to note that Bermuda's landscape character is not homogeneous across the island. There are distinctive areas of the island that exhibit their own unique character. For example, the Royal Naval Dockyards on the western tip of Bermuda exhibit large buildings used previously for naval shipbuilding; parcels in the City of Hamilton are typically developed to their maximum extent; and the Town of St. George in the eastern end of the island was established before development pervaded the rest of the island and most of its architecture dates from the 17th-19th centuries. Outside these commercial centres, there are particular parts of the island that have been preserved in a rural state, as well as large tracts used as golf courses. These areas are typically large and not broken into many smaller parcels. They are thus less likely to exhibit the recurrent landscape features that contribute to a pervasive character.

Conversely, residential landscapes are present in virtually all areas (Table 6.1, Figure 6.2, Figure 6.3). There are some differences in residential character, such as between the consolidated structures of condominium complexes, the close but separate higher density cottage areas and the larger and greener lower density properties. However, many visual similarities exist across the island's residential neighbourhoods. Given the relatively high development densities, and since the architectural character of the built residential environment is central to the island's image both at home and abroad, it was decided to focus attention on developing a Bermudian landscape grammar only for residential development. Supporting this choice, Raine (1989) comments that:

“... even in the midst of such changes [in the availability of new materials and methods], the basic designs of the majority of Bermudian homes have resisted the swings into modernity which one finds elsewhere in the World. In Bermuda, each home has essentially retained its individuality and the features which were made fashionable during the previous three centuries are still firmly embedded in contemporary tastes”.

Thus, the Bermudian residential landscape character is particularly amenable to grammatical modelling, due to its traditional landscape composition that has remained relatively static over time and a high degree of consistency in its application.

The availability of digital geographic data for Bermuda, at a large scale, was also deemed beneficial as an information base from which to perform spatial analyses in the development of a landscape vocabulary and grammar rules. A further consideration on the choice of study area was that the author is a native Bermudian and therefore highly familiar with the landscape, allowing some

grammar development away from the island. For these reasons, the Bermuda residential landscape character was selected for the development of a demonstrative landscape grammar.

6.5 Selection of a Case Study Neighbourhood

Rather than attempting to develop a grammar that is universally applicable across Bermuda, a particular residential neighbourhood was selected as a test area to develop a 'Bermudian' grammar. This allowed automation of the generation of scenes using syntactical rules for object placement and definition. The resultant grammatically generated landscape scenes can then be evaluated visually for similarity with the existing site.

Site selection criteria were identified in order to choose an area that would be particularly amenable to grammar development. Clearly, sites with high degrees of regularity are easier to implement in a grammar, since less variability has to be accommodated. Also, it is reasonable to suggest that areas with a zoning designation for high density use (Residential 1) would exhibit this regularity more so than other areas. A rationale for this is that the relatively smaller lot sizes typical of Residential 1 areas would reduce the range of viable development options on any specific lot. Further, these areas are subject to more regulations due to the higher densities of residents. This view is confirmed through discussion with Development Control Planners in the Department of Planning (pers. comm., 2000). Residential 1 areas are also pertinent to the objective of assessing the landscape effects of new planning regulations, since a change in lot development rules would have a more significant impact in high density neighbourhoods than in other areas.

It was further considered expedient to focus on parcels with detached houses, rather than attached houses that overlap a parcel boundary, as the former are more typical of Bermuda's 'cottage' style of landscape. Single detached homes are also less complicated in terms of planning regulations, since they are considered the norm, while attached dwellings have additional planning considerations. Because sites with highly irregular topography must often feature creative design solutions to building placement, form and orientation, regular topography was considered a desirable criterion, especially if parts of the landscape are also flat. The site should also contain relatively few anomalous architectural features. Sites that are wholly contained within one development zone were considered more attractive for grammar development than those encompassing multiple zones. It was also considered beneficial to select sites that had been developed subsequent to the last development plan. This feature isolates the regulatory influences on the landscape to one piece of legislation.

While the use of these criteria may artificially limit the definition of the Bermuda landscape character, it was deemed appropriate for the development and testing of a prototype landscape grammar

using the theory and application developed in this dissertation. However, identifying sites in Bermuda that meet these criteria is not straight-forward. Local planners were surveyed to identify potential sites and it was discovered that most of the recently developed neighbourhoods have condominium complexes, which are not considered typical residential accommodation on the island (although they are a newer kind of development that may be worth future study). Hence, the criterion emphasizing the recency of development was relaxed.

Based on the above considerations, Southcourt Avenue in Paget Parish was identified as a neighbourhood that exhibits features considered beneficial to landscape grammar development. Southcourt Avenue is a relatively straight road that slopes gradually southwards from a hilltop near South Shore Road to the rocky shoreline (Figure 6.8 and Figure 6.9). The development is approximately 40,400m² (4 hectares), zoned Residential 1 and contains a strip of Green Space conservation zone along the shoreline. The fifty-two properties along the length of the road are regularly proportioned by Bermuda standards, suggesting all of the properties and the road were likely created from the original parcel at once. This could not be confirmed as the archives of the Department of Planning and the Bermuda Government contain no records of the subdivision of this land. It was determined that the properties existed during the earliest topographic survey in 1962 and planning officials suspect it was probably created in the 1950s. The lots are not heavily landscaped but the structures exhibit typical Bermuda cottage architecture. The houses are oriented to face the road (an east-west direction) and many also take advantage of the southerly view of the ocean on the adjacent elevation.

While Southcourt Avenue was selected to represent Bermuda's residential landscape character, it may be said to have its own landscape character that differs from or resembles, to varying extents, the other residential landscapes of the island. Southcourt Avenue's characteristics are similar to neighbourhoods elsewhere in Bermuda insofar as they share the general Bermudian residential landscape character. However, Southcourt Avenue also exhibits differing features because it represents a type of residential landscape based on high density and a linear axis near the ocean. Using the terminology of landscape grammars developed in Chapters 2, 3, and 4, this observation may be restated as the expectation that some of the landscape classes and rules in a Southcourt Avenue landscape grammar are shared with neighbourhoods elsewhere, while other classes and rules will be unique to the neighbourhood. By extension, the grammars of various other residential neighbourhoods may be incorporated to create a composite residential landscape grammar that is applicable across the island. This extension is suggested here only theoretically as its feasibility for a dissertation study is questionable due to the need for extended periods of research.

The Southcourt Avenue neighbourhood thus forms the case study in this dissertation for the development of a Bermudian residential landscape grammar. While its linear orientation may not



FIGURE 6.8 AERIAL PHOTOGRAPH OF SOUTHCOURT AVENUE



FIGURE 6.9 TOPOGRAPHIC AND ZONING DETAILS OF SOUTHCOURT AVENUE

necessarily correspond with the general Bermuda residential landscape character, this feature was desirable as a criterion for facilitating landscape grammar development. In other regards, however, the site is typically Bermudian in its visual characteristics. The grammar for Southcourt Avenue presented in the following penultimate chapter forms the basis from which other Bermudian neighbourhood grammars can be developed, gradually moving towards a more sophisticated island-wide residential landscape grammar.

6.6 Summary

A traditional, consistent and locally valued landscape character, as well as a history of planning activities designed to protect that character, make Bermuda a fitting study for landscape grammar development and implementation. The development of digital geographic databases in Bermuda and the use of them in the Department of Planning, is a necessary first step to realizing a useful landscape grammar implementation.

Not only is Bermuda an appropriate location for the evaluation of landscape grammar theory and implementation, but also, conversely, the study is of utility to the planning process in Bermuda. Under a rationalist approach, the study and analysis of landscape character is a prerequisite to planning for it. The concept of landscape grammars may serve as a next step in the evolving definition of the Bermuda Image. It may be a future vehicle for planning authorities to deal with the definition of the local landscape character and visualize the spatial consequences of proposed regulations.

A number of criteria were suggested for selecting a suitable site for landscape grammar development. The selected site, Southcourt Avenue, was subjected to further on-site inspection and subsequent grammatical modelling. The application of the landscape grammar implementation to this site is the subject of the next chapter.

Chapter 7

Construction of a Bermuda Grammar

The previous chapters of this dissertation have presented a theoretical model for landscape grammars, described a computer-based implementation of a landscape grammar interpreter, discussed the context for use of grammars in planning agencies, and identified the Bermuda residential landscape as a context that is amenable to landscape grammar analysis and use. At the end of the previous chapter, the specific neighbourhood of Southcourt Avenue was identified as a case study for grammatical modelling. This chapter provides details of the analysis of Southcourt Avenue, elements of the landscape grammar that was consequently constructed, and outputs of the grammar itself as it relates to the existing landscape and other possible landscapes on the Southcourt Avenue site.

The chapter is divided into six main sections. First, the data collection methods are described in relation to the landscape grammar construction approach proposed in Chapter 5. This describes the site in general terms that can be formalized into a landscape grammar. Next, the components of the grammar are presented. A vocabulary of hierarchical landscape classes and their attributes are described, followed by a discussion of the grammar rules that were developed to reconstruct the Southcourt Avenue landscape in its present character. Discussion of the grammar rules is facilitated by a presentation of the grammatical scene construction that closely resembles Southcourt Avenue, although not replicating it exactly. The translation of a generated scene to a 3D form in GDS software is then demonstrated, followed by a further section that modifies the Southcourt Avenue grammar by introducing planning regulations that generate new landscape scenes. The chapter concludes with a discussion of several issues identified during the grammar development and suggestions for future landscape grammar and interpreter implementations.

7.1 Knowledge Acquisition

The methodology proposed in Chapter 5 was followed in order to acquire landscape knowledge relevant to the Bermudian visual landscape character as embodied in Southcourt Avenue. The first step in this process entailed scoping the objects and patterns of Southcourt Avenue that contribute to its visual character. As noted in Section 5.2.1 of the previous chapter, this scoping process was not linear and involved several iterations of field observation, document review, interview and data analysis. The results of these stages are presented as a set of figures at the end of this section. Due to time and

financial constraints, a fully fledged implementation of the proposed approach to landscape grammar development was not undertaken. Rather, a proof of concept approach was adopted that used the researcher as grammarian.

Several field visits were made to Southcourt Avenue to make visual observations of the landscape objects and patterns there. A photographic survey was performed recording each property as viewed from the road and the photographic records were later used for further off-site observations. Further photographs of the site were obtained from helicopter surveys by the Bermuda Government Department of Planning. Objects and patterns were described initially as natural, cultural or regulatory. The description of cultural features included the size, position, orientation and physical features of buildings, driveways, and garden and retaining walls. The major vegetation, including hedges and trees, was also surveyed with an ecologist (see Table 7.1).

Trees	Hedges
Bermuda Cedar, White Cedar, Poinciana, Brazil Pepper, Clerodendrum, Frangipani, Baygrape, Olivewood, Umbrella Tree	Giant Privet, Viburnum, Oleander, Surinam Cherry, Hibiscus, Pittosporum, Match-Me-If-You-Can, Mock Orange, Casuarina Bush, Brazil Pepper, Bougainvillea, Croton
Palm Trees	Decorative Individual Bushes
Chinese Fan Palm, Palmetto, Cuban Royal Palm, Screw Palm, Dracona Palm, Solitaire Palm, Coconut Palm	Hibiscus, Pittosporum, Agave, Blue Plumbago, Snow Bush,
Fruit Trees	
Lemon, Lime, Orange, Avocado, Loquat, Banana	

TABLE 7.1 VEGETATIVE SPECIES OBSERVED AT SOUTHCOURT AVENUE

A search of local documentation sources was performed to identify any information relevant to the development of Southcourt Avenue. The records of the Department of Planning were searched for evidence of the legal parcel boundaries. Records of the original subdivision of the entire site were not available in the Department's archive of applications for the subdivision of land, indicating that the neighbourhood was originally subdivided pre-1950. In lieu of a comprehensive parcel record, survey plans of parts of Southcourt Avenue were located at local surveying firms and a composite map of the original parcelation of the site was assembled from these documents.

The available survey records and composite parcel map allowed for a differentiation between those parcel boundaries delineated for the original subdivision of the study area and those boundaries that exist currently. Comparison of original to current parcel boundary records showed that since the original subdivision there have been seven amalgamations of parcels, one property subdivision, and one

boundary adjustment in the Southcourt Avenue neighbourhood. Bermuda's Development and Planning Act 1974 and its subsequent amendments define a number of ways in which such changes in the parcel configuration may be accounted for. Primarily, plans of subdivision may be approved and registered by the Department of Planning, or plans that are approved but not registered are also recognized if subsequently conveyed under separate title (Section 35 of the Act).

The Department of Planning issued approval for the subdivision, boundary adjustment and one amalgamation, but records do not exist for the site's other six amalgamations. These amalgamations are allowed by Section 35 of the Act in which adjacent parcels owned by the same person may be considered as a single holding. The Act, through Section 41, also recognizes any parcels existing prior to 26 June 1974, even though they may not conform to current development plan regulations. Therefore, it is most likely the case that some adjacent land parcels on Southcourt Avenue were purchased by a single owner and, at some point prior to the establishment of the Department of Planning, the individual titles were amalgamated into a single land holding.

As identified in Section 6.5 of the previous chapter, the currently operating Bermuda Plan of 1992 administers a Residential 1, or high density residential, zoning to the entire Southcourt Avenue neighbourhood. While there are particular regulations pertaining to Residential 1 zoning (ref. Table 6.3), the majority of development in the study area occurred prior to the influence of the Department of Planning. Alterations and additions to buildings on Southcourt Avenue in the past three decades would have required planning approval. The Department of Planning's records identified twenty-seven development applications since 1989, but most of them were for internal renovations or minor works. It is assumed then that major additions to the buildings on the site occurred in prior years, and the currently available documents therefore do not give an indication of how or when buildings were changed significantly.

The participant interview portion of the scoping exercise was intentionally excluded from the case study in order to devote efforts to the development of the LGS application and assembly of the landscape grammar rules. This decision was made in consideration that the aim of the case study was to demonstrate the mechanics of landscape grammars rather than the complete methodology proposed for planning agencies (Section 5.2). Instead of involving local experts and non-experts, reliance was placed on the author as an observer and expert. This was considered acceptable as the author, a native Bermudian as well as a planner in the Department of Planning, is highly familiar with the Bermuda residential landscape. Input was unofficially sought from other local planners in Bermuda; however this was not performed in a consistent manner.

Data for the Southcourt Avenue neighbourhood were made available in digital form by the Government of Bermuda. The primary data were extracted from the Topographic Mapping Database

maintained by the Ministry of Works & Engineering, as summarized previously in Table 6.5. From these data, secondary data sets were also compiled through analytical methods using conventional GIS software. For example, contour lines were used to construct 1-metre resolution raster data including elevation, slope and aspect values. GIS software tools were also used to calculate distances between sets of objects (such as buildings and the edges of the road), areas and lengths of objects, and to overlay spatial data to look for spatial patterns, such as the location of a building in relation to the elevation values within its parcel boundaries. Ortho-rectified photographs, captured in April 1997 at 20cm resolution, were obtained and used to collect further landscape data that were not readily accessible in the field, for instance, the vegetation and structures in backyards. Aerial photographs of the site were also obtained during informal helicopter surveys of the island conducted by the Government of Bermuda. Yet further data sets were assembled from empirical data collected on-site. These data included the locations and heights of vegetation and garden walls, neither of which were discernible from the two-dimensional map data and ortho-photography.

Colour data were recorded using standard image editing software to extract colour values from digital photographs taken on site. Because of the presence of shadows in the photographs and the variance of colour values over some surfaces, the colours obtained directly by this method were found not to be visually representative of the colours experienced in the landscape. Specifically, the extracted colours were too dark and grey in tone. Consequently, they were adjusted manually until they matched the author's visual interpretation of the colours in the site photographs. The colour data were stored as Hue-Saturation-Lightness (HSL) value triples which were found to be more intuitive to modify than Red-Green-Blue (RGB) triples for the same colours.

The results of these data collection exercises are presented in the following pages. Figure 7.1 and Figure 7.2 display aerial perspectives of the study site from helicopter surveys in 1990 and 2002. Figure 7.3 and Figure 7.4 give a general impression of Southcourt Avenue roadscape taken respectively from the middle of the neighbourhood looking south, and from the southern (shoreline) end of Southcourt Avenue looking north. Excerpts of the photographic survey of Southcourt Avenue properties are presented in Figure 7.5. The images relate the visual characteristics of the properties as viewed from the road. Figure 7.6 presents selected landscape data that were collected and assembled in a GIS database.

7.2 Grammar Construction and Execution

Recall from Chapter 3 that, in order to generate a landscape scene a landscape grammar (LG) must consist of a vocabulary of landscape object-types or classes (V), a set of spatial rules (R), and an

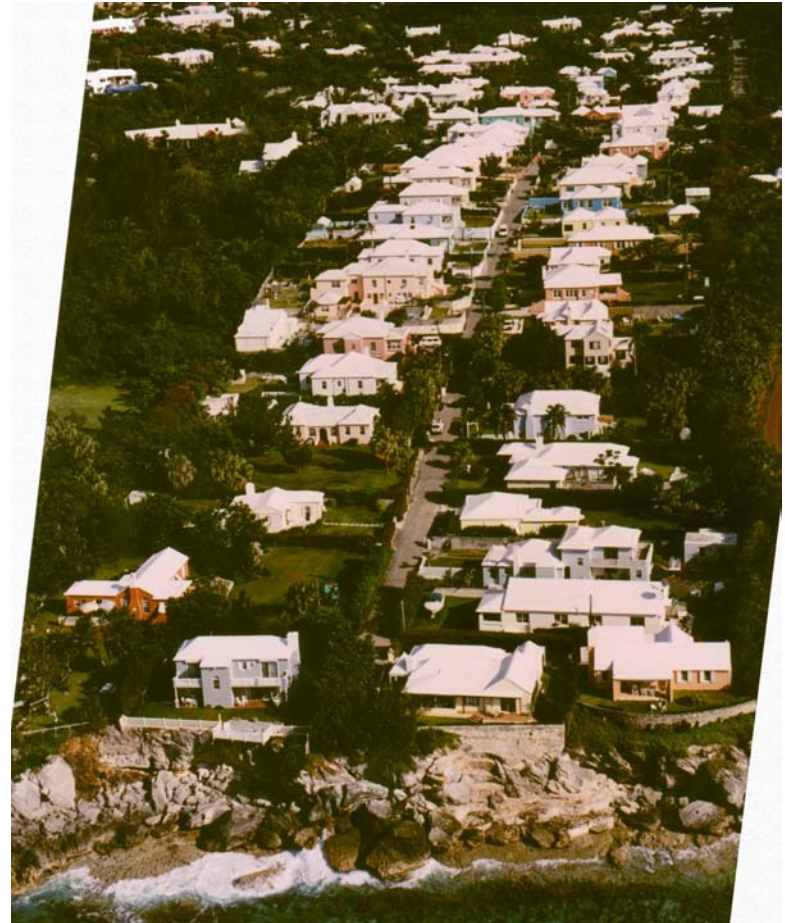
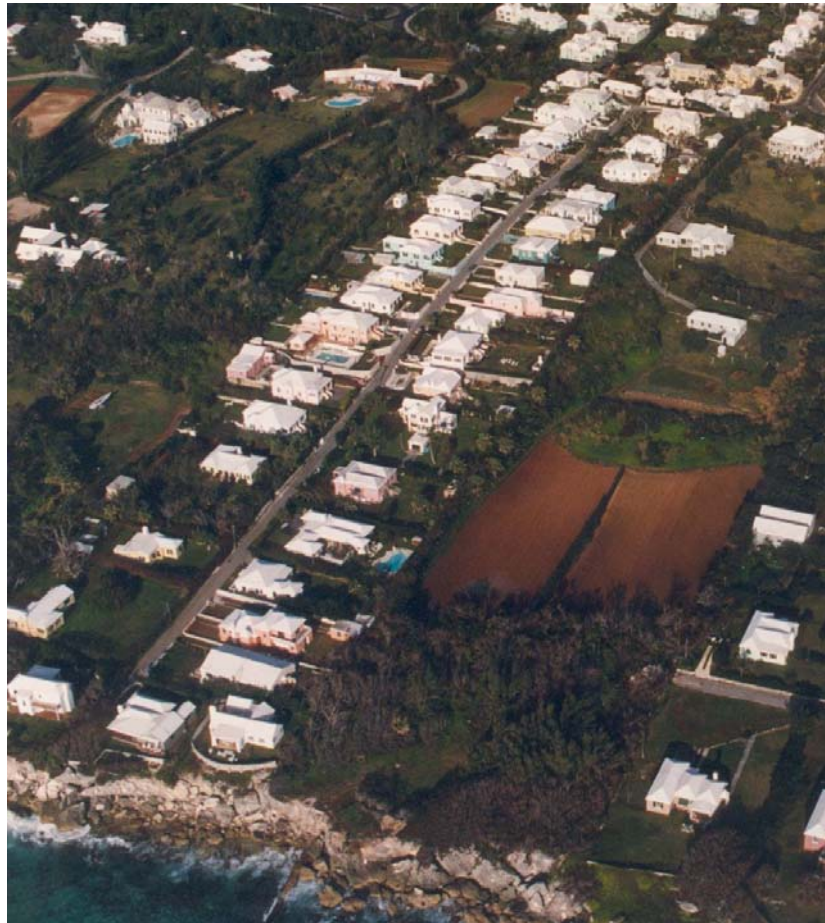


FIGURE 7.1 AERIAL VIEWS OF SOUTHCOURT AVENUE LOOKING NORTHWEST IN (i) 1990 & (ii) 2002



FIGURE 7.2 AERIAL VIEW LOOKING UP SOUTHCOURT AVENUE FROM OCEAN (2002)



FIGURE 7.3 ROADSIDE VIEW OF SOUTHCOURT AVENUE LOOKING NORTH



FIGURE 7.4 ROADSIDE VIEW OF SOUTHCOURT AVENUE LOOKING SOUTH



FIGURE 7.5 PHOTOGRAPHIC SURVEY OF SOUTHCOURT AVENUE PROPERTIES



FIGURE 7.5 PHOTOGRAPHIC SURVEY OF SOUTHCOURT AVENUE PROPERTIES (CONTINUED)

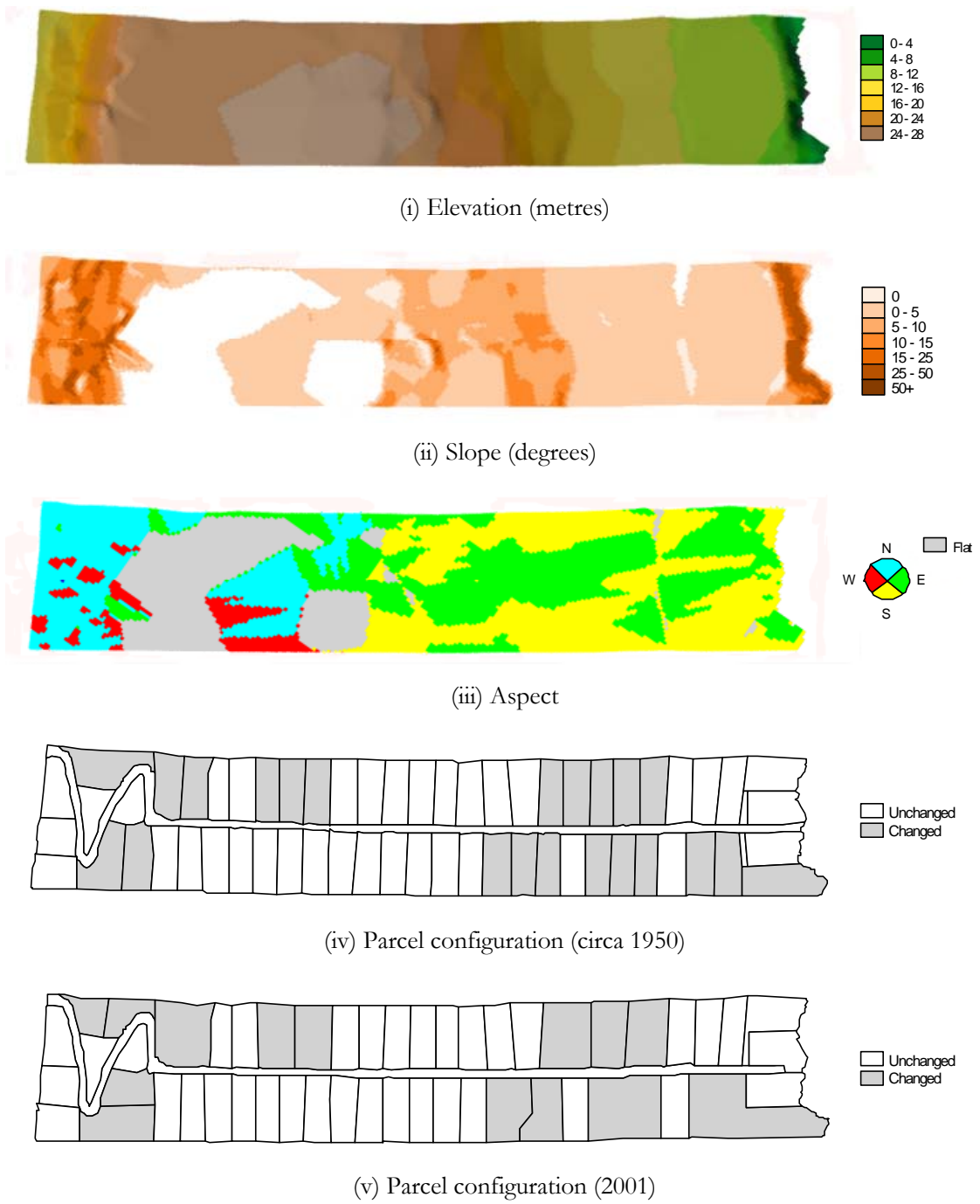


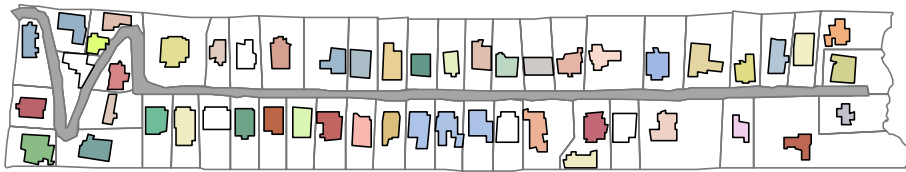
FIGURE 7.6 LANDSCAPE DATA FOR SOUTHCOURT AVENUE



(vi) Ortho-Photography



(vii) Buildings and Driveways



(viii) Houses with Paint Colour



(ix) Major Vegetation



(x) Garden/Retaining Walls (height in feet)

For all maps:

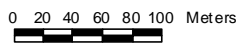


FIGURE 7.6 LANDSCAPE DATA FOR SOUTHCOURT AVENUE (CONTINUED)

initial scene of landscape objects (IS). A vocabulary of landscape classes was assembled for Southcourt Avenue and is presented in the next section. Sets of spatial rules, organized into modular rulesets, were gradually constructed to reproduce the objects and patterns of the neighbourhood. These rules, the initial scene of landscape objects, and the generated scenes are presented in Section 7.4.

With these three grammar elements, (V, R and IS), and the landscape grammar interpreter mechanism available in the LGS software, Southcourt Avenue was reconstructed as near to current form as possible using generalized grammatical rules. Because there are stochastic elements in the rule syntax, executing the grammar several times led to the generation of different grammar-conformant landscape scenes on the site. The Southcourt Avenue vocabulary is presented in the next section, and then the sets of landscape rules and their execution to generate a particular scene are presented in the subsequent section. Alternative completed scenes are subsequently provided and discussed.

The landscape grammar demonstrated here is comprised of twenty-nine vocabulary classes, 150 grammar rules, thirty-five preset materials (or colours; fifty more were generated randomly), and eight global parameters. It is worth noting at the outset that these figures do not necessarily reflect the exact nature of the landscape grammar. A high level of creativity is involved in translating ideas pertaining to the landscape character into the formal geometric mechanisms of the landscape grammar. The grammarian soon recognizes that there are many possible variations in the manner in which objects may be classified in a vocabulary, rules grouped into rulesets, or functions utilized within the rule syntax. The corollary for this observation is that a given landscape scene can be generated using different landscape grammars. One version may use fewer classes but richer attribute definitions on each class. Another grammar may be written with fewer rules, endowing each rule with a powerful string of functions instead of breaking them out into separate and simpler rules. Therefore, the numbers of classes and rules, while presented here for information purposes, are not meaningful descriptions of a landscape grammar.

7.3 Vocabulary Classes

A vocabulary for Southcourt Avenue was compiled from the landscape observations undertaken during the scoping exercises. The vocabulary of twenty-nine landscape classes, shown in Figure 7.7, is not complex, yet it represents the visually prominent and meaningful aspects of the landscape character in Southcourt Avenue based on the interpretation of the researcher. The classes used here are also more generally applicable to Bermuda residential landscapes as a whole than to Southcourt Avenue in particular.

Point landscape classes included only Tree and its subclasses. The attributes of the Tree class, and its subclasses, included a ground elevation and the name of the corresponding 3D tree object in GDS

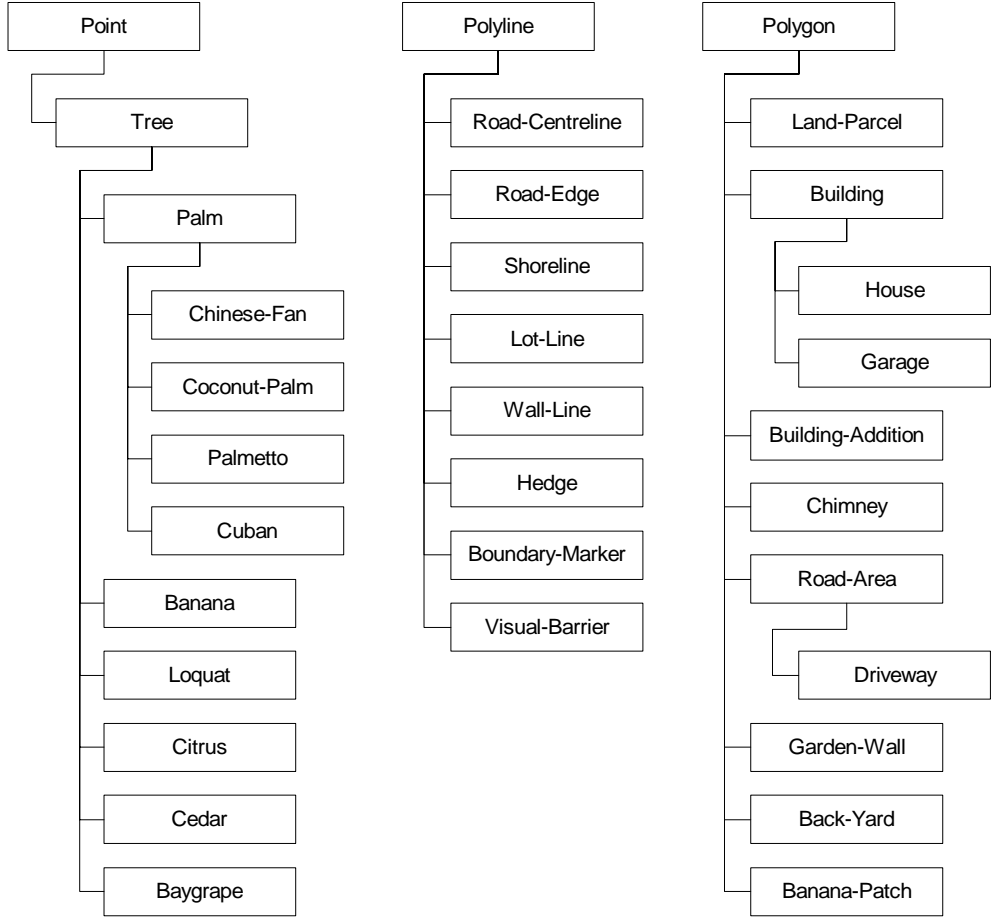


FIGURE 7.7 LANDSCAPE VOCABULARY FOR SOUTHCOURT AVENUE

(other possible attributes, such as age and height, were not used for modelling). Four types of palm tree prevalent in the area are represented as separate subclasses. Banana, Loquat and the generic Citrus subclasses of Tree represented types of fruit trees that are generally found in the side and back yards of Southcourt Avenue residences. Other Tree subtypes included the salt-resistant Baygrape tree which occurs mainly near the shore, and the Cedar which is a highly valued local variety of tree. These nine types of trees are conspicuous in the study site in terms of their size, prevalence and their visual identity. Other tree species were generalized into the three miscellaneous deciduous tree classes.

The Polyline landscape classes include the Road-Centreline, Road-Edge and Shoreline classes. While instances of Road-Edge or Shoreline can be included in the edges of a Land-Parcel polygon (defined below), other non-physical parcel edges are represented by the Lot-Line polyline class. The Wall-Line and Hedge classes are used to represent the linear extent of garden walls (usually four feet or less in height) and hedges (greater than three feet in height). In Southcourt Avenue, walls and hedges

appear to serve the functions of delineating the boundaries of a property, providing privacy from neighbours or the road, or retaining earth where the grading of a parcel had been altered. Boundary-Marker and Visual-Barrier are intermediate, abstract polyline subclasses used by the landscape rules to determine the nature and placement of the Wall-Lines and Hedges.

The Polygon landscape classes represent landscape objects with an areal extent. The Land-Parcel class represents a legal parcel of land. The Building class is divided into House and Garage subclasses. The Building class is assigned attribute definitions for elevation, ground-floor elevation, height, material (which stores the paint colour), as well as attributes relating to the presence, height, slope, and material of a roof. The Building class is also given an attribute, 'front-vector', defined as a vector indicating the direction of the front of the Building. The Building-Addition class characterizes additions to a building's footprint. The Road-Area class embodies the areal extent of the Southcourt Avenue road surface, and its Driveway subclass, smaller Road-Areas located inside the Land-Parcels. The Garden-Wall class corresponds to physical garden walls, generated from Wall-Lines. Back-Yard is another abstract, or non-terminal, class used to define an open area in the rear of a House. Back-Yard objects are used by some rules to place other objects, such as instances of the Citrus tree. Similarly, instances of the Banana-Patch class defined small areas within which Banana tree objects are located.

The classes can be viewed in terms of their natural, cultural and regulatory content. The vegetative landscape classes can be considered natural, but because the area is principally landscaped by its residents, the presence and spatial patterns of hedges and trees may also be considered cultural artifacts. As noted in the previous chapter, Southcourt Avenue was primarily developed prior to institution of Bermuda's planning regulations. Consequently, it is difficult to identify or associate regulatory objects or patterns with the Southcourt Avenue landscape. This is not to say that there are no regulatory spatial rules in its design, rather such rules were likely to have been enforced at the whim of the developer and subsequent landowners rather than the Bermuda Department of Planning and its plans. The vocabulary for Southcourt Avenue is therefore primarily cultural, rather than natural or regulatory in nature.

The Southcourt Avenue landscape vocabulary includes those types of features that are visually prominent in the landscape and excludes those that are not. It includes intermediate classes related to landscape function that are utilized in the Southcourt Avenue landscape rules, while remaining simple enough for it to respect the 3D modelling capabilities and scene limitations of the GDS software. More sophisticated 3D modelling software and a more powerful computing environment than that available to the research would allow for much greater realism and detail, at the expense of considerably more complex grammatical rules. The following section presents the landscape rules for the Southcourt Avenue grammar.

7.4 Grammar Rules

As prescribed in Chapters 3 and 4, the spatial rules for describing Southcourt Avenue were modularized into rulesets consistent with the vocabulary classes outlined in the previous section. As noted earlier, the development of the grammar resulted in a collection of 150 rules organized into nineteen rulesets. Although it is not feasible in this chapter to explain each of these rules in detail, this section follows the interpretation of the grammar rules during the generation of an example landscape scene for Southcourt Avenue. The syntax of some example rules is provided to illustrate their content, whereas other rules are explained as functional groups.

Although various scenes were generated from this landscape grammar, each of these interpretations was begun with an initial scene consisting of only four objects and two grids of topographic values. This initial scene included one Road-Edge object, two Lot-Line objects and one Shoreline object. Together, these four polylines also define the boundary of the Southcourt Avenue study area (Figure 7.8). In addition, a grid of one metre cell resolution was generated comprising elevation values above mean sea level, slope and aspect values (slope values are displayed in Figure 7.8, with lighter shades indicating steeper slopes). In this and all subsequent diagrams, the shoreline is oriented toward the bottom of the display. Red squares in this and subsequent graphics from the LGS Scene View symbolize Node objects (where it is expedient to present the locations of Nodes).

It is also worth noting at this stage that eight global parameters were utilized at various places in the syntax of the rules. Some of these parameters were numeric (e.g. **lot-frontage**, **road-width**, **driveway-width**, **wall-width** and **hedge-width**), while others served as named placeholders for geometric shapes that held relevance for the entire site (**the-site-boundary**, **the-site-axis**, and **the-shoreline-axis**). The numeric parameters served as constants in this grammar, although their values could be modified by a rule at any stage.

Figure 7.9 shows an overview of the organization of rules into their main rulesets. A World ruleset (see Section 4.5) was used as a master, top-level ruleset containing all rules and other subordinate rulesets. While the World ruleset served as an all-encompassing ruleset, it also served two specific purposes. First, it initialized some parameters and labels for use in later rulesets. In this case, the **the-site-boundary**, **the-site-axis**, and **the-shoreline-axis** parameters were set to calculated geometric shapes, and the Road-Edges were labelled *the-main-road* and the length of the shoreline, *the-shore*, so that they could be identified as a whole even when they are split by other objects in later rules.

Second, the World ruleset controlled the sequence in which all rulesets were interpreted. The rulesets in this grammar were thematic and ordered, being fired by the interpreter in the order shown in

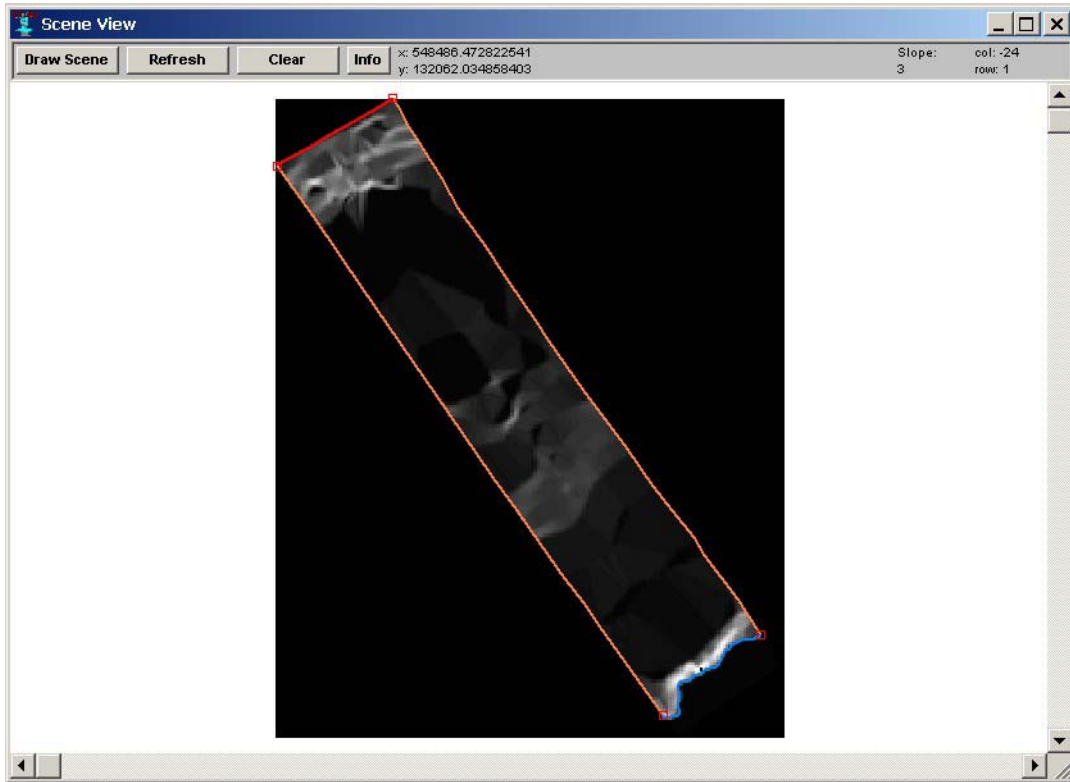


FIGURE 7.8 INITIAL SCENE FOR GENERATION OF SOUTHCOURT AVENUE

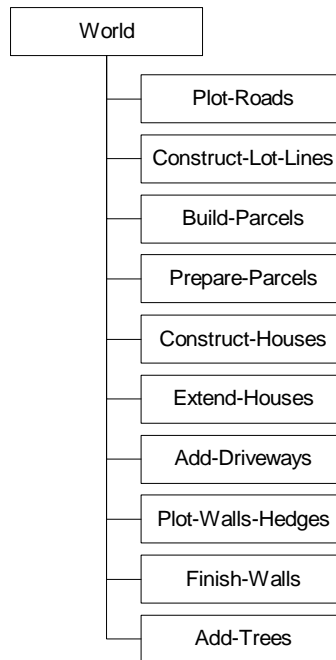


FIGURE 7.9 RULESET ORGANIZATION FOR SOUTHCOURT AVENUE LANDSCAPE GRAMMAR

Figure 7.9 (top to bottom). The structure of the Southcourt Avenue landscape was viewed in this sequence, that is, parcels were designed around the road, houses constructed relative to the parcel configuration, driveways added in relation to the houses, and walls, hedges and trees were those objects that were least influential on other landscape objects. In addition to the structural logic of the landscape, the ordering of rulesets in this manner was essential for developing and testing the grammar. By modularizing the grammar so that it iteratively returned to the World interpretation loop after each ruleset, an intermediate scene could be saved after the completion of each ruleset. During grammar development and testing, the appropriate intermediate scene was loaded for the particular ruleset under consideration. If the rules were not organized in this way, then the grammarian would have to start the interpreter from the initial scene and then wait for the other rulesets to fire before reaching the rules of interest. This process can become very tedious when the number of rules is large.

The rules of the World ruleset were primarily Starting-Rules (introduced in Section 4.5) that initiated a nested interpretation loop with one of the smaller thematic rulesets. Each of the rulesets in Figure 7.9 had a corresponding Starting-Rule in the World ruleset, that began a nested interpretation loop using a subset of scene objects and a thematic ruleset. Most of the landscape construction work was performed within these thematic rulesets. Each of the following six sections describes a set of thematic rules (roads, land parcels, houses, driveways, walls and hedges, and trees) which are also listed in Appendix D. Because the rulesets were fired in sequence, the generation of a working-scene of objects is described at the same time. The process began with the initial scene in Figure 7.8 and the World ruleset.

7.4.1 Roads

Generally speaking, Southcourt Avenue itself bisects the neighbourhood area from South Road (top) to the southern shoreline. At the southern end, the study area reaches its terminus before entering the steep rocky shore. At the northern end, near South Road, it winds in order to make the steep slope traversable. The average width of the road was measured from site data to be approximately 10 metres. The surface of the road is asphalt. Given these observations, the goal of the roads ruleset is to create a central axis as a road path and then progressively modify it according to the topography of the site. The path can then be buffered to create the extent of the road.

The grammar interpreter used one parameter (**road-lane-width**) and eighteen rules organized into three named rulesets to characterize the road of Southcourt Avenue. The firing of the *start-road-ruleset* rule from the World ruleset initiated a nested interpretation loop using the rules of the *road-plotting* ruleset and the initial scene of objects consisting of a *Road-Edge*, two *Lot-Lines* and a *Shoreline*. The *create-road-centreline* rule identified the objects labelled *the-main-road* and *the-shore* and inserted a *Road-Centreline* object between their midpoints, thus providing the central

axis of the site for the parameter **the-site-axis** (Figure 7.10 (i)).

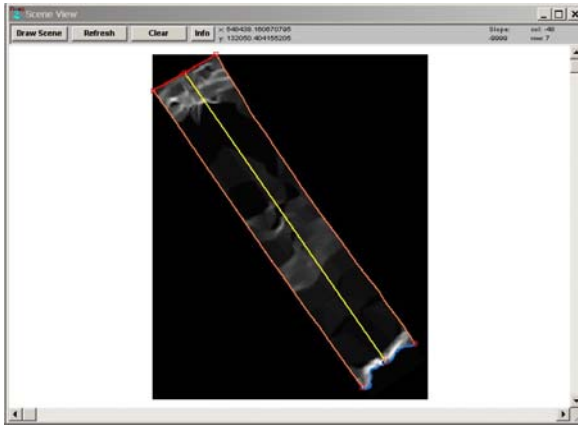
In order to modify the road's path, the Road-Centreline was split into several segments based on changes in slope. The `define-road-centreline-slope-segments` rule iterated repeatedly inside a nested interpretation loop until the Road-Centreline was split into segments with low and high slope values (as calculated from the underlying grid; Figure 7.10 (ii)). The splitting operations may have created segments so short that they should effectively be ignored. As the interpreter cannot choose to ignore objects in the working-scene, these short segments were removed using the `eliminate-small-road-segments-in-middle` and `eliminate-small-road-segments-at-ends` rules inside a loop initiated from a starting rule (Figure 7.10 (iii)). Once the Road-Centreline was cleaned in this way, the segmentation was complete and the grammar processing returned to the main `road-plotting` ruleset.

The `label-steep-road-segments` and `label-gentle-road-segments` rules labelled a Road-Centreline object as either `steep-slope` or `gentle-slope`, depending on whether its average slope value was greater than or less than seventeen degrees respectively.

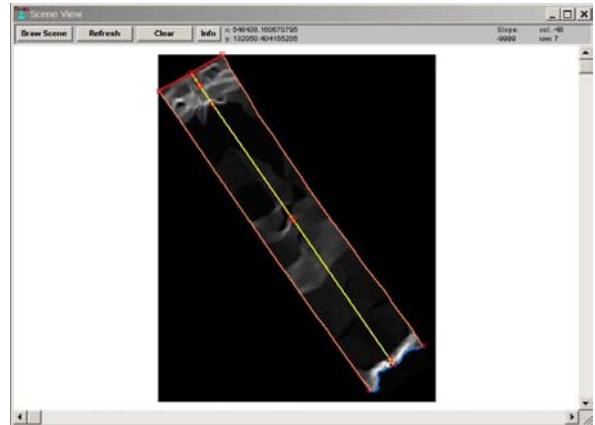
```
(define-rule
  :NAME label-steep-road-segments
  :RULESETS road-plotting
  :IF ((is-a ?a 'Road-Centreline)
       (has-notany-labels ?a '(gentle-slope steep-slope))
       (>= (average-cell-value-on-line ?a 'slope) 17))
  :THEN ((set-label ?a 'steep-slope))
)
```

In addition to the labelling, the Road-Centreline objects in the working-scene were subjected to optional cleaning rules. If a Road-Centreline object was labelled `steep-slope` and connected directly to a Shoreline object, then the `truncate-road-near-shoreline` rule removed that centreline, thus following the observation that a road does not extend into the rocky shoreline of an area. The `remove-roadlines-outside-area` rule removed any Road-Centreline or Road-Edge that sat outside the boundary of the study area as defined by the **the-area** parameter. These four rules could have been fired in any order, that is, the rule selection was random. The `truncate-road-near-shoreline` rule could not effectively fire until the labelling rules had labelled a Road-Centreline as either steep or gently sloping. The `remove-roadlines-outside-area` rule could have been fired at any time during the interpretation process, but at this stage, all Road-Centrelines fell within the study area so the rule did not get fired.

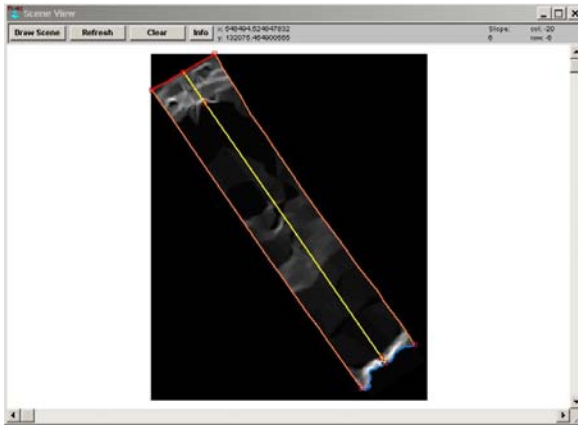
Since Road-Centreline objects had been created, split and labelled as steeply or gently sloping, the road's path was selectively modified to wind over steep slopes. The `steep-road-curves` ruleset was started in a nested interpretation loop and consisted of just one rule. The `curve-road-on-steep-slopes` rule identified any Road-Centreline object that was labelled `steep-slope` and was not curved (by



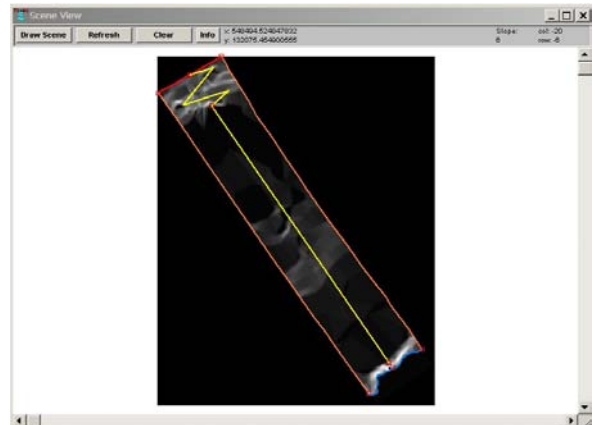
(i) create-road-centrelines



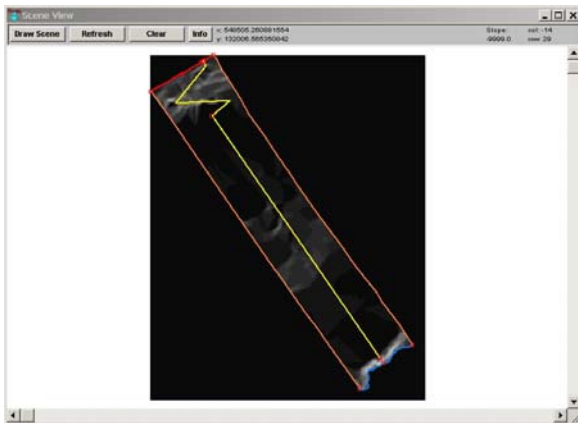
(ii) define-road-centrelines-slope-segments



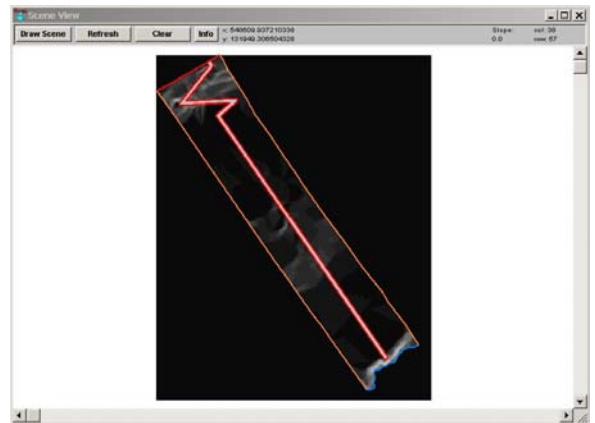
(iii) eliminate-small-road-segments-in-middle,
eliminate-small-road-segments-at-ends



(iv) curve-road-on-steep-slopes



(v) extend-curved-road-on-steep-slopes,
truncate-road-curve-near-entrance



(vi) create-road-edges,
remove-roadlines-outside-area, create-road-area)

FIGURE 7.10 GENERATION OF ROAD FEATURES FOR SOUTHCOURT AVENUE

virtue of a label). The only object matching this description was found at the northern end of Southcourt Avenue. Using the rule's code, the Road-Centreline object was replaced with another whose geometry was calculated as a sine-curve approximation (Figure 7.10 (iv)). Rules of the `modify-road-curves` ruleset were applicable to any curving Road-Centreline object in the scene. The `extend-curved-road-on-steep-slopes` rule modified the geometry to extend the final bend that lies near the straighter, gently sloping road segment (Figure 7.10 (v)). The `truncate-road-curve-near-entrance` rule re-routed any Road-Centreline that had a bend located near (less than ten metres from) the main road. The portion of the Road-Centreline between the main road and the identified curve was replaced with a shortest distance route from the curve to the closest point on the main road (Figure 7.10 (v)). This nested interpretation then completed and returned to the road-plotting ruleset.

At this stage of the scene generation, all of the rules that affected the path of Southcourt Avenue had been exhausted. The `create-road-edges` rule buffered a chain of Road-Centreline objects by the distance contained in the `*road-lane-width*` parameter and created Road-Edge objects from the buffer lines (Figure 7.10 (vi)). As some of these lines may have extended outside the study area (as defined by `*the-area*`), a short loop was initiated to allow the `remove-roadlines-outside-area` rule (identified above) to iteratively fire upon and remove any of these errant Road-Edges. The rule syntax for these operations in the `create-road-edges` rule follows:

```
(define-rule
  :NAME create-road-edges
  :RULESETS road-definition
  :IF ((is-a ?a 'Road-Centreline)
       (is-connected-to-a ?a 'Road-Edge)
       (is-not-labelled ?a 'road-created))
  :THEN ((let ((the-avenue-edge
                (new 'Road-Edge
                    (buffer-completely
                     (path-of-vertices
                      :along-the-lines (instances 'Road-Centreline)
                      :starting-from (from-node ?a))
                      (/ *road-width* 2))))))
          (set-label (instances 'Road-Centreline) 'road-created)
          (set-label the-avenue-edge 'the-avenue-edge)
          (interpret (list (get-rule 'remove-roadlines-outside-area))
                    (instances 'Road-Edge))
          )))
```

Once the Road-Edges were contained to the study area, the `create-road-area` rule was able to be fired creating a Road-Area polygon object from the Road-Edges (Figure 7.10 (vi)).

This concluded the construction of the Southcourt Avenue road objects. This phase of scene generation was relatively linear and deterministic compared with other rulesets. This was probably due to the structural role of the road in the scene, necessitating its creation early in the interpretation process and independent of many other objects that were created later. However, the linear sequencing of rules

was not an enforced procedure in the interpreter, which may potentially select and fire any matching rule from a given ruleset. In this case, the sequencing of rules was achieved by the modularization of rules into named sets, the isolation of rules into their own nested interpretation loops, and the use of labels – in which a rule will not fire on an object until it has received a particular label from another rule. Importantly, linearity in the interpretation process was therefore due to the restrictive conditions in the antecedents of the rules rather than specific firing instructions in the interpreter mechanism. Other methods for controlling the sequence of rule firing are presented with later rulesets. The next stage of the scene generation determined the configuration of land parcels around the Southcourt Avenue road.

7.4.2 Land Parcels

The lots of land along the straight portions of Southcourt Avenue are perpendicular to the road and occupy fifty-foot lot frontages. The lots around the road curves are more awkward in shape but influenced somewhat by the direction of the steeper slopes there or the central axis/orientation of the neighbourhood. Waterfront lots are oriented towards the shoreline rather than the road presumably in order to maximize their number. Approximately twenty percent of the lots have been reconstituted to form larger parcels of land, most often as two lots amalgamated into one, three into two, or, more rarely, three into one (Figure 7.6 (iv, v)). The already small lots are rarely subdivided, but there is one such case at the northern end of the neighbourhood.

From these observations, the goal of the parcels rulesets was to generate the locations of Lot-Lines in order then to construct Land-Parcel polygons. The Lot-Lines perpendicular to the straight road segments were iteratively constructed with rotational modifications at the shoreward end. Where the road curves in the steeper northern end of Southcourt Avenue, Lot-Lines adjoining the main road were created perpendicular to it, the land on the interior of the concave curves was isolated to form triangular lots, and the rest of the Lot-Lines were oriented around the steep slopes there. The line-polygon topology for the Land-Parcels was easily constructed once the Lot-Lines were finalized. The polygons were then subdivided or amalgamated as appropriate, and then labelled for their use in future rulesets.

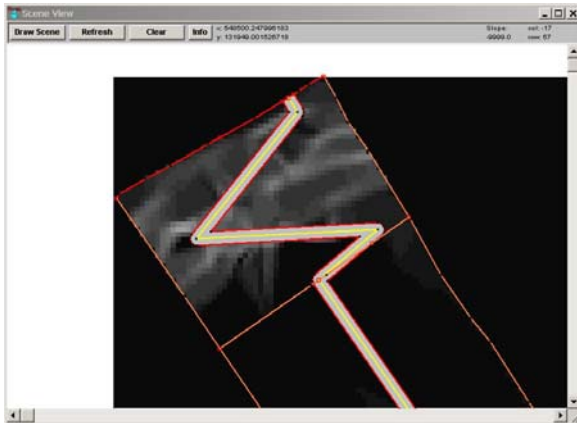
The grammar interpreter used one parameter (**lot-frontage**) and twenty-one rules (in four rulesets) to construct Lot-Lines and thirteen rules (in seven rulesets) to assemble, modify and label the Land-Parcels. The *start-lotline-ruleset* rule was fired from the World ruleset initiating a nested interpretation loop using the rules of the *construct-lot-lines* ruleset and the current working-scene of objects following the completion of the roads ruleset. The rule-based insertion of Lot-Lines was performed with reference to the Road-Centreline, but also intersecting any existing Road-Edges, Shorelines and Lot-Lines along their path.

In order to begin inserting the regular Lot-Lines on the straight road, a starting insertion point was calculated by the `add-initial-lot-line` rule (Figure 7.11 (i)). Once an initial line was established, a starting rule initiated an interpretation of the `adding-perp-lot-lines` ruleset comprised of a single rule, namely `add-perpendicular-lot-line`. This rule was fired continuously, each time inserting a Lot-Line object with the predefined `*lot-frontage*`, extending the new Lot-Line through the nearest Road-Edge, Shoreline or Lot-Line object, and then splitting the polylines that it touched (Figure 7.11 (ii), (iii), (iv)). The rule could no longer fire when an unprocessed Road-Centreline was shorter than the `*lot-frontage*` value. Returning to the main interpretation loop of `adding-lot-lines`, the cleaning rules `remove-lot-lines-from-road` and `consolidate-centreline-segments`, respectively, deleted any Lot-Lines inside a Road-Area object and reconsolidated the Road-Centreline objects that were split during the Lot-Line insertions.

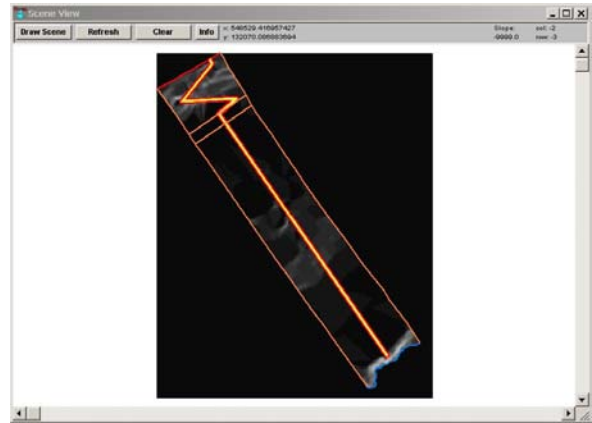
The `half-the-waterfront-area` rule inserted a Lot-Line where a Road-Edge was near to the Shoreline. The `rotate-lot-lines-near-shoreline` rule identified Lot-Lines that were perpendicular to the road but too close to the Shoreline, rotated them ninety degrees, and extended them to the Shoreline objects (Figure 7.11 (v)). The `label-remnant-centreline` rule merely labelled the last Road-Centreline object so that it was no longer considered for inserting Lot-Lines.

At the other end of Southcourt Avenue, the curving Road-Edges were labelled as such by the `label-curved-road-edges` rule. For each labelled curved Road-Edge, a nested interpretation loop was started using the rules of the `adding-lot-lines-to-road-curves` ruleset. Rules in this set add Lot-Lines to concave or convex Road-Edge curves. The `add-lot-lines-on-concave-road-curves` rule created a preliminary Lot-Line (a ‘sketch’) that is perpendicular to the angular bisector of the road curve. The `ignore-tiny-concave-corners` rule ensured that corners that are geometrically, but not effectively, concave (e.g. almost 180 degrees) were ignored by the other rules by the use of a label. Further rules, `remove-sketch-for-short-lots` and `align-sketch-with-neighbourhood-axis` (both within a `cleaning-concave-corner-lots` ruleset) respectively deleted short sketches that would have created unrealistic Land-Parcels, and aligned valid sketch lines with the central axis of the neighbourhood (stored in the parameter `*the-area-axis*`). Finally, the finished Lot-Line was extended and joined to the nearest Lot-Lines or Road-Edges by the `create-lot-line-from-sketch` rule (Figure 7.11 (vi)). When a curved Road-Edge was convex, the `add-lot-lines-on-convex-road-curves` rule added a new Lot-Line calculated as the angular bisector of the curve (Figure 7.11 (vi)). Some necessary cleaning operations, launched as rules in a `cleaning-convex-corner-lots` ruleset, removed the inserted Lot-Lines that were too close to other Lot-Lines or Road-Edges creating thin, nonsensical lot shapes.

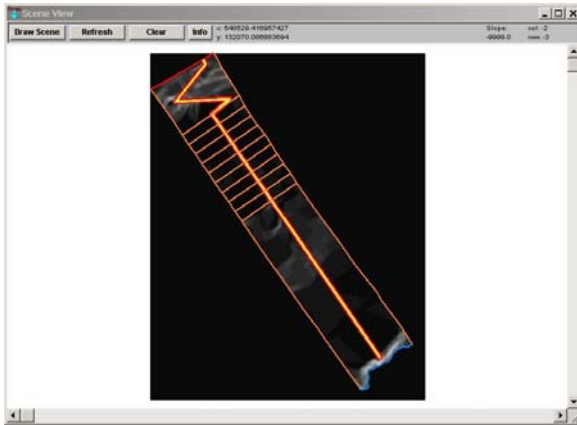
At this stage, the Lot-Line objects were laid out for the site, and the processing returned to the



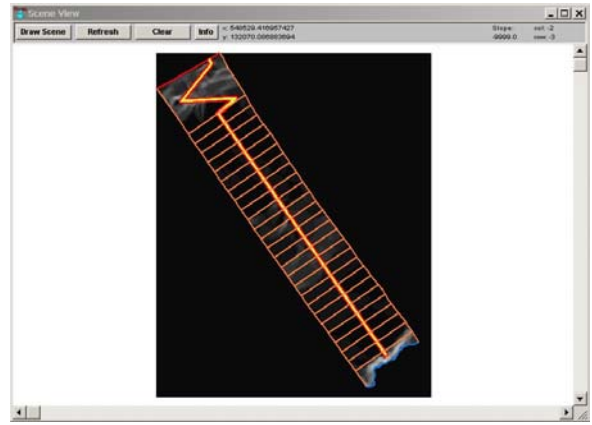
(i) add-initial-lot-line



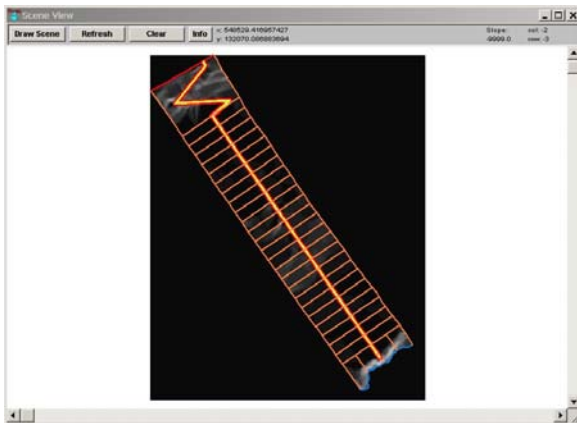
(ii) add-perpendicular-lot-line



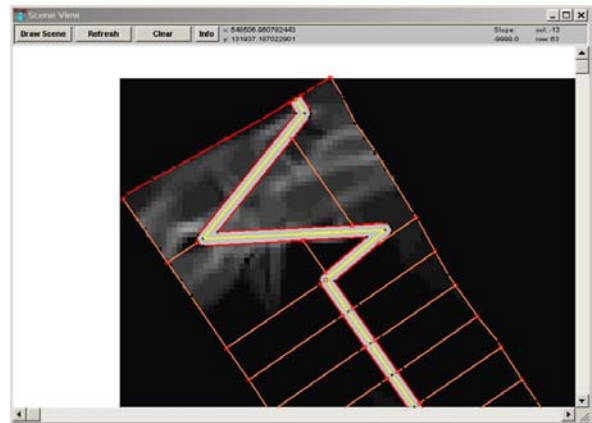
(iii) add-perpendicular-lot-line



(iv) add-perpendicular-lot-line

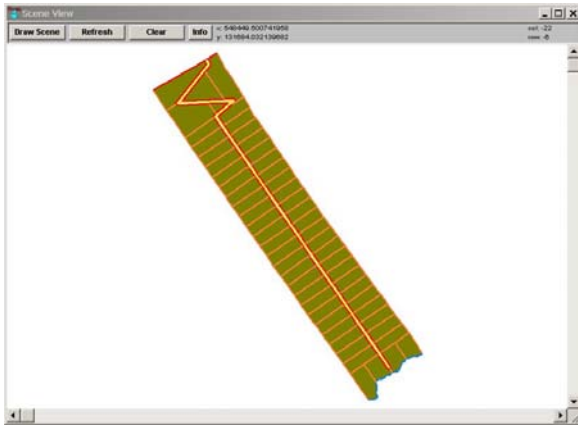


(v) rotate-lot-lines-near-shore

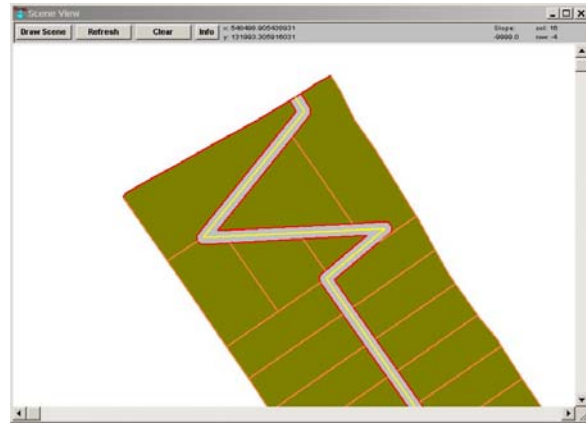


(vi) add-lot-lines-on-concave-road-curves,
add-lot-lines-on-convex-road-curves

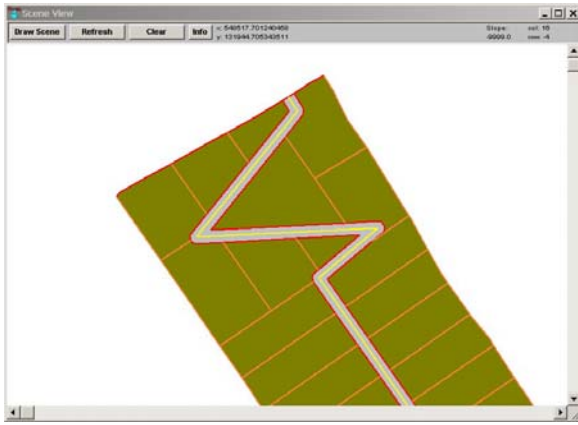
FIGURE 7.11 GENERATION OF PARCEL FEATURES FOR SOUTHCOURT AVENUE



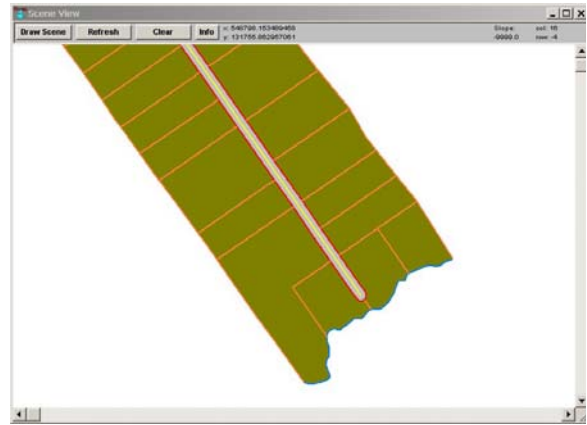
(vii) build-parcel-polygons



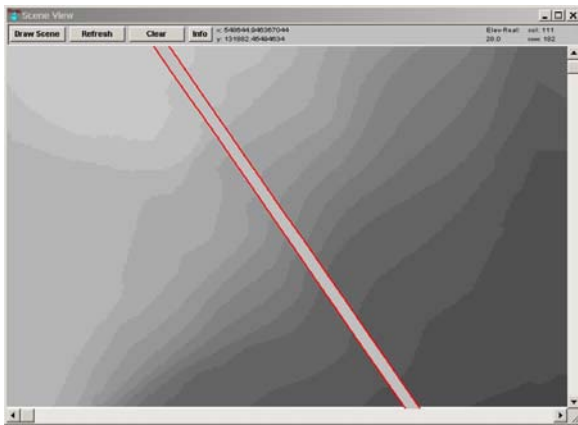
(viii) subdivide-large-parcels



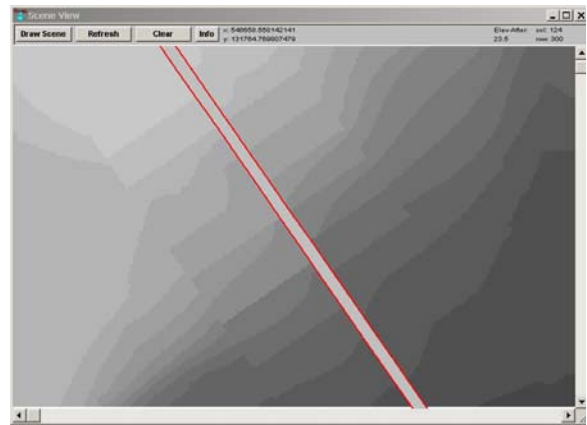
(ix) subdivide-long-large-parcels



(x) amalgamate-parcels



(xi) *before* regrade-parcels



(xii) regrade-parcels

FIGURE 7.11 GENERATION OF PARCEL FEATURES FOR SOUTHCOURT AVENUE (CONTINUED)

outermost interpretation loop using the World ruleset. Another starting rule initiated the processing of rules of the `build-parcels` ruleset. The `build-parcel-polygons` rule found any Lot-Lines, Road-Edges or Shoreline objects that were not topologically part of a Land-Parcel object, and used network traversal methods to determine from them a closed loop of joined polylines, which were then used to form the edges of a new Land-Parcel polygon object. This rule fired continuously until all such polylines were part of a Land-Parcel (Figure 7.11 (vii)).

All Land-Parcels had been generated at this stage, but were still subject to subdivision and amalgamation. The `select-parcels-to-subdivide` rule determined that parcels that met a minimum area requirement were eligible to be subdivided and were passed to a nested interpretation loop with the `subdivide-parcels` ruleset which contained two rules. The syntax of this rule is included below:

```
(define-rule
  :NAME select-parcels-to-subdivide
  :RULESETS build-parcels
  :IF ((is-a ?a 'Land-Parcel)
       (has-notany-labels ?a '(subdivided subdivision-failed
                               amalgamated amalgamation-failed))
       (> (area ?a) 900)
       (rule-has-not-fired-with-objects 'select-parcels-to-subdivide ?a))
  :THEN ((set-label ?a 'to-be-subdivided)
         (interpret (rules-of 'subdivide-parcels) (list* ?a (lines ?a)))
         (when (exists ?a) ; if ?a still exists, it was not subdivided.
              (remove-label ?a 'to-be-subdivided)
              (set-label ?a 'subdivision-failed)))
  )
```

The `subdivide-large-parcels` rule applied to those parcels that were roughly rectangular in shape and divided them into Land-Parcels of approximately 600 square metres (an average for the Southcourt Avenue neighbourhood; Figure 7.11 (viii)). A second rule, `subdivide-long-large-parcels`, applied to those parcels that were elongated and allowed them the flexibility to be subdivided into Land-Parcels of a slightly smaller area (Figure 7.11 (ix)).

Similarly, the `select-parcels-to-amalgamate` rule passed Land-Parcels to the `amalgamate-parcels` ruleset for interpretation. All Land-Parcels were eligible for amalgamation but this interpretation loop was passed a simplistic goal to continue rule processing until twenty percent of the Land-Parcels had been amalgamated. While more sophisticated criteria could be established for lot amalgamation, this stage illustrates the use of a goal to achieve a basic frequency of objects that reflects the situation of the actual study area. In the `amalgamate-parcels` ruleset, one of the rules amalgamated two adjacent Land-Parcels into one, while another amalgamated three Land-Parcels into two (Figure 7.11 (x)). A third rule handled the special case of waterfront Land-Parcels. It ensured that waterfront lots were not amalgamated with each other and that any amalgamation with a waterfront lot adjacent to the road did not result in the restriction of access to another waterfront lot further from the road.

Finally, with processing returning to the World ruleset again, the Land-Parcel objects were prepared for the placement of houses by the `prepare-parcels-for-development` ruleset. The rules of this set mostly labelled the Land-Parcel objects for the later rulesets. Each Land-Parcel was labelled as developable if it met a minimum area criteria, or undevelopable if it did not. Additional labels were used to identify those Land-Parcels on the shoreline or on the curved part of the road. These criteria were then available to be re-used in the syntax of any future rule, but the processing of such rules was made easier by referencing these labels instead of performing calculations.

While the `plot-roads` ruleset illustrates the influence of raster slope data on the geometry of vector Road-Centreline objects, a final rule in this ruleset, `regrade-parcels`, illustrates the influence of vector objects on the underlying raster data. This rule applied to any developable Land-Parcel for which the mean slope value was greater than three percent (excluding the waterfront and road-curve parcels which had exceptionally high slope values). The consequent of this rule adjusted the elevation values of each grid cell within the Land-Parcel to ensure that each elevation was no more than 0.5 metres from the mean elevation of the parcel, thus levelling its topography although not completely flattening it (Figure 7.11 (xi), (xii)). With the Lot-Lines delineated and Land-Parcels assembled and ready for development, the interpreter returned to the World ruleset to proceed with the placement of houses on the parcels. The Land-Parcels presented herein were manually chosen as the most similar lot configuration to that of the existing Southcourt Avenue neighbourhood. The purpose of this interjection was to facilitate a reasonable visual comparison between the existing Southcourt Avenue landscape and the end result of this grammar interpretation example. Other possible lot configurations are shown in alternative landscape scenes following this example of scene generation.

7.4.3 Houses

The general character of the buildings on Southcourt Avenue can be appreciated from the photographic survey in Figure 7.5. For the most part, the houses are comprised of orthogonal rooms and are aligned with the edges of the lots. Due to the small size of many of the lots, the houses are often located at the geometric centre of the property, thus maximizing the use of available space. Where there is more space available on the larger lots, the houses tend to be located based on the highest elevations of the parcel. Despite their ocean views, the houses are often oriented towards the road. Exceptions include waterfront homes which face the shoreline and houses at the northern end of the neighbourhood which face the main road or align with the steeper slopes there. The walls of the houses are painted using various pastel colours, while the roofs are invariably painted white.

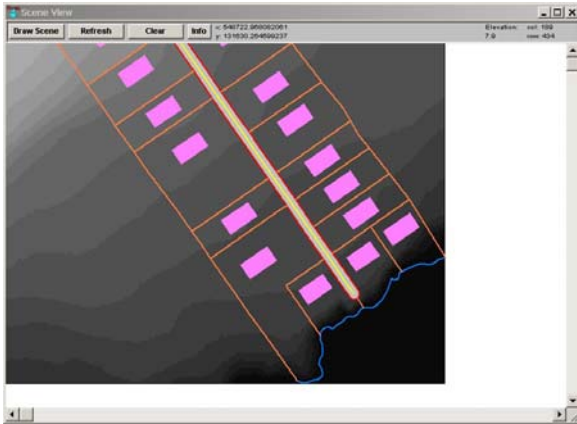
Without any data available on the interior of the Southcourt Avenue houses, it was difficult to derive a grammatical model for House shapes (c.f. Palladian floor plans in Stiny & Gips, 1978; Stiny &

Mitchell, 1978a). A surrogate technique was developed that started with a rectangular House object in the centre of a Land-Parcel. The simple House was then aligned, oriented with a ‘front’ direction, and relocated if appropriate. A more convincing geometric shape was developed by creating Building-Addition objects that overlapped the House. Building-Additions were added using randomized parametric dimensions and locations and sometimes rejected due to proximity to Lot-Lines or other Building-Additions. The House objects were augmented with Chimney objects, assigned a material colour, and formally associated with the Land-Parcel within which they were situated.

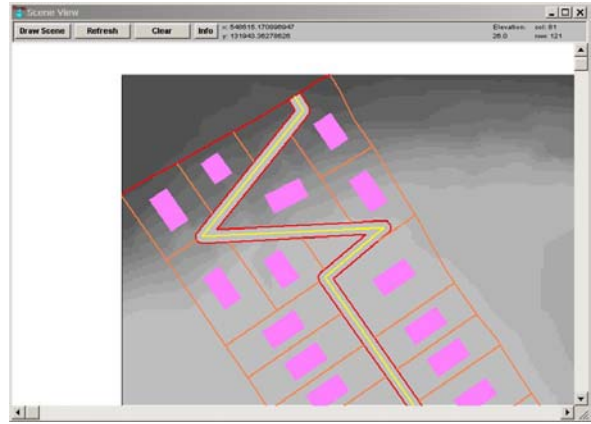
Twenty rules organized into three rulesets implemented the above strategy of generating and locating House objects. The `construct-house` ruleset was initiated from a starting rule in the World ruleset. For each Land-Parcel, the `insert-house` rule created a standard rectangular House object at the centroid of the parcel. With the creation of a House object, the alignment rules then became applicable to the working-scene. The `align-house-with-lot-line` rule rotated the House until its long axis was aligned with a Lot-Line that is joined to the Road-Edge (Figure 7.12 (i)). For waterfront Land-Parcels, the `align-house-with-shoreline` rule rotated the House to align the axis of the shoreline as stored in `*the-shoreline-axis*` parameter (Figure 7.12 (i)). For Land-Parcels on the curving road that were not rectangular, the `align-house-with-corner-axis` rule aligned the House with the bisector of the road curve (Figure 7.12 (ii)). Once the simple House object was aligned, an orientation vector was calculated to identify the front of the House. The `associate-front-of-house-to-road` and `associate-front-of-house-to-shore` rules calculated and stored this vector for each House in the working-scene.

Four rules were designed for relocating a House object on a Land-Parcel. If a Land-Parcel met a minimum area criterion then the `relocate-house-on-large-lots` rule translated its House object in the direction of the highest elevation on the Land-Parcel, but not so far as to overlap the edges of the Land-Parcel (Figure 7.12 (i)). Similarly, near the shoreline, the `relocate-house-on-waterfront-lots` rule ensured that a House was located at least a minimum distance from the Shoreline (Figure 7.12 (i)). The `move-house-from-main-road` and `move-house-from-avenue` rules altered the shape of the House object if it was situated too close to the main road or other Road-Edges (Figure 7.12 (ii)).

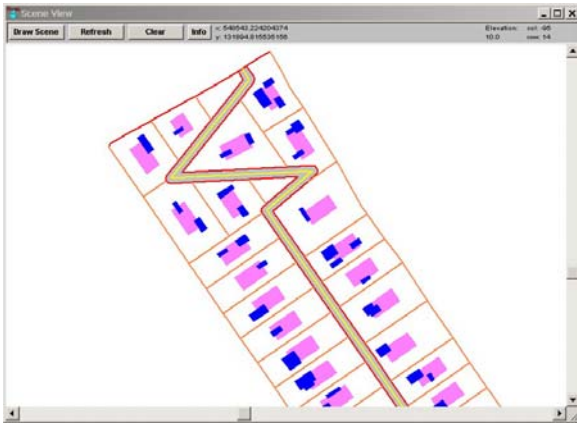
At this stage of the scene generation, the position and orientation of the simple House shapes were established. A new ruleset, `extend-houses`, was started from the World ruleset to begin the process of generating additional rooms on each House. The `house-additions` ruleset was then processed until a random number of additions (between 0 and 4) had been generated. Within this ruleset, the `create-addition` rule determined a random point on the perimeter of the House and created a Building-Addition object centred on that point and aligned with the edges of the House (Figure 7.12 (iii), (iv)). The selection of the insertion point was stochastically biased towards points on the shorter sides of the rectangular House, and the dimensions of the Building-Addition were determined randomly within



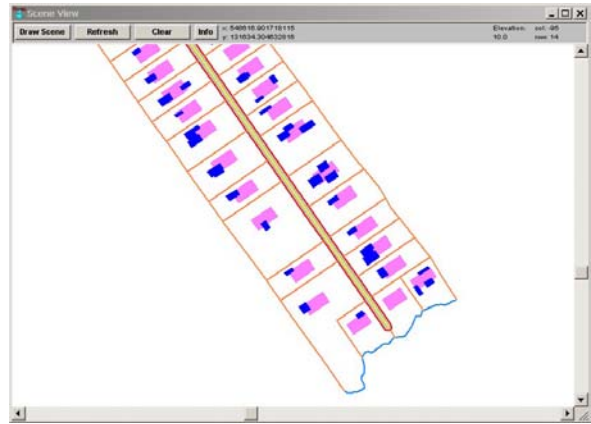
(i) insert-house, align-house-with-lot-line, relocate-house-on-large-lots



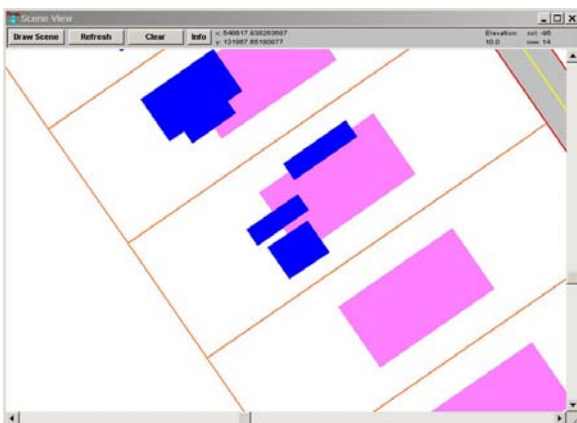
(ii) insert-house, align-house-with-corner-axis, move-house-from-main-road



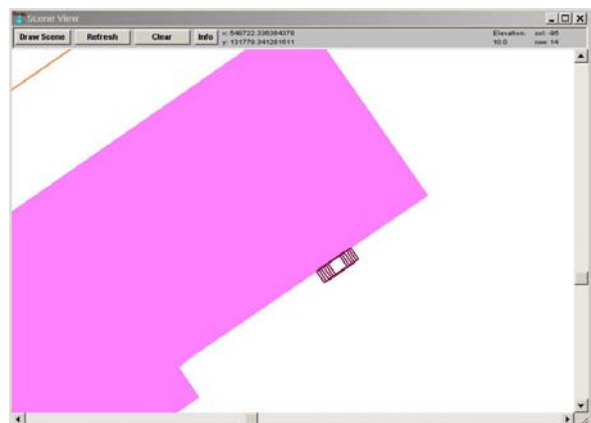
(iii) create-addition



(iv) create-addition



(v) *before* remove-addition-with-narrow-gap



(vi) build-chimney

FIGURE 7.12 GENERATION OF HOUSE FEATURES FOR SOUTHCOURT AVENUE

numeric limits.

Five further rules in the `house-additions` ruleset were designed to modify or remove Building-Additions that were not suitable. The `reduce-narrow-addition-on-short-segment` and `reduce-narrow-addition-on-long-segment` rules used different numeric tolerances but both resized long, narrow Building-Additions that protruded from the side of a House. The `remove-addition-near-lot-line` rule removed any Building-Addition that fell too close to the edges of a Land-Parcel. The `remove-addition-with-narrow-gap` rule removed a Building-Addition if it resulted in a narrow gap between it and another Building-Addition (Figure 7.12 (v)). Finally, the `remove-proximal-additions` rule ensured that additions were not created on top of each other by removing any Building-Addition whose centre was located too close to that of another Building-Addition. The earlier rules that created the Building-Additions introduced a certain arbitrariness to the Southcourt Avenue scene, but the subsequent rules regrounded the scene in the reality of the houses in that specific location. Although the `house-additions` ruleset employed a gross simplification of room configuration, it generated polygons that were credible as building outlines. After the `house-additions` ruleset completed its interpretation loop, the generated Building-Additions were combined with the original rectangular House (a union operation) to form a new more complex House object. Also at this stage, the elevations of the House were calculated from the grid data for each vertex in the perimeter.

The rule interpreter then returned processing to the World ruleset and three further rules were applied to the working-scene. The `build-chimney` rule randomly inserted a Chimney object at a point on the House perimeter, provided it was a minimum distance from a corner as fireplaces are not usually located in the corner of a room (Figure 7.12 (vi)). The `associate-houses-with-parcels` rule simply associated the name of a House object with the name of a Land-Parcel object so that the spatial calculation did not have to be performed for further rules.

Finally, the `paint-house` rule created and assigned a material, representing a pastel colour, to each House object in the scene. Section 7.1 described how colour data were extracted from digital photographs, while Figure 7.6(viii) showed the distribution of house colours along Southcourt Avenue. Figure 7.13 shows the frequency distributions of hue, saturation, and lightness values for the houses on Southcourt Avenue. Generally, the hue values vary across the range of 0-255, but there is some concentration in the lower values (red, orange, yellow). The saturation values are typically low and the lightness values typically high, producing the light pastel colours displayed on the map in Figure 7.6(viii). From these data, minimum and maximum bounds were selected for each of hue, saturation and lightness and a rule was developed to randomly generate a colour from within these bounds. This rule was used to assign a colour (material) to each House object in the scene. The rule syntax is as follows:

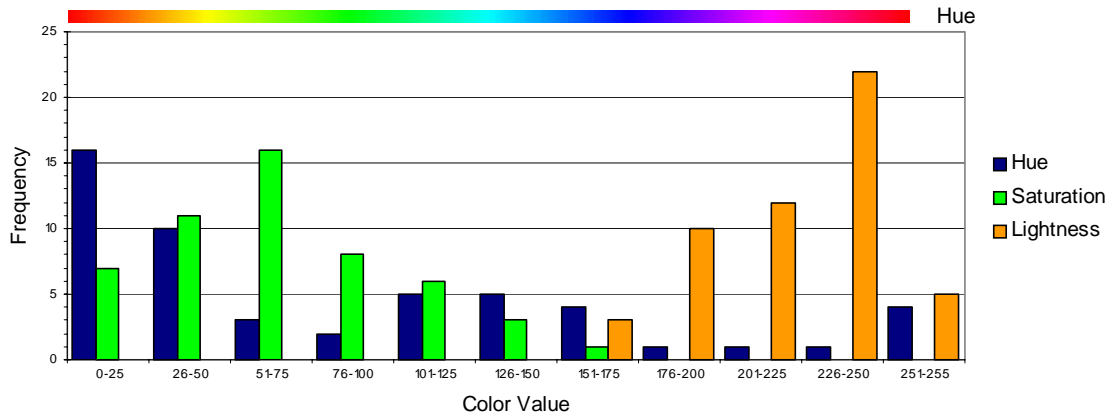


FIGURE 7.13 DISTRIBUTION OF HUE, SATURATION AND LIGHTNESS IN HOUSE COLOURS

```
(define-rule
 :name set-house-paint-color
 :rulesets construct-house
 :if ((is-a ?a 'House)
      (eql (material ?a) 'DEFAULT))
 :then ((set-material ?a (define-random-material :hue-min 0 :hue-max 360
 :lightness-min 78 :lightness-max 100
 :saturation-min 20 :saturation-max 40))))
```

The presence of Houses on developable lots set the stage for further rules to add secondary features to each Land-Parcel. Driveways, walls, hedges and trees were created in the remainder of the rulesets for the Southcourt Avenue grammar.

7.4.4 Driveways

Driveways are found on virtually every property facing Southcourt Avenue. They obviously occur at the road's edge for a parcel but, more specifically, they are most often located at the corners of a parcel rather than entering the property at the middle. While driveways are generally located at positions that are level with the road and/or near to a house, some driveways have required regrading of the land they occupy. Where the slope and elevation of the lot and the location of the house permit, driveways may extend to the house, beside the house, or past the house into the back-yard. Driveways that extend past the house sometimes have a separate garage that is painted the same colour as the house.

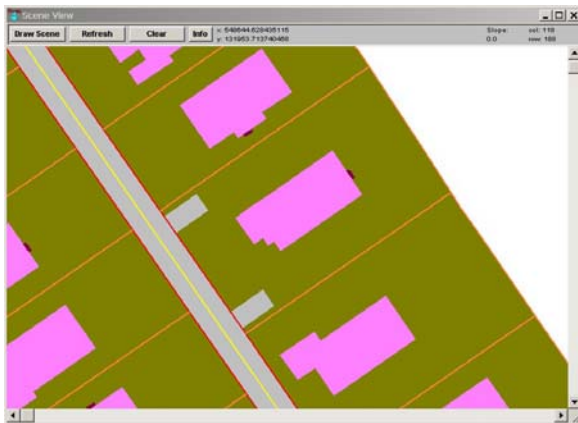
The strategy employed by the Driveways rules to generate this character was to find all potential Driveway locations for a Land-Parcel and then eliminate individual Driveways until only one potential location remained. Driveways on waterfront lots required special operations as there is only one potential location for a Driveway (away from the Shoreline) and a single Driveway may be shared between two Land-Parcels. Once a single Driveway object location was determined, further rules

extended it, levelled it, and created and painted a Garage object.

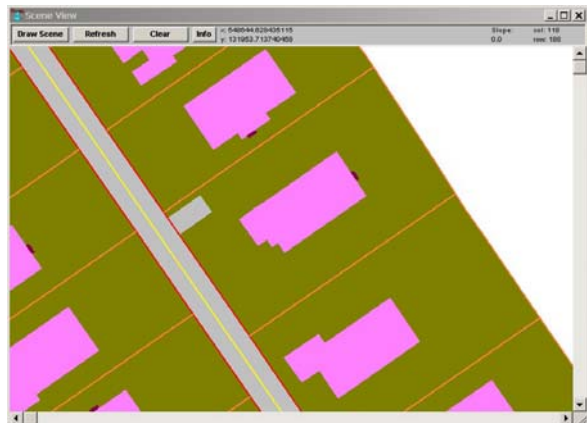
The landscape grammar used one parameter (**driveway-width**) and fourteen rules to generate Driveway objects. Unlike many of the previous rulesets, all fourteen rules were contained within the *add-driveways* ruleset (there were no subordinate modular rulesets within it). Despite this, rule sequencing was still partially controlled by a rule selection strategy stated in the starting rule that initiated the interpretation of the driveway rules from the World ruleset. This particular selection strategy was a function that accepts a master list of rules and a second list of rules defining an order of priority. The function preserves the highest order rule occurring in the first list and removes all lower order rules. Any rules present in the first list but not in the ordered list are preserved. In this case, the rule sequencing was *partially* controlled because the priority order of the four rules to remove an invalid driveway was specified, but the other ten driveway rules could be fired at any time that they were found applicable.

The *place-driveways* rule inserted two potential Driveway objects, each centred on an end of a Road-Edge in the boundary of a Land-Parcel. The *align-and-move-driveways* rule rotated the initial Driveway object to align with the Land-Parcel edges and then moved its centre from the Road-Edge slightly further into the Land-Parcel (Figure 7.14 (i)). Once this occurred, the rules for removing invalid driveways became applicable to the scene. One or more of these rules were fired according to the priorities set in the rule selection strategy. The end result was a single Driveway object that was judged a better choice than the others. The rules are presented here in order of decreasing priority. The *remove-dway-over-lot-line* rule removed Driveway objects that extend over an adjacent edge of the Land-Parcel. This mainly occurred in awkwardly shaped lots where the particular driveway location was infeasible. The *remove-steeply-sloping-dway* rule removed the Driveway with the largest mean slope value (which would be the more difficult to traverse). The *remove-furthest-dway-from-house* rule assumed that it is desirable to locate a Driveway close to the House for ease of use Figure 7.14 (ii). Finally, in the unlikely event that two Driveway objects were identical in their slope and proximity to a House, the *remove-random-driveway* rule ensured that one Driveway was selected randomly and removed.

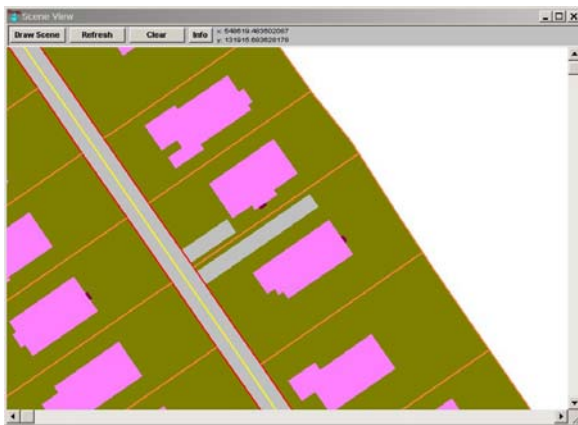
The selected Driveway was then able to be modified by further rules in the ruleset. The *ensure-driveway-joins-avenue* rule extended the Driveway into the Road-Area in case there was no longer a complete overlap as a result of the earlier modifications. Two rules were potentially applicable to extend a Driveway towards a House. The *extend-driveway-past-house* rule applied if a Driveway object could be extended past a House object. A Driveway was extended either to a point beside the House or directly past the House and into the back yard (Figure 7.14 (iii)). If the location of a House prevented this, the *extend-driveway-to-house* rule extended the Driveway only to the nearest edge of the House (Figure 7.14 (iii)).



(i) place-driveways, align-and-move-driveways



(ii) remove-furthest-dway-from-house



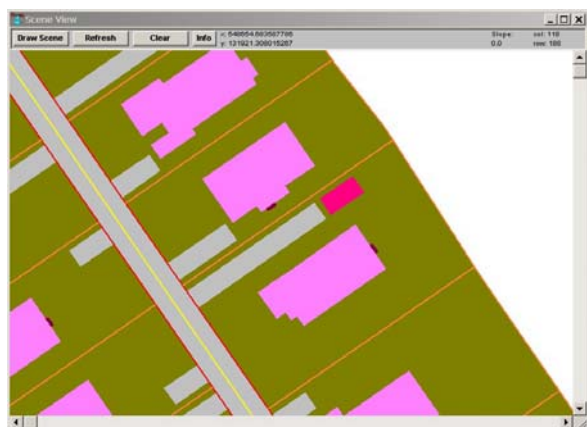
(iii) extend-driveway-past-house,
extend-driveway-to-house



(iv) share-driveway-for-waterfront-lots



(v) level-driveway



(vi) create-garage

FIGURE 7.14 GENERATION OF DRIVEWAY FEATURES FOR SOUTHCOURT AVENUE

Given the orientation of Land-Parcels on the shore, there were fewer options for potential Driveways located there. The `place-driveway-for-waterfront-lots` rule inserted a single Driveway object at a location on the Road-Edge furthest from the Shoreline. The `share-driveway-for-waterfront-lots` rule applied to a Land-Parcel that was not directly connected to a Road-Edge. The consequent of the rule extended the Driveway of the neighbouring Land-Parcel to the House of the current Land-Parcel (Figure 7.14 (v)).

When a Driveway object was finalized, the `level-driveway` rule ensured that the elevation values of the grid cells within the Driveway did not vary by more than a small amount, thus creating a reasonably level surface (Figure 7.14 (vi)). If a Driveway extended past a House into the back yard, then it potentially had a Garage located at its terminus. The `create-garage` rule inserted a Garage object (sixty percent of the time as reflected in the observed frequency from the site) and the `paint-garage` rule ensured that the material of any Garage was the same as that of the House which it serves (Figure 7.14 (viii)). With the Building objects constructed, rules for more superficial landscape objects could then be applied. Application of the rules for walls and hedges on the Land-Parcel boundaries were applied next.

7.4.5 Walls and Hedges

The properties on Southcourt Avenue often have walls or hedges located around their periphery. The walls, constructed of concrete-block or stone, are most often painted the same colour as the house and are usually between 3 and 6 feet in height. Some are retaining walls on a regraded lot. The tops of some walls, for instance retaining walls, have a level elevation along their length regardless of the elevation of the ground beneath them. Others slope gradually, reflecting the terrain. The waterfront lots have short stone walls on their shoreward side. The stone is more visually appealing when viewed from the water and the low height does not impede the ocean views of the houses. The hedges are of a variety of species and also between 3 and 6 feet in height. Both walls and hedges were inferred as serving the purpose of either a simple property boundary marker or a visual barrier from people on the road or neighbouring properties. Visual barrier functions are served by higher walls and hedges.

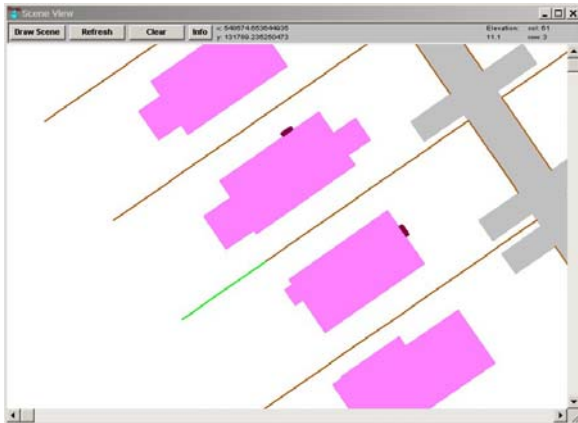
Given the common functional roles identified above, it was decided to model walls and hedges in the same ruleset. Because objects serving as visual barriers also serve a dual function as boundary markers, linear, but non-physical, Boundary-Marker objects were constructed on all Lot-Lines and Road-Edges. Where appropriate, some Boundary-Marker objects were converted into Visual-Barrier objects. Both Boundary-Markers and Visual-Barriers were then actualized as Wall-Line or Hedge objects. Wall-Lines on the shore were constructed directly as they served the function of neither a boundary marker nor a visual barrier. Once the linear objects were generated, polygonal Garden-Wall objects were created

from the Wall-Lines and then coloured to match the nearest House.

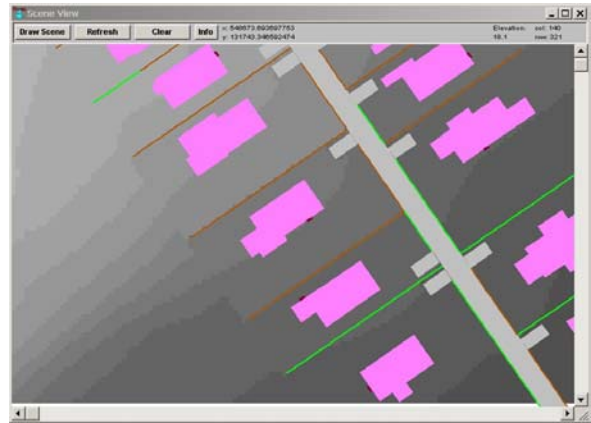
Twenty-seven rules, organized into four rulesets, were used to implement walls and hedges in the working-scene. Interpretation of the `plot-walls-hedges` ruleset was initiated from a starting rule in the World ruleset. A further starting rule (`start-creating-boundary-structures`) inside the `plot-walls-hedges` ruleset initiated interpretation of the `boundary-structures` ruleset, which creates Boundary-Marker and Visual-Barrier objects. A rule selection strategy was employed to ensure that this starting rule was fired first in the `plot-walls-hedges` ruleset.

The `boundary-structures` ruleset contained six rules. The `boundary-marker-between-parcels` rule created a Boundary-Marker object on any Lot-Line that separated two Land-Parcels. Likewise, the `boundary-marker-at-roadside` rule created a Boundary-Marker object on any Road-Edge. Three rules accommodated the conversion of Boundary-Markers into Visual-Barriers. The `visual-barrier-from-neighbours` rule applied to Boundary-Markers on a Lot-Line between Land-Parcels. The `visual-barrier-in-backyard` rule was a variant of this rule that split a Boundary-Marker in two at approximately the location of the rear of an adjacent House. The portion of the Boundary-Marker in the back of the House was converted to a Visual-Barrier object, in order to provide privacy in the back yards of adjacent houses (Figure 7.15 (i)). The `visual-barrier-from-roadside` rule converted a Boundary-Marker located on a Road-Edge into a Visual-Barrier. In each of these three rules, the interpreter may or may not have fired on the applicable Boundary-Marker objects. Some Boundary-Markers may have been left as they were. The `retaining-structures` rule identified those Boundary-Markers where there was such a difference in elevation on either side of it that a retaining wall was warranted (Figure 7.15 (ii)). In this case, the role as a retaining structure prevailed over boundary markers and visual barriers, and a Wall-Line was constructed immediately with the appropriate elevation and height data.

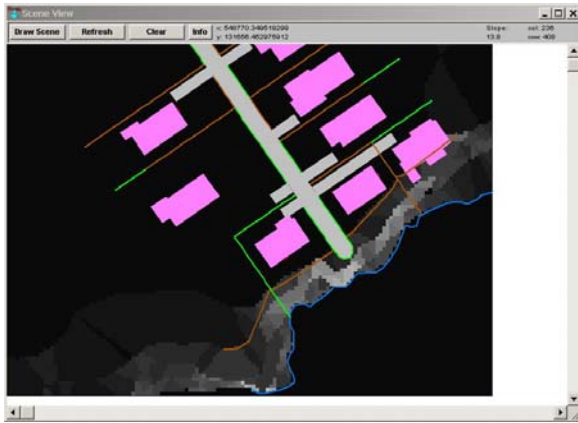
With the Boundary-Marker and Visual-Barrier objects generated, the rule interpreter returned processing to the `plot-walls-hedges` ruleset. Five rules were employed to convert the Boundary-Markers and Visual-Barriers into Wall-Lines and Hedges. The decision between the creation of a Wall-Line or a Hedge was mostly arbitrary, except for at the Road-Edges labelled `the-main-road` where only walls were used (presumably related to the traffic density there). The distinction between the five rules was due to the method used to calculate the polyline's elevation and height values that were used later to create 3D objects. While a 3D wall was modelled in the GDS software as a vertically extruded polygon object, applying this technique to a hedge produced a solid that appeared identical to green walls. Therefore, a hedge was modelled differently by sweeping a standard vertical shape along a 3D line to form a solid. The differences in these two 3D modelling techniques had implications for the 2D landscape grammar in that the top of a wall could be modelled with a flat elevation (extruded downwards



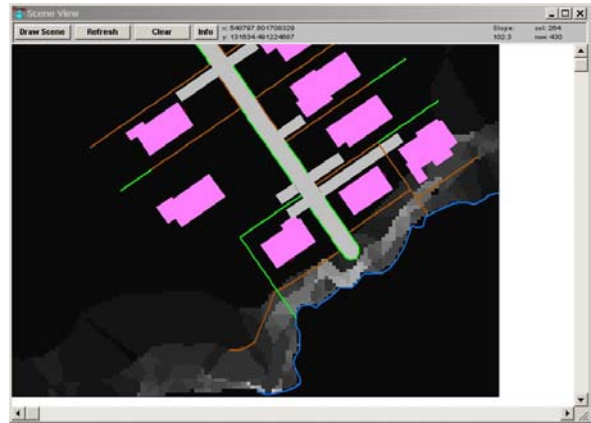
(i) visual-barrier-in-backyard



(ii) retaining-structures



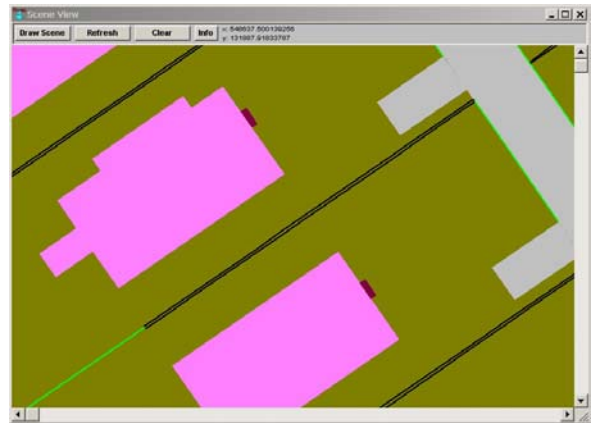
(iii) create-shoreline-wall



(iv) move-shoreline-walls-near-house



(v) remove-wall-hedge-on-shore



(vi) create-wall-from-wall-line

FIGURE 7.15 GENERATION OF WALL AND HEDGE FEATURES FOR SOUTHCOURT AVENUE

into the ground) or as a sloping object that followed the terrain with varying elevations along its length. A hedge, however, had to be modelled using the latter technique in order to later create a 3D line that followed the terrain surface. For walls, the decision to create a flat-topped or sloping wall was based on the range of elevation values along the polyline or on the difference in elevation values on either side of the polyline. A smoothing function also had to be employed to regularize sudden rises and falls in elevation due to the high granularity of the underlying grid data.

The `lotline-wall-from-visual-barrier` rule created a Wall-Line object from a Visual-Barrier located on a Lot-Line. The `roadside-wall-from-visual-barrier` rule created a Wall-Line object from a Visual-Barrier located on a Road-Edge. The `main-road-wall-from-visual-barrier` rule performed the same activity when the Road-Edge was labelled `the-main-road`, but assigning a larger height value to provide better separation from fast-moving traffic on that thoroughfare. The `hedge-from-visual-barrier` rule created a Hedge object from a Visual-Barrier, and the `wall-or-hedge-from-boundary-marker` rule randomly created a Wall-Line or Hedge object from a Boundary-Marker object that had not been previously converted to a Visual-Barrier.

Some cleaning operations were necessary following the processing of the above rules. Where a Hedge and a Wall-Line topologically shared a node, they also shared the elevation and height data of that node. This was not desirable for the 3D component of the landscape modelling procedure because the base elevation of a flat-topped Wall-Line is zero (it is extruded from underground to the terrain elevation plus its height) while that of a Hedge is the surface elevation at that node. The `detach-hedges-from-walls` rule was used therefore to ensure that walls and hedges were not joined by truncating one of the lines by a very small distance. The `tie-ret-road-wall-heights` and `tie-nonret-road-wall-heights` rules both adjusted the elevations and heights of two Wall-Lines. The former applied to the joining of a retaining Wall-Line to another Wall-Line (in which case the height of the retaining wall prevailed), while the latter applied to the joining of any other two Wall-Lines (in which case a smoothing adjustment was made).

A separate ruleset was developed for generating walls at the shoreline. Interpretation of the `shoreline-walls` ruleset was initiated whenever a Land-Parcel labelled `waterfront-lot` was under review. The `create-shoreline-wall` rule analyzed the topographic grid data for the Land-Parcel and identified the boundary between the flat usable land and the steep rocky shore. This boundary was used as the geometry of a new sloping Wall-Line object with a low height of three feet (Figure 7.15 (iii)). Sometimes this raw calculation created a Wall-Line that either intersected the House on that parcel or was so close to the House that little space was left in front of it. The `move-shoreline-walls-near-house` rule rectified this situation, when it occurred, by recreating the Wall-Line as a polyline parallel to the shoreward edge of the House and offset to allow a small front yard area (Figure 7.15 (iv)).

The waterfront Land-Parcels were also subject to the previous rules that created Wall-Lines and Hedges on the Lot-Lines and Road-Edges of a Land-Parcel. Where these objects were generated, the `join-shoreline-wall-to-boundary`, `split-boundary-at-shoreline-wall`, and `remove-shoreline-wall-dangles` ensured that the shoreline wall was cleanly connected to them. Any parts of Wall-Lines or Hedges that extended past a shoreline wall towards a Shoreline object were removed by the `remove-wall-hedge-on-shore` rule (Figure 7.15 (v)). Likewise, any Wall-Lines or Hedges in the working-scene that intersected a Driveway (e.g. where a shared Driveway crossed a Lot-Line to another Land-Parcel) were truncated by the `remove-wall-hedge-in-driveway` rule.

Upon thorough application of the above rules to the working-scene, the Wall-Lines and Hedges were finalized. At this point, the `finish-walls` ruleset was applicable to the scene. The Wall-Lines were converted to polygonal Garden-Wall objects and then assigned a material. The `create-wall-from-wall-line` and `create-shoreline-wall-from-wall-line` rules buffered a Wall-Line to create a Garden-Wall and then transferred its elevation and height data to the new object (Figure 7.15 (vi)). Three rules applied colour materials to the Garden-Wall objects. The `colour-roadside-wall` and `colour-boundary-wall` rules applied the material of the nearest House to the Garden-Wall object. The `colour-shoreline-wall` rule set the material of shoreline walls to a natural stone material. After the Hedge objects, the remaining vegetation missing from the working-scene included the various types of Trees present in the Southcourt Avenue landscape vocabulary.

7.4.6 Trees

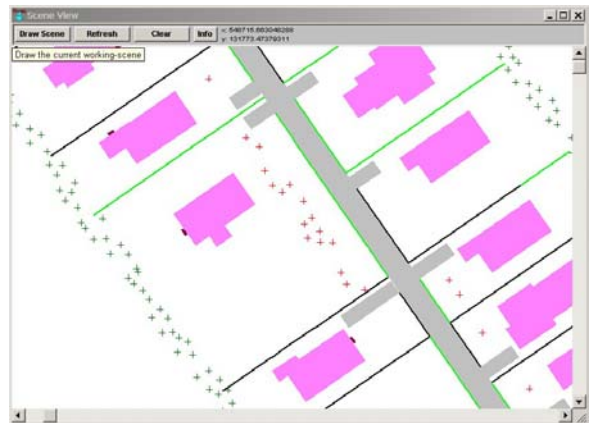
The Southcourt Avenue neighbourhood is flanked on both sides by a mixed belt of trees. These trees separate it from adjacent neighbourhoods. Additionally there are trees located within the core of the parcels on the front, sides and rear of the houses. In the front of homes, there is sometimes a single tree or pair of trees planted at or near the middle of the front yards. In the rear, when trees occur they are often fruit or palm trees, and occasionally a small patch of banana trees is found as well. The species of trees were presented in Table 7.1 and generalized as classes in the Southcourt Avenue vocabulary.

Tree object locations were not very difficult to model grammatically as their point geometry is very simple. The overall approach was to identify a point location and then select the subclass of Tree to be instantiated there. Rows of Trees were created by dispersing points along a polyline. For forested areas, a polygon was defined and random Tree points were located within it providing for desired densities and minimum distances from other objects. The species subclass of every Tree was selected according to the frequencies with which they occur in the study area. In every rule, each new Tree was assigned an elevation value calculated from the elevation grid layer (c.f. the method used in Mayall, 1993).

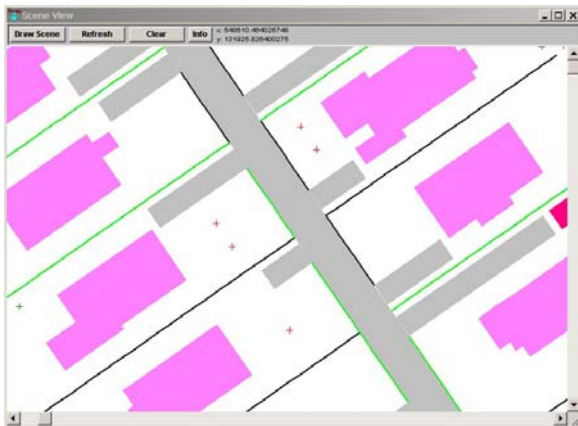
Thirteen rules comprised the `add-trees` ruleset. Two rules added a random mix of Trees



(i) area-boundary-trees



(ii) front-yard-trees



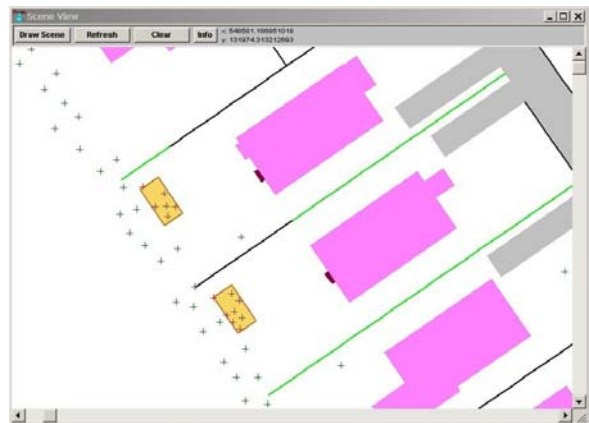
(iii) front-yard-trees



(iv) define-back-yard



(v) back-yard-trees



(vi) create-banana-patch

FIGURE 7.16 GENERATION OF TREE FEATURES FOR SOUTHCOURT AVENUE

around the neighbourhood's perimeter. The `area-boundary-trees` rule created a two-metre buffer around a Lot-Line on the periphery and then randomly created Tree points within the buffer polygon at a density of fifteen Trees per 100 square metres and at a minimum distance of two metres apart (Figure 7.16 (i); the actual tree density may be higher but large numbers of trees could not be displayed in the GDS software). Each Tree was created as an instance of one of the Misc-Tree subclasses. A similar rule, `shoreline-area-boundary-trees`, performed this function but with no Trees located near the shoreline and a higher probability of Baygrape trees occurring with proximity to the shore (orange points in Figure 7.16 (i)).

Further rules generated Trees in the front and back yards of each developed Land-Parcel. With each rule, the Tree objects were only created probabilistically according to the frequencies of their patterns found in Southcourt Avenue. That is, the firing of a rule may have created no trees at all. The `front-yard-trees` rule inserted Tree objects in the front of a House with a probability of forty percent. If the front yard was elongated, then a narrow polygonal area containing random Trees was generated (red points in Figure 7.16 (ii)). Otherwise, a row of zero, one or two Trees was created (red points in Figure 7.16 (iii)). A similar rule, `front-yard-trees-near-main-road`, performed the same function with Road-Edges labelled `the-main-road`, but with a lesser probability of having zero trees created.

In the rear of a Land-Parcel, the `define-back-yard` rule constructed a Back-Yard polygonal object that extended from the rear of a House to the rear Lot-Line in the edges of the Land-Parcel (Figure 7.16 (iv)). Additional specializations of this rule included `define-back-yard-near-main-road` and `define-back-yard-at-waterfront`, as has been seen in other rulesets. The `back-yard-trees` rule applied to any Back-Yard object and inserted Tree objects randomly within the area of the Back-Yard, allowing for a minimum distance from any House and from any other Tree (red points in Figure 7.16 (v)). A similar rule, `create-banana-patch`, was applied to any Back-Yard that met minimum area and maximum average slope requirements. If such a large flat Back-Yard was identified and the rule was selected to be fired, the rule's consequent identified an area of the lowest elevations in the Back-Yard and constructed a small rectangular polygon at its mean centre point. The rectangular polygon was then rotated to align with the edges of the Land-Parcel and repositioned slightly towards the nearest rear corner of the property. The polygon was then converted to a Banana-Patch object and filled randomly with Banana point objects (Figure 7.16 (vi)).

The `add-trees` ruleset also contained some cleaning rules to remove Trees from invalid locations. The `remove-trees-from-banana-patch` rule removed Trees, other than Bananas, that fell inside a Banana-Patch object. The `remove-trees-from-road-areas` rule removed Trees from the central Road-Area or a Driveway. The `remove-trees-from-buildings` rule removed Trees from inside a House or Garage. The `remove-trees-from-walls` rule removed Trees that may have

inadvertently been inserted near the edge of a Land-Parcel and inside a Garden-Wall object.

After applying the `add-trees` ruleset to every Land-Parcel in the working-scene, the interpreter returned to the World ruleset. At this stage, there were no more rules in the World ruleset that were applicable to the working-scene. The interpreter therefore ceased to process rules and output the final scene of landscape objects. The final scene generated by this particular interpretation of the Southcourt Avenue landscape grammar contains 1,440 objects and is shown in Figure 7.17. The landscape grammar interpreter completed this generative process in 1 hour 50 minutes using 1,962 iterations.

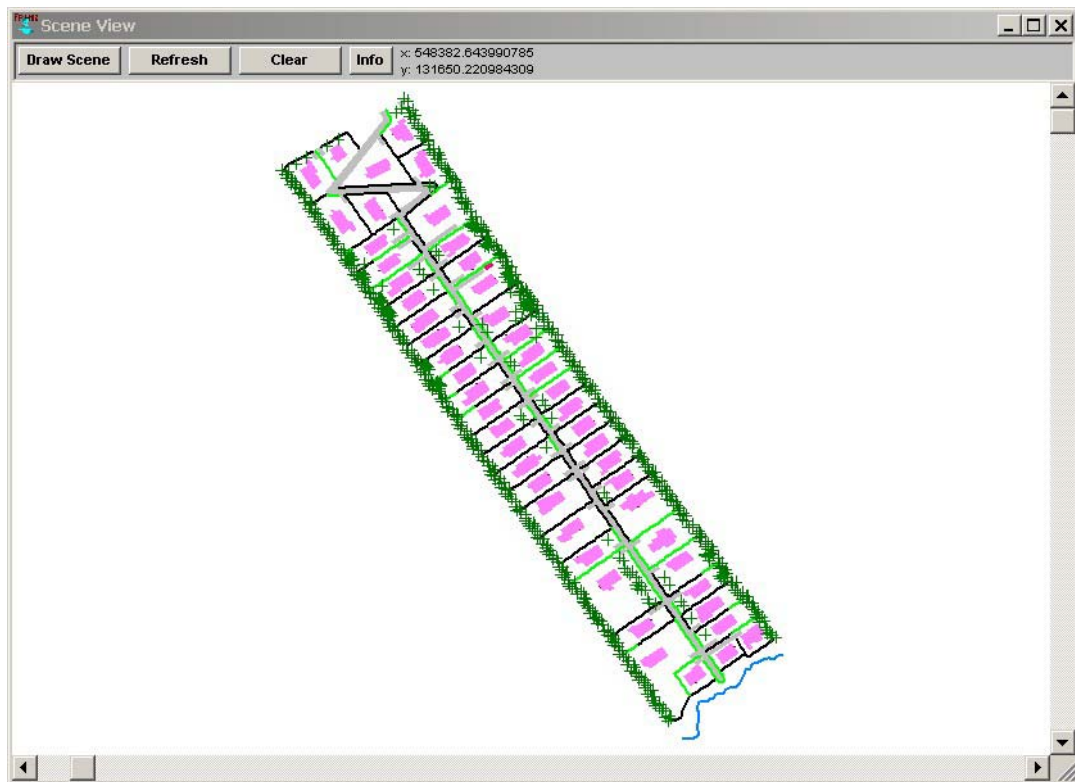


FIGURE 7.17 FINAL WORKING SCENE FOR THE DESCRIBED GRAMMAR INTERPRETATION

This section has explained step-by-step in some detail the content of the Southcourt Avenue landscape grammar rules and how they were operationalized to produce the final scene as described above. The resultant scene is a database of geospatial objects representing a possible landscape that conforms to the Southcourt Avenue landscape grammar. As the grammar defines a landscape character, the generated scene embodies the spatial aspects of that defined character.

Because there are randomized elements in the grammar, it is possible to generate other scenes from the same grammar and initial scene of four landscape objects and an elevation grid. Because they are generated by the same grammar rules using the same vocabulary, such additional landscapes also

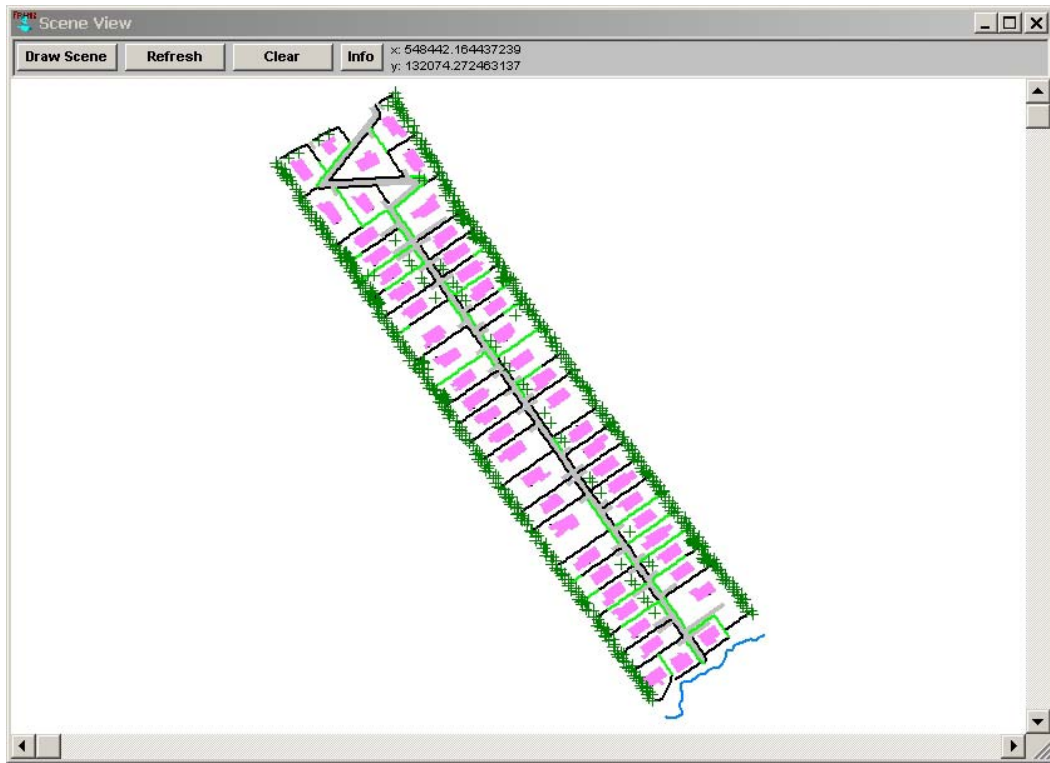
embody the Southcourt Avenue landscape character. Examples of alternative scenes are presented in Figure 7.18 and were generated from the same initial scene shown in Figure 7.8 and without any modifications to the landscape grammar itself. Generation times ranged from 85 to 120 minutes for each scene.

To this point in the prototype, the LGS software application was used to store a landscape grammar and use it to generate 2D landscape databases with spatial and non-spatial content. While the results from the LGS software application are useful, further enhancements are necessary to fulfil the needs of visual landscape planning. The LGS application does not represent the visual resource in the 3D form customarily experienced in the actual landscape. To facilitate this, the final scene generated in the step-by-step example above was exported to files formatted for use in the GDS 3D modelling software environment.

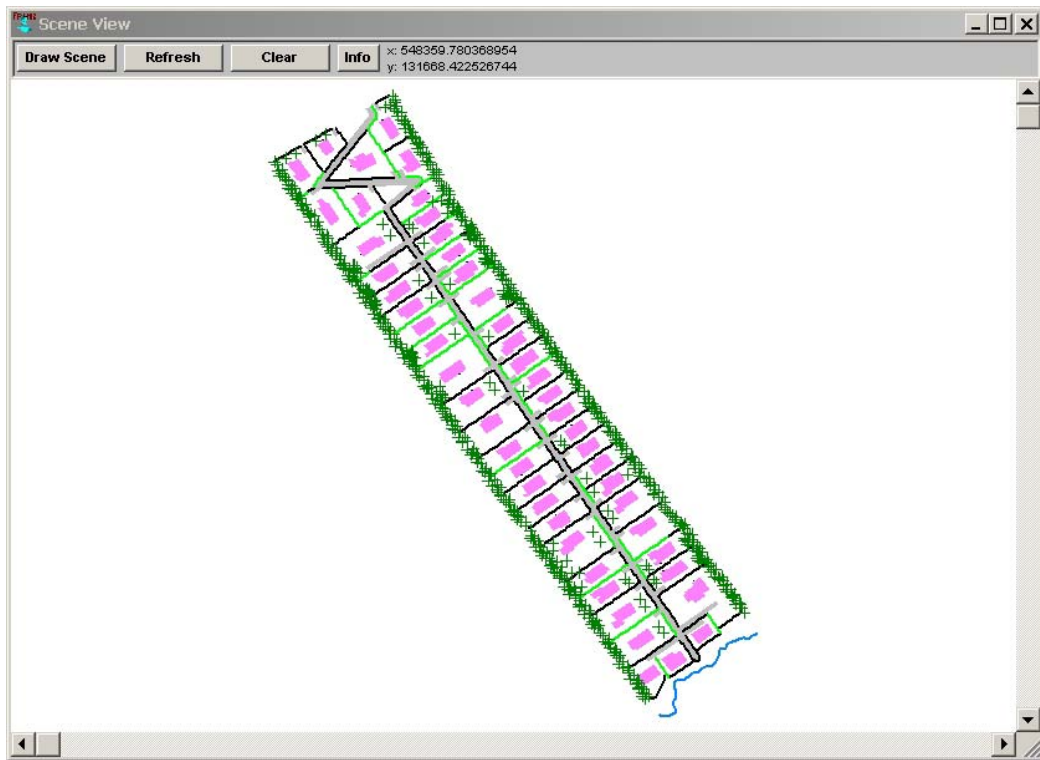
7.5 3D Visualizations

The final scene described above was exported from the LGS software application as data and script files for the GDS 3D modelling software, as was explained in detail in Section 4.8. All data files were created in the GDS 3D THINGS file format and all script files contained commands in the GDS scripting language. The following export procedure was conducted for each final scene. All export routines were custom developed in the LGS application. First, the elevation grid was output to a 3D data file as a terrain solid. The Road-Area objects (including Driveways) were then output to a data file and a script was generated to cut their shapes into the surface of the terrain solid in the GDS Solid Modeller module. The terrain solid for Southcourt Avenue had to be divided into four smaller solids in order to ease this cutting operation in GDS. Building objects were exported from GDS to 3D data files as solids extruded from sea level to the rooftop elevation attribute values calculated by the grammar rules. At the same time, the export routine created a GDS script to cut a roof into each Building using the roof slopes and materials contained in its attribute data.

Chimneys and Garden-Walls were both extruded from sea level to the heights calculated for them by the grammar rules. The linear Hedge objects were exported to a GDS script that defined the Hedge's path in the GDS Solid Modeller as a 3D line, and then swept a vertical Hedge shape of the appropriate height along that line thus creating an elongated solid. The Tree objects were output to a command script that recalled for each Tree the appropriate predefined 3D shape for that class and placed it at the correct location in the 3D scene. Since the 3D versions of the Trees were vertical and flat, another routine was exported to be used at any time in the GDS Scene Viewing System to rotate each Tree around its z axis to face the observer. Following all of these thematic outputs, the LGS export

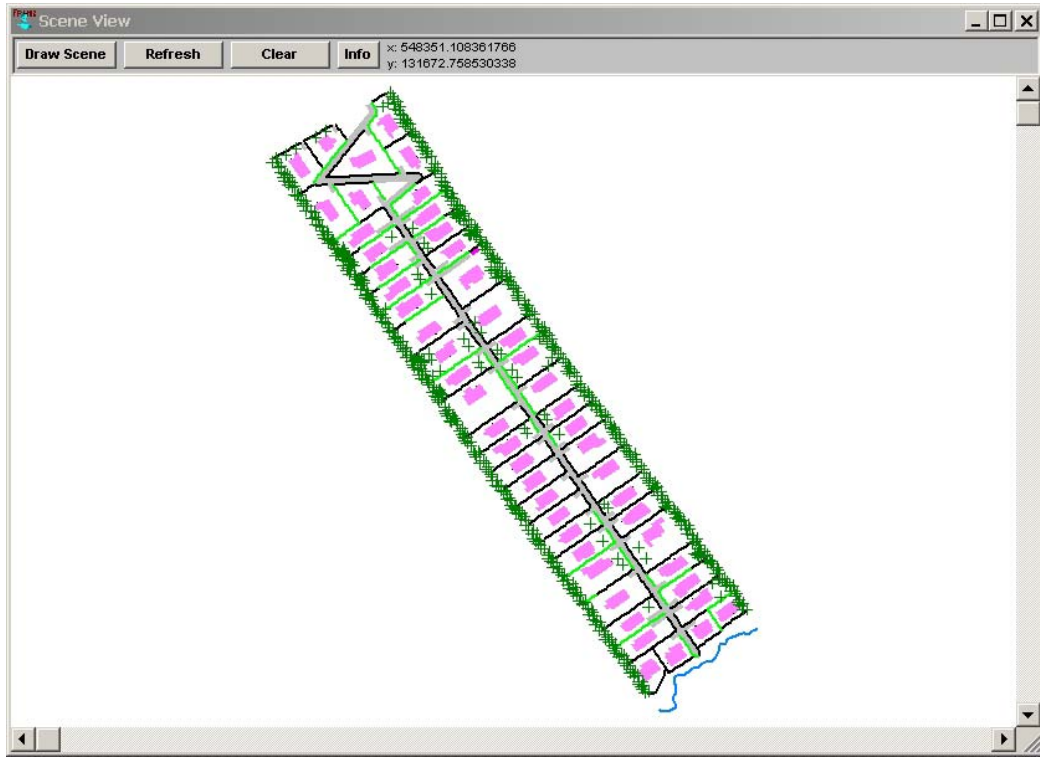


(i)



(ii)

FIGURE 7.18 ALTERNATIVE SCENES GENERATED BY THE SOUTHCOURT AVENUE GRAMMAR



(iii)



(iv)

FIGURE 7.18 (CONTINUED)

routine also created a master script that, when run in GDS, executed in sequence all of the scripts described here. All materials, including those randomly created for the House colours, were exported to a script file to recreate them in GDS. Instances of the abstract landscape classes (Road-Centreline, Lot-Line, Wall-Line, Boundary-Marker, Visual-Barrier, Land-Parcel, Back-Yard, and Banana-Patch) were not exported for modelling in GDS because they had no physical representation.

The completed 3D model was assembled and rendered in the GDS Scene Viewing System. Various views of the model are shown on the following pages. In these and subsequent 3D views, the landscape surrounding Southcourt Avenue is also displayed in order to provide a visual context for the neighborhood. The surrounding buildings, constructed from GIS data of the actual landscape, are shaded in grey in order to distinguish them from the grammar-generated buildings.

Figure 7.19, Figure 7.20 and Figure 7.21 provide a visual comparison of some of the original site photographs with the 3D scene generated by the grammar. Figure 7.22 provides additional perspective views of the 3D scene. The irregular lots surrounding the curving road in the northern end of Southcourt Avenue are shown in Figure 7.22(i). This figure also shows the performance of the grammar in the steeper and more awkward terrain. Figure 7.22(ii) illustrates the levelling of the terrain within Land-Parcels. Because this levelling produced sharp changes in elevation between adjacent parcels in this part of the site, retaining walls are situated between them (pink and green in the centre of Figure 7.22(ii)). Figure 7.22(iii) and (iv) exhibit some of the various Tree objects in the scene. Figure 7.22(iii) shows the placement of one or two Trees, specifically Palmetto, Loquat and Cedar, in the front yard of selected properties. Figure 7.22(iv) shows the randomly placed Trees in a back yard, in this case, instances of the Coconut-Palm, Citrus, Loquat and Palmetto classes. Small patches of Banana trees can be seen in the back yards of the foreground parcels of Figure 7.22(iv). The foreground properties also display different Driveways: one that allows parking near the road only, another that extends to the house, and another that extends past the house with a Garage at its terminus that mimics the material of the relevant House object. Figure 7.22(v) presents the upland part of Southcourt Avenue looking south to visualize how the terrain, and House objects situated on it, slope downward disappearing from view as the observer looks southward towards the ocean. The ocean view and the visual impact of intervening houses are also depicted in Figure 7.22(vi) from the perspective of a single property on the crest of Southcourt Avenue. This section has demonstrated the translation of the 2D landscape grammar outputs into a 3D form that allows better visualization and exploration of the scene in a manner that is more complementary to an observer's real-world landscape experiences.

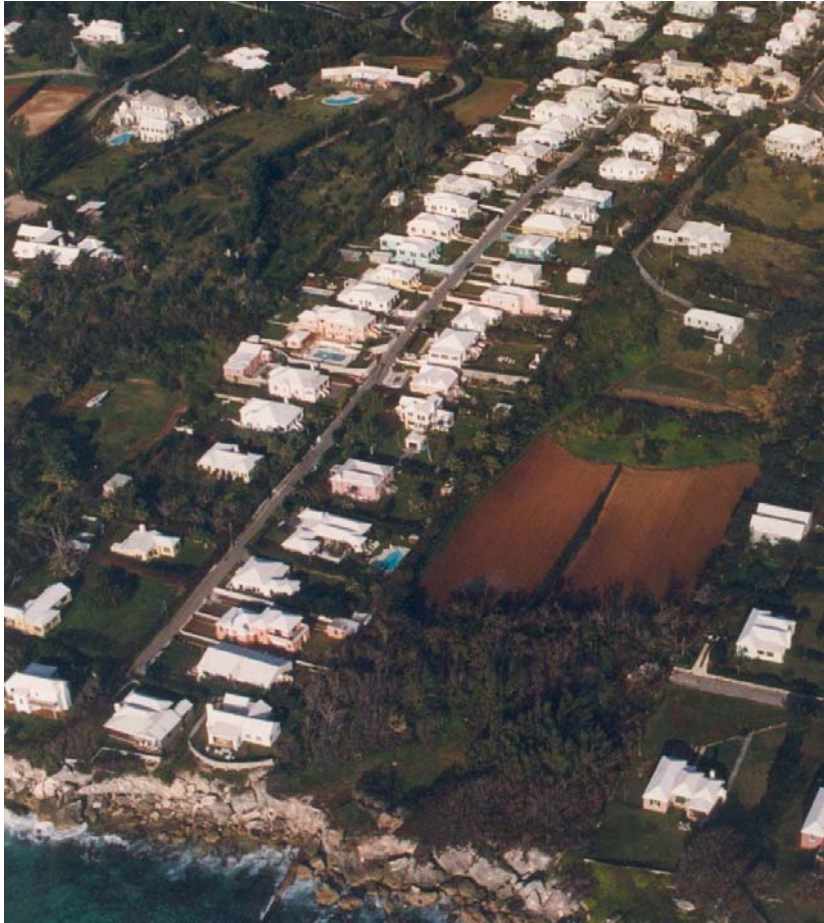


FIGURE 7.19 AERIAL VIEW OF SOUTHCOURT AVENUE AND GENERATED 3D SCENE



FIGURE 7.20 OCEAN VIEW OF SOUTHCOURT AVENUE AND GENERATED 3D SCENE

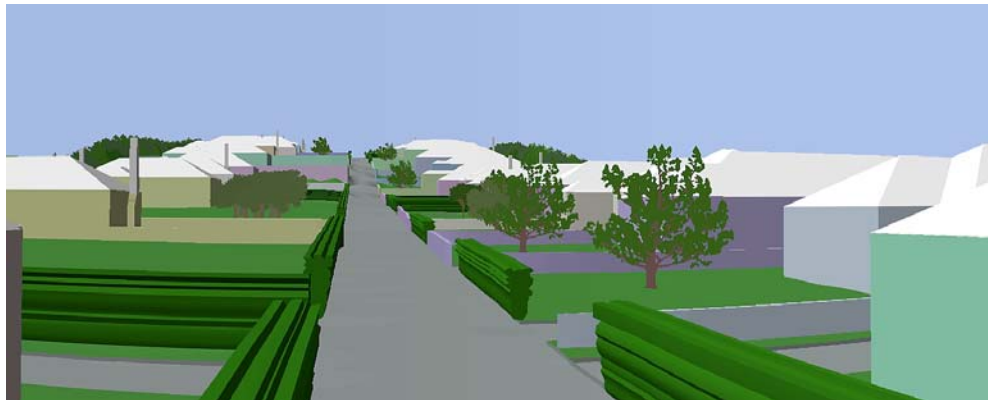


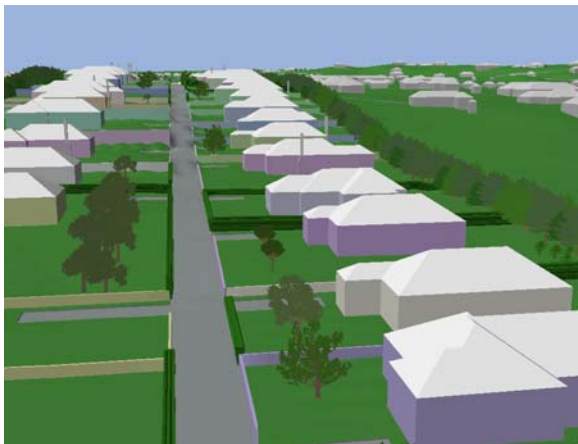
FIGURE 7.21 ROADSIDE VIEW OF SOUTHCOURT AVENUE AND GENERATED 3D SCENE



(i) Curving road



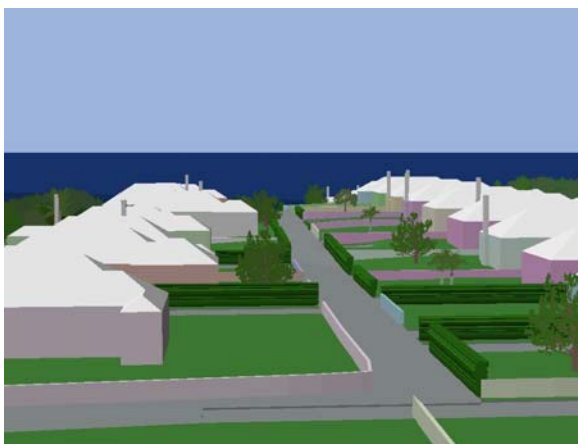
(ii) Levelled parcels and retaining walls



(iii) Front yard trees



(iv) Back yard trees, banana patches and garage



(v) Upland end of Southcourt Avenue



(vi) Ocean view from a house

FIGURE 7.22 3D VISUALIZATIONS OF GRAMMAR-GENERATED SCENES

7.6 Planning Scenarios

The results described above show the potential of landscape grammars to define a landscape character as a landscape grammar in terms of its cultural and ecological elements. The landscape grammar can then be used to generate landscape scenes that exhibit the defined character and the generated scenes can be visualized in 2D and 3D forms. While this demonstration is useful in itself, it was noted in earlier chapters that planning regulations can be amenable to representation in a grammatical format and therefore be used by the grammar interpreter to influence the landscape character. The resultant visualizations from such scenes illustrate the possible spatial and visual effects of planning regulations as embodied in the landscape.

This section demonstrates the effects of modifying the Southcourt Avenue grammar with new regulatory parameters and rules to explore the visual implications of planning regulations from the 1992 Bermuda Plan. Table 6.3 in the previous chapter presented some of the development regulations for the Residential 1 (high density) zoning under the 1992 Bermuda Plan. To summarize for the purposes of this section, Residential 1 zoning requires, for detached houses, a minimum lot size of 6,000 square feet, a maximum site coverage of 35%, a setback distance of twenty feet from an estate road and ten feet from a lot line, and a maximum building height of two storeys. Although Southcourt Avenue was developed prior to the 1992 Bermuda Plan, it is currently zoned as Residential 1.

The Southcourt Avenue landscape grammar, as presented in the previous sections, was written to describe the existing landscape character of the site. To demonstrate the incorporation of planning regulations, the grammar was modified to include the standards embodied in Residential 1 zoning. Three development scenarios were explored and are described below. In each case, the modified grammar was re-executed in the LGS interpreter generating new landscapes that illustrated different development scenarios permissible under the Residential 1 standards. Figure 7.23 shows these landscape changes as 3D visualizations in the GDS software.

The first and simplest scenario was to increase the height of the buildings to the maximum permissible height of two storeys. In the grammar, a House object is created after a nested interpretation of the `house-additions` ruleset has generated the desired Building-Additions objects. The assignment of an overall height in this rule was modified to assign a height of six metres instead of three metres. The generated scene is shown in Figure 7.23(i) and (ii). Alternatively, the rule could randomly assign building heights of three or six metres to generate a landscape with perhaps more realistic variability.

The second scenario explored the implications of development using much larger land parcels within the Southcourt Avenue neighbourhood. On Southcourt Avenue, the land parcels are generally between 6,000 and 12,000 square feet in area. In relation to the Plan's requirements for 6,000 square foot



(i) Second storey permitted



(ii) Second storey permitted



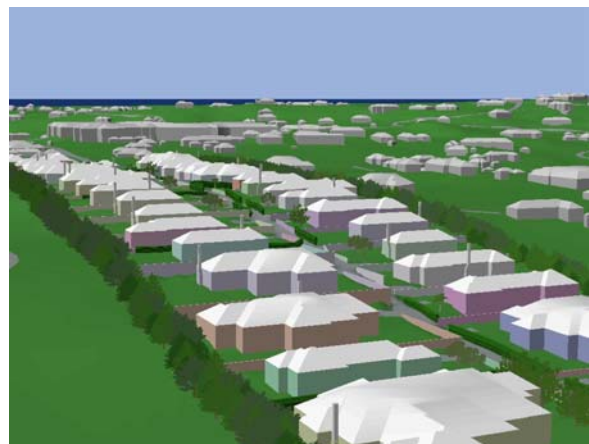
(iii) Large minimum parcel area required



(iv) Large minimum parcel area required



(v) Minimum setbacks and maximum site coverage required



(vi) Minimum setbacks and maximum site coverage required

FIGURE 7.23 CHANGES IN LANDSCAPE SCENES BASED ON REGULATORY GRAMMAR MODIFICATIONS

parcels in Residential 1 zones, the Southcourt Avenue parcels are considered small but permissible. The grammar does not generate Land-Parcels from the beginning based on their area, but rather places Lot-Lines along a road and later constructs a polygonal Land-Parcel object using the lines as a perimeter. To generate larger Land-Parcels in the working-scene, the **lot-frontage** parameter was increased from fifteen metres to thirty metres. This affected the operation of the *add-initial-lot-line*, *add-perpendicular-lot-line* and *rotate-lot-lines-near-shore* rules.

A landscape scene generated from this modified grammar is shown in Figure 7.23(iii) and (iv). The resultant Land-Parcels ranged generally from 12,000 to 18,000 square feet in area, corresponding to larger Residential 1 building lots but not so large as to be considered a Residential 2 (lower density) zone, which would likely require additional cultural grammar elements such as larger homes, swimming pools and more abundant landscaping. The larger Land-Parcels resulted in more extensive re-grading and more flexible location of buildings on sloping land. An unexpected observation was that the rules of the *house-additions* ruleset generated larger House objects than previously observed for the grammar. This is explained by the more spacious environment provided by the larger Land-Parcels which permitted the *house-additions* ruleset to create Building-Addition objects in a less restricted manner.

The third scenario was designed to address some of the other Residential 1 regulations from the 1992 Bermuda Plan. The intent of this scenario was to model another maximum development scenario in which buildings were allowed to increase in size, within the prescribed setback distances, until the maximum site coverage was reached. The minimum setbacks and maximum site coverage standards (Table 6.3) were incorporated into the original Southcourt Avenue landscape grammar, specifically in the *house-additions* ruleset.

First, a new rule was defined to remove any Building-Addition object situated within ten feet (3.05 metres) of the edge of its Land-Parcel. Second, another new rule was defined to remove any Building-Addition object situated within twenty feet (6.10 metres) of a Road-Edge object. A third new rule was defined to remove Building-Additions within twenty-five feet (7.62 metres) of a Shoreline (this is a general requirement of all development zones to preserve coastal green space). The rule syntax for each of these three rules is very similar. The syntax of the second rule is defined as follows:

```
(define-rule
  :name remove-additions-near-road-edge
  :rulesets remove-bad-additions
  :if ((is-a ?a 'Building-Addition)
      (is-a ?b 'Road-Edge)
      (< (distance ?a ?b) 6.10))
  :then ((remove-object ?a)
        (increment *num-current-failed-attempts*))
)
```

A fourth new rule was defined to remove any Building-Addition object that, when combined with the original House rectangle and existing Building-Additions, resulted in a building footprint area that exceeded the 35% maximum site coverage for the Land-Parcel. The syntax for this rule follows:

```
(define-rule
  :name remove-additions-that-exceed-site-coverage
  :rulesets remove-bad-additions
  :if ((is-a ?a 'Building-Addition)
       (is-a ?b 'House)
       (is-a ?c 'Land-Parcel)
       (> (/ (area (union-polygons ?a ?b)) (area ?c))
          0.35))
  :then ((remove-object ?a)
         (setq *current-num-failed-attempts*
               *max-num-of-fails-on-house-additions*))
  )
```

Finally, it was considered that situations could arise in which a restrictive Land-Parcel shape could not possibly allow a 35% site coverage within the prescribed setback distances. In such a case, the interpreter would loop infinitely adding Building-Additions that would immediately be removed by other rules. In order to avoid this situation, a goal was provided to the interpretation of the house-additions ruleset – once a maximum number of twenty consecutive failed attempts to create a Building-Addition had been reached, the grammar abandoned the house-additions ruleset and continued interpretation.

Each of the above modifications were implemented in the Southcourt Avenue landscape grammar and the interpreter was re-executed to generate new landscape scene data that conform to the setback and site coverage requirements of the 1992 Bermuda Plan. The spatial and visual results of the new regulatory grammar rules are presented in Figure 7.23(v) and (vi). The inability of several of the smaller Land-Parcels to achieve the maximum site coverage can be observed in the generated scene. This was attributable to the restrictive setback distance from a Lot-Line required by the Residential 1 standards. The remove-addition-near-lot-line rule continually removed many of the Building-Additions created for these parcels leading the interpreter eventually to ‘give up’. These smaller Land-Parcels illustrate the impact of the required setback distances on the ability to achieve the allowable site coverage for this parcel configuration. Even when reductions in the sizes of the initial House rectangle and the Building-Additions were later instituted in the landscape grammar, these parcels were still not able to achieve 35% site coverage. The larger amalgamated Land-Parcels in the scene were able to achieve near the maximum allowable site coverage within the prescribed setback distances. Although some surprising building footprints were generated, this is mainly due to the simplistic technique employed for constructing Building-Addition objects. However, it can also be seen how the maximum allowable site coverage can be misapplied on the largest parcels to create exceedingly sprawling houses, suggesting that a progressive site coverage maximum based on developable parcel area may produce

better results.

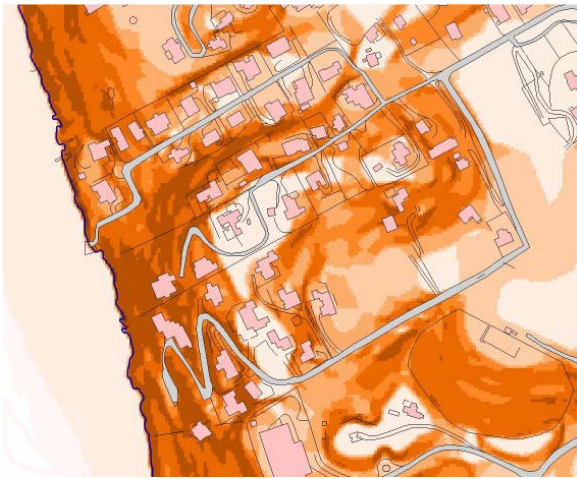
This section has demonstrated that not only can landscape grammars be used to describe a landscape character but also to include planning regulations that influence a landscape's form and character. The implications of the initial and revised grammars are discussed in the following section.

7.7 Discussion

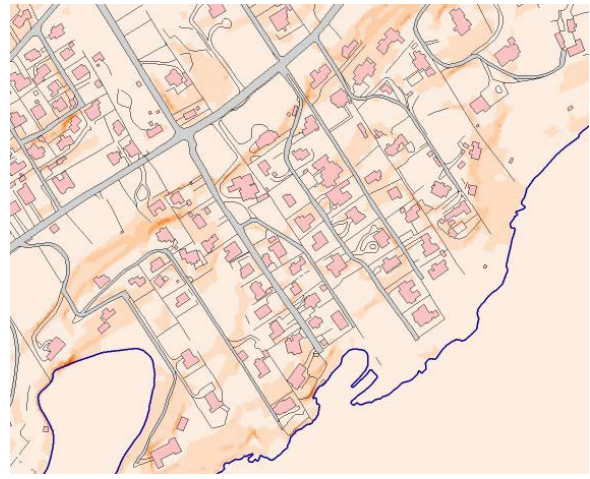
This chapter has demonstrated the use of the landscape grammar theory and LGS software application for a particular residential neighbourhood in Bermuda. The landscape character of Southcourt Avenue was described based on field observation, documented history and planning regulatory characteristics, encoded in a landscape grammar, and that grammar was then used to create similar potential neighbourhoods of the same character. These hypothetical neighbourhoods were visualized as 3D landscape simulations. Further, the landscape grammar rules were modified to incorporate various existing planning regulations and explore the visual consequences of these changes.

The visual results presented in this chapter convey the degree to which the landscape character of Southcourt Avenue was captured in the landscape grammar. Allowing for the differences in the level of abstraction between the site photographs and the 3D model views, some general differences can be identified. For example, the 3D models exhibit less vegetation than is observed on Southcourt Avenue. This was a necessary limitation on the complexity of the scene imposed by the capabilities of the GDS renderer. Similarly, the hedges in the 3D model should perhaps exhibit a larger range of heights. During the design of the 3D modelling techniques, this issue begged the question of whether a particular piece of vegetation was to be modelled as a linear hedge or a group of singular tree objects. It may be argued that the models presented in this chapter use the former method where perhaps the latter could be more effective. Again, this has an effect on the complexity of the scene and therefore the capabilities of the 3D software environment. Finally, more variation in the shapes and sizes of houses is evident in the site photographs than in the generated scenes. This is a consequence of the simplified methodology for adding extensions to houses as discussed earlier in section 7.4.3. Despite these visual differences, it is asserted that the landscape character of Southcourt Avenue has been adequately captured in the case study grammar and serves the purposes of proving the concept of landscape grammars.

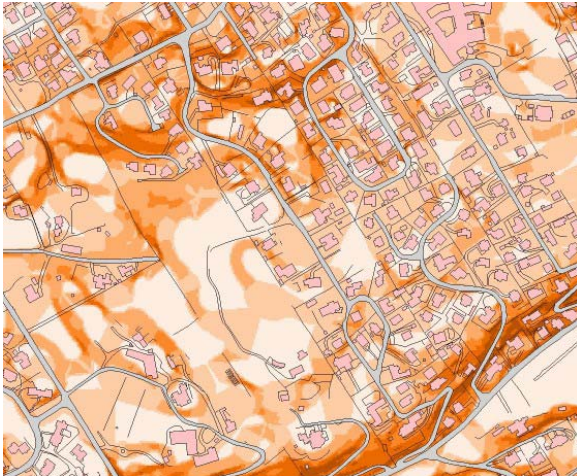
Many of the elements of the Southcourt Avenue grammar are applicable to other residential neighbourhoods in Bermuda. Figure 7.24 shows some other areas in Bermuda that display similar characteristics to Southcourt Avenue (slope values are displayed in orange with lighter shades indicating steeper slopes), where all of the vocabulary classes used are applicable with the inclusion of additional classes. For example, further Tree subclasses would be necessary, although a grammarian could carry



(i) Tribe Road No 6



(ii) Ocean Avenue



(iii) Poinciana Road



(iv) Devon Springs Road

FIGURE 7.24 BERMUDIAN NEIGHBOURHOODS SIMILAR IN STRUCTURE TO SOUTHCOURT AVENUE

species information as an attribute definition instead of hierarchical subclasses. In lower density, wealthier neighbourhoods, the vocabulary would likely include other features such as swimming pools, flag poles, docks and entrance pillars (at the driveways).

Some of the grammar rules are undoubtedly applicable to other areas of Bermuda. The rules that assign colours to Buildings, create Garden-Walls and Hedges, and the various ‘cleaning’ rules are obvious choices. The `plot-roads` ruleset effectively applies to narrow residential areas between a main thoroughfare and a shoreline, and would surely need expansion to allow for other kinds of roads winding around large hills or traversing a relatively flat terrain. Likewise, for lower density neighbourhoods, the construction process for Buildings would likely need rules to allow for more extensive Houses and the

addition of swimming pools.

The strategies employed in applying the rules may also be different in other neighbourhoods. Some residential estates evolved from a single large home on a sprawling parcel of land, to the incremental creation of additional cottages on its periphery, to a more complete subdivision of the estate including higher density townhouse-style development. In other scenarios, the grammar might start by culling trees from a forested area and retaining the largest and most desirable trees which influence the location of development around them.

By and large, the Southcourt Avenue landscape grammar provides a reasonable case study that is extensible to apply to the rest of Bermuda. Its classes and rules reflect back to the Bermuda Department of Planning's definition of the "Bermuda Image" that was presented in Figure 6.7. Particularly, the grammar exhibits the location of buildings based on topography, the traditional building form with white pitched roofs, and the use of subtropical vegetation, stone walls and roadside hedging. Relative to the overall goals of the dissertation, the Southcourt Avenue case study has shown that it is possible to encode landscape characteristics in a grammatical form and subsequently use them to generate and visualize new landscape scenes conforming to the character definition.

Despite fulfilling the dissertation objectives, several practical improvements are apparent in the design of the interpreter software and the landscape grammars themselves. Insights derived from the case study relate to the demands of the landscape grammar and interpreter on the GIS data model and functions of landscape grammar software. Further, techniques for designing rule syntax and the functioning of the grammar interpreter are also apparent. These observations are discussed below in the interests of future developers of landscape grammars and interpreters.

The LGS implementation was developed as a proof-of-concept application. The GIS functionality was entirely custom written because of the inflexibility of commercial GIS software at the time the application development began. The GDS 3D modelling techniques were adapted from previous research, however, during the research, the GDS software was decommissioned as a commercial product. In light of this, further development of the landscape grammar interpreter should use more recently available software environments for managing knowledge-bases, GIS and 3D modelling and visualization. New implementations would ideally incorporate all of these elements in one software system or a cluster of systems interfacing directly with each other. Principally, the 3D modelling environment should be tightly coupled or, at best, integrated with the grammar interpreter, while continuing to allow for many of the grammar rules to operate in two dimensions. If the rules operated directly within a 3D modelling environment, rather than relying on intermediate data and script files, more options would be available to the rule syntax to influence the 3D forms. However, because many 3D environments do not offer the topological data structures and analytical tools of a GIS, tradeoffs may

still be necessary in overall system functionality.

The reliance on intermediate data and script files also had consequences on the implicit and explicit representation of knowledge in the landscape grammar. While the grammar rules discussed in this chapter are explicit representations that were interpreted by the LGS application mechanisms, some interpretation was also hidden in the export routines. In transforming the GIS data from LGS into 3D data and script files some assumptions were made in the generation of 3D forms. For example, because of the scripted manner in which roofs were constructed from buildings in GDS, it was assumed that all roofs were entirely hipped. Ideally, a grammar rule would be able to explicitly state which walls of a house have hipped or gabled roofs above. This is remedied somewhat by the use of 3D data as attributes in LGS, but providing grammar rules with access to full 3D manipulation functions (as proposed in the previous paragraph) would eliminate the need for this implicit interpretation of the GIS data.

As stated above, the LGS implementation was designed to demonstrate the concepts of landscape grammars. Consequently, the 3D landscape models presented in this chapter were simplified. Future implementations could utilize recent 3D software environments to generate models with more realistic appearances to aid visualization. For example, a library of tree images and building construction details could be compiled to this end to aid the insertion of standard objects by grammar rules. Such a library could be composed of 3D solid objects or texture mapped images. Furthermore, grammar rules could be used to modify images used as texture maps. For example, a ruleset could arrange images of windows on a composite image of a wall façade that, during visualization, is mapped to a vertical face of a 3D house object.

It was also clear from the research undertaken that landscape grammar rules utilize many different expressions of spatial relationships between objects. The differences can be subtle, for example, does an ‘intersects’ relationship apply only when two polygons touch each other’s perimeter? These variations in the basic spatial relationships between pairs of objects need to be implemented systematically. A cataloguing of fundamental spatial relations as the “9-intersection model” was developed by Egenhofer (1990, 1991) and has since become available in some commercial GIS software (for example, Oracle 2002).

Depending on the circumstances in the scene, it may be more appropriate for certain grammar rules to analyze or manipulate landscape objects using particular data representations or operations. In the Bermuda case study, rules quickly switched between polygon and line topologies, network operations, grid operations, vector geometry, and angle measurement. For example, Land-Parcels were sometimes treated as a connected chain of polylines (during creation and when measuring distances) while in other situations they were treated as areal objects (for subdivision and amalgamation). Landscape grammar development seems to require a fluid transition back and forth between different data representations.

Any software environment used to build a new implementation should facilitate this fluid exchange without requiring a user to move between software modules. Similarly, human spatial reasoning is very fluid in the deconstruction and reconstruction of spatial objects as required by the circumstances or task at hand. A consideration that treats a polyline as if it were split into two, or a chain of polylines as if they were one, happens fluidly in human cognition. However, rule syntax in landscape grammars must account for these operations explicitly.

Experience with the Bermuda landscape grammar also showed that the maintenance of topological relationships between objects in the scene is important for comparisons made by rules. It was also shown, in the case of contiguous hedges and walls, where the use of line-node topology caused the undesirable sharing of attribute data at the common node (hence the `detach-hedges-from-walls` rule). A useful enhancement to the data structure of the scene would be the development of ‘selective topology’ in which a user could specify which classes share topological relationships and which remain separate even if they occupy the same location.

The use of elevation data in the landscape scene is fundamental to the operation of landscape grammars. However, some difficulties can arise with terrain processing and representation. The grid representation is very easy to modify but can provide a dense and awkward 3D terrain model that exhibits the grid cell structure. Furthermore, the determination of elevation values for specific point locations can be problematic unless it is enhanced with interpolation routines or a very fine grid resolution is used. The alternative is to represent elevation data as a triangulated irregular network (TIN). TIN representations allow better modification of the terrain (not restricted to the shape of grid cells) to simulate excavation and fill operations. However, this representation also requires more advanced data handling capabilities for storage and modifications, as it is computationally complex to insert and delete elevation points in the TIN surface. If TIN surface modification is provided by a commercial software environment, then this problem may be resolved.

Another problem with grid-based elevation data arises from the granularity of the grid. The modelling of small landscape features can require a fine resolution of elevation values. Fine resolutions of elevation data are not generally available unless a specific topographic survey has been performed of the area. However, even where such data are available, their use can slow immensely the performance of the grammar interpreter when matching rules whose syntax examines elevation values. Thus, the impacts of the resolution of elevation data need to be considered in landscape grammar development.

In the LGS software application, rule matching often involved the consideration of all instances of a class in the working scene regardless of their proximity to other key objects. It became apparent that the use of spatial indexing of the objects in the landscape scene would prove useful in allowing a more efficient matching of objects to rules by the interpreter. This was not implemented in the LGS

application but is commonly available in commercial GIS software products.

While the above paragraphs address enhancements for the spatial components of landscape grammar software, other observations from the Bermuda landscape grammar experiences inform the design of spatial landscape rules. First, it must be noted that landscape grammar development clearly requires considerable creativity in the application of computational geometry to the description and manipulation of landscape data. Landscape objects and their patterns are rarely so regular that the rules are straightforward to understand and to write. The irregular geometries of landscape data provide a challenge not found in other generative spatial grammars, such as the orthogonal arrangement of rooms or building construction details or for the generation of decorative geometric shape patterns.

Another noteworthy observation was the importance of the use of object labels in landscape grammars. Labels provided several functions in the Bermuda landscape grammar. Obviously, they provided a system for storing information related to an object even when an appropriate attribute slot had not been defined for its class. More specifically for the grammar interpreter, the labels of an object provided a means with which to mark the objects that had been processed by a rule (e.g. the use of labels such as “amalgamated” and “has-driveway” on Land-Parcels). Such labels helped to direct the sequential flow of rule interpretation from one ruleset to another. Similarly, labels were used to prevent a rule from firing on an object more than once. Rules using this technique often contained in the antecedent a predicate such as (`is-not-labelled ?a 'done-processing`) with a counteracting statement in the consequent such as (`set-label ?a 'done-processing`). Labels can also be used to avoid time-consuming computations. Rather than recalculating the slope values along a Road-Centreline each time a rule required that information, Road-Centrelines were labelled as “steep” or “gentle” upon the first such calculation. Prolific use of object labels can thus make rule processing much less complicated and more efficient.

Some rules in the landscape grammar for Southcourt Avenue were generally applicable to the rest of Bermuda. Others were even more widely applicable to other jurisdictions, including, among others, the `remove-lot-line-from-road`, `remove-tree-from-road` and `remove-tree-from-building` rules. Given this, there is perhaps potential for some ‘common sense’ rules that are always applicable. Experiences with the landscape grammar presented in this chapter suggest that many of these rules are ‘cleaning’ rules that remove nonsensical object patterns.

Landscape grammars involve the expression of many spatial and non-spatial relationships between objects. In the Bermuda grammar rules, many of these relationships were expressed in explicit programming terms within the syntax of a rule’s antecedent and consequent. It is suggested here that future landscape grammars extend the LGS application to provide the ability to define relationships between objects. These user-defined predicates can remove some of the programmatic complexity from

the syntax of the landscape rules making them more readable. For example, a “flows into” relationship could be defined between two river objects, thereby hiding the program code that performs the necessary topological analysis operations and allowing readable rule syntax such as (`flows-into ?a ?b`).

Some further experiences with the case study inform the design of interpreter mechanisms. In the theoretical discussion of landscape grammars in Chapter 3, potential strategies for selecting one rule among matching rules and one set of objects among matching sets were identified. However, rule and object selection strategies did not play a prominent role in the Bermuda case study. Rule selection strategies were sometimes used to ensure that particular rules would be selected over others as a matter of priority, such as the selection of which of the potential Driveways to remove first. However, apart from this, the `least-recently-fired-rules` strategy helped to ensure that one rule was not fired continuously and the `first-rule` or `random-rule` strategies were used as a last resort to select one matching rule over the others.

Object selection strategies played even less of a role in the Bermuda grammar. Apart from the use of the `first-environment` and `random-environment` strategies, the only use of an object selection strategy was in the selection of parcels to amalgamate. In this case, the strategy ensured that the `select-parcels-to-amalgamate` rule favoured those sets of matching Land-Parcels whose mean slope value was less than seven percent. This biased the rule towards flat parcels over sloping ones. It is not clear why the strategies turned out to play a lesser role than expected in theory. It may be because the scene was modularized into Land-Parcels and therefore a ruleset often operated solely within each Land-Parcel rather than having to choose among a larger set of objects. The use of rules to remove objects from consideration (e.g. the Driveways and Building-Additions) rather than using an object selection strategy to steer the interpreter towards certain objects may also contribute to this observation.

Where rule selection strategies were used, however, they were found to be essential as a prioritizing mechanism. An LGS function was written as a rule selection strategy to select from a list of rules in the order of a specified priority – for example, (`rules-of-highest-priority list-of-rules (rule1 rule2 rule3)`) returns either `rule1`, `rule2` or `rule3`, in that priority, if they occur in the list and all other rules in the list if they do not. Another approach could be to assign numeric ranks to the rules as a priority index. This may become difficult, however, where rules are shared by multiple rulesets.

During the development of the Bermuda landscape grammar, it became evident that there were different possible approaches to directing the flow of rule processing. The use of modular rulesets allows rules to be segregated into closed sets thereby controlling which rules are considered by the interpreter for firing. Rule selection strategies, as discussed above, also provide a selective mechanism for firing particular rules before others. The use of object labels has also been mentioned as a means with which to ensure that a rule does not fire on an object more than once. Finally, the LGS application also used

particular functions, such as `rule-has-fired` and `rule-has-fired-with-objects`, to directly test for previous firings of a rule. These functions were used in a rule's antecedent to ensure that for the rule to be considered for matching, another rule must have or must not have been fired previously. Recognizing these various approaches to rule sequencing should aid future landscape grammar developments.

The use of thematic rulesets in the case study allowed for the modularization of the grammar for development purposes. Each of these rulesets was provided to the interpreter with a set of objects as a nested interpretation within the main `world` ruleset. The sequence of interpreting modular rulesets was necessary for development and testing of parts of the landscape grammar. However, it may have been more realistic for various rules to apply to objects on various parcels at the same time, instead of a linear progression through all parcels with a given ruleset. In this manner, one Land-Parcel may be fully developed and landscaped before another has begun its development. This is a significant structural departure from the grammar demonstrated here and should be considered for future landscape grammars. Such an approach is difficult to implement, however, until after the thematic rulesets have been developed and tested. The use of nested interpretations was also found to be useful as a mechanism with which to fire a rule repeatedly until it is no longer applicable to the scene. In the Bermuda grammar, the rules supplied to a nested interpretation loop were not necessarily a named ruleset but instead a list of one or more individual rules.

Finally, it should be stressed that the performance of the grammar interpreter, in terms of speed, is an important matter. Landscape grammar development involves a considerable amount of waiting time while rules are processed. The speed of the LGS interpreter was influenced by several factors. The number of rules to be considered for matching was reduced by the use of modular rulesets where applicable. Likewise the number of objects in the scene to be considered for matching was reduced by using spatial modularity or landscape partitioning into smaller sets of objects (such as the objects within a particular Land-Parcel). The complexity of object geometries impacted the performance of geometric functions, such as calculating intersections with the complex Shoreline objects compared to a simpler Lot-Line. Similarly, the complexity of spatial reasoning operations can slow an interpreter's performance and this was aided in the case study by storing the results of a complex calculation as an attribute value or label for future reference. Finally, serial rule and object processing was found to be much slower than parallel methods. In a serial environment, the interpreter matches a rule to, say, five objects from which one is selected for firing. On the next iteration, the interpreter has to find the four remaining objects and select one again for firing again, and so on until the rule has fired upon all five objects. In parallel object processing, the rule is fired upon all five of the objects during the first iteration. In the Bermuda grammar, the interpreter was instructed to fire a rule on all sets of matching objects where the independence of the objects from each other could be guaranteed (for example, when assigning the

colour of a House).

As this section has illustrated, the process of developing a Bermuda landscape grammar as a case study has revealed insights that were not obvious in the theoretical model and its implementation presented in Chapters 3 and 4. Aspects of the design of the landscape representations, rule syntax and interpretation mechanisms have been identified here as practical issues that can impinge upon the implementation of landscape grammar theory. It is hoped that these insights will serve as an aid to future research efforts in landscape grammar implementation.

7.8 Summary

This chapter has provided in detail a demonstration of the landscape grammar theory and implementation presented in earlier chapters, utilizing a landscape grammar for the Southcourt Avenue neighbourhood of Bermuda. First, the scoping activities to define the Southcourt Avenue landscape character were discussed, including field observations, document review at the Bermuda Department of Planning and GIS data sets. Next, the contents of the landscape grammar were presented. A vocabulary of twenty-nine hierarchical landscape classes for Southcourt Avenue were described. The bulk of the chapter discussed the 150 grammar rules organized into several thematic rulesets. The rules were presented by stepping through the iterations of an example scene generation in the LGS landscape grammar interpreter. It was also shown that the interpreter can use the same landscape grammar to produce alternative scenes that conform to the character definition encapsulated in the grammar. To aid the visualization of the landscape character, 3D models were automatically constructed from the 2D scene data and then presented here using various perspectives of the generated Southcourt Avenue landscape. The incorporation of planning regulations was performed by utilizing existing development standards from the 1992 Bermuda Plan. New 3D scenes were generated and displayed to illustrate this capability. Finally, a reflection on the lessons learned from the case study were discussed referring specifically to practical issues of GIS capabilities, rule syntax and interpretation mechanisms for future grammar interpreters.

In this dissertation, landscape grammar theory has been described formally, implemented as a software application, related to a context of planning activities and then applied successfully to a study area in Bermuda. The following chapter revisits the dissertation objectives, summarizes the broader contributions of this research and suggests future directions for further research on landscape grammars.

Chapter 8

Evaluation and Conclusions

This dissertation has now defined, developed, implemented and demonstrated the concept of landscape grammars. Following the introduction of goals and objectives in Chapter 1 and review of relevant concepts and literature in Chapter 2, landscape grammars were formally defined in Chapter 3 as a definition framework for landscape character and an executable mechanism with which to generate possible landscapes. Chapter 4 presented the implementation of a computer-based landscape grammar interpreter tightly coupled with an object-oriented GIS and loosely coupled with an additional 3D modelling environment. Methods for developing landscape grammars and their context in the practices of a planning agency were addressed in Chapter 5. Chapter 6 presented the island of Bermuda as an appropriate case study for landscape grammar development, and Chapter 7 demonstrated a particular landscape grammar developed for a Bermudian residential community, Southcourt Avenue.

This concluding chapter evaluates the theory, implementation and demonstration of landscape grammars, as presented in this dissertation, in terms of the original research goal and objectives. In addition, the contributions of the dissertation to landscape planning, geographic technologies and our understanding of landscape structure are specified. Finally, taking into account the experiences gained from this dissertation, suggestions for further research are proposed.

8.1 Evaluation of Research Objectives

The protection and enhancement of the visual resources of a place constitute an on-going challenge to the planning authorities of many communities. The crux of this challenge is to guide development towards built forms that will not cause detriment to an existing landscape character. To understand and cope with this problem, there is the need for a means to define and model a landscape character, methods for constructing that character definition, tools for storing and using such a definition to visualize its spatial manifestations, and the ability to incorporate alternative regulatory parameters in order to assess their impact on the landscape character.

Arising from this problem statement, the primary goal of this dissertation, as described in Chapter 1, was to develop a means of defining landscape character to allow the investigation of possible landscapes that conform to a prescribed or desired definition. Specific research objectives were stated as

follows:

1. to advance the capacity of conventional geographic information technologies for landscape character planning;
2. to develop a theoretical framework that allows landscape character definition;
3. to construct a computer-based implementation that integrates the landscape character theory into conventional geographic information technologies and planning practice; and
4. to allow, in that implementation, the visualization of the potential consequences of a landscape character definition.

The concept of landscape grammars served as the central mechanism for meeting the research goal and objectives of this dissertation. The structure of a generative spatial grammar was shown, first, to facilitate the definition of a landscape character using the familiar landscape-language metaphor, and second, to permit the visualization of such a character definition by the generation of various scenes as collections of spatial objects. The first objective is broad in scope and will be discussed last.

The second objective was met by the development of a theory of landscape grammar as presented in Chapter 3. The theory was prefaced by the premise that a landscape character is composed of those visual and spatial elements that recur across a region. Consequently, the elements of landscape character have no specific geographic location in the region, but rather a generalized, descriptive form that expresses the types of objects and patterns that can be found. This premise was translated into a theory of landscape character definition using the landscape-language metaphor. While this metaphor is well known as a poetic device, this dissertation utilized it to derive the concept of a landscape grammar as a formal definition that includes a vocabulary of object types and sets of syntax rules defining typical spatial patterns between such objects. The grammar metaphor provided the theoretical framework with which to define a landscape character.

The third objective referred to the integration of the theoretical framework with conventional geographic information technologies and planning practice. It was demonstrated that the grammar formalisms are amenable to implementation in a GIS. The LGS application demonstrated the implementation of an object-oriented GIS which used a hierarchical vocabulary of classes of landscape objects as its central organizational mechanism. To this end, the LGS application functioned similarly to a conventional layer-based GIS in that the scene of landscape objects represented a geographic database and a host of spatial functions were available. However, the LGS application also provided the ability to include a knowledge-base of rules that define the spatial relationships between classes of objects. The integration of a vocabulary of landscape classes and sets of spatial grammar rules within a system of storing and managing geographic data achieved part of the third objective.

The other part of the third objective related to the integration into planning practice of the

landscape grammar theoretical framework. In Chapter 5, landscape grammar structures were more explicitly connected with the activities of visual resource planning. First, landscape character and landscape grammars were re-framed in terms of their natural, cultural and regulatory elements. The planning regulations of a community were thus identified as an influence on the local landscape character. In addition, methods were then suggested for the construction of a local landscape grammar and a range of planning situations were identified in which landscape grammars can be useful. The range of methods and uses from Chapter 5 indicated that the landscape grammar formalism can be integrated with planning activities relating to visual landscape character.

The fourth objective highlighted the role of visualization in planning for landscape character. Once the ability to define a landscape character is provided, the validity of such a definition must be tested through the visualization of landscapes embodying that character. This objective was satisfied through the use of a landscape grammar interpreter, that is, the use of a landscape grammar as a production system for generating simulated landscapes. Chapter 3 discussed formally this mechanism for the selection and firing of landscape rules and Chapters 4 and 7 provided demonstrations using the LGS implementation of a landscape grammar interpreter. By generating a range of possible scenes from the landscape grammar, visualization of the potential spatial consequences of the grammar was achieved thus demonstrating the feasibility of encoding a landscape character in a grammatical form. This in turn means that planners may visualize in two and three dimensions the effects of their regulatory ideas embodied in the grammar before such ideas are instituted as planning regulations.

The first broader objective was to advance the capacity of current technologies for landscape character planning. This objective was met through the achievement of the other three objectives. By developing a landscape grammar theory, implementing it by integration with geographic technologies, and then using that implementation to visualize the consequences of a landscape character definition, the capacity of geographic technologies to be useful in landscape character planning has been improved. Whereas current technologies allow landscape planners to represent and analyze individual landscape scenes, the research demonstrated in this dissertation significantly advances the ability to allow generalized statements to be made about objects and patterns that are more in step with the generalized nature of visual landscape character.

8.2 Contributions of the Research

The dissertation makes several contributions to the field of visual landscape planning, to geographic information technologies, to shape grammar research and to research pertaining to landscape structure. An early contribution in the research was the focus on visual resource planning and the

definition of visual landscape character as a composition of recurring visual and spatial elements. As identified in Chapter 2, most literature on visual landscape resources has been concerned with visual resource management and the assessment of visual impacts of specific development proposals. The distinction for visual resource planning emphasizes planning for the general landscape character and unknown future developments. Some authors have approached the topic of visual landscapes in verbal terms (Lynch, 1976; Jacobs, 1993; Spirn, 1998), while others have devised methods for assessing particular visual impacts on landscapes (Litton, 1982; Smardon et al., 1986; Sheppard, 1989; Struckmeyer, 1994; IEA/LI, 1995). Few, however, have attempted to address formally the topic of landscape character in general. The value of this definition of landscape character is augmented by the recognition that planning regulations are directly related, because they are also defined spatial relationships that often do not relate to a specific location on the ground.

Extending this topic, deficiencies of conventional geographic technologies were identified in terms of their usefulness for defining a landscape character and for planning in general. In particular, the lack of generalization capabilities for typifying objects and spatial patterns was highlighted. The corollary of this realization is that there is an alternative and richer way to represent landscape data in a GIS environment. This research has shown that geographic data models can be constructed to relate specific objects in a scene to their generalized type definitions (vocabulary) and to associate those objects with other objects via the spatial relationships (rules) that govern their location and character. In this way, geographic databases can be enriched with meaningful spatial relationships, beyond those of simple geometric topology, and therefore become more useful to landscape character definition and planning.

The review of the types of spatial grammars in this dissertation considered shape grammars, graph grammars, Lindenmayer systems, and cellular automata together as contributing to the applied perspective of landscape grammars. These disparate types of spatial generative mechanisms have not previously been considered as one form of system in the shape grammar literature. This work draws the links between these systems using the medium of landscape representation in which the utility of an assortment of data models is already well recognized.

The landscape-language metaphor has most often been used in design literature as a device in verbal descriptions of a landscape. The research presented in this dissertation has transformed this metaphor into a formal structure making it more than just a literary device. By relating it to landscape character, the metaphor has been made useful as a theoretical framework for defining the visual character of a place, and as a mechanism for systematically constructing landscapes that portray the defined character. The reformulation of this metaphor into a formalized theory of *landscape grammar* is the major contribution of this dissertation, as this provides the visual landscape planning community with a means to describe series of related landscapes in a general manner.

A further contribution of the dissertation is the demonstration that conventional landscape representation technologies can be redesigned and extended to allow for such generalized statements as vocabulary type definitions and spatial grammar rules. This potentially gives the geographic technology community new areas of functionality with which to experiment and expand the utility of GIS software. The potential role of knowledge-bases in GIS is broadened from the traditional analytic and diagnostic tools (for example in site selection and location-allocation problems) to generative uses in the construction and arrangement of objects for landscape simulations. In addition, it has been demonstrated that geographic knowledge-bases need not be implemented as closed 'black box' expert systems but can instead be open bodies of knowledge that can be inspected, modified and tested.

The benefits described to this point lead to some implications for approaches to landscape planning. Because conventional GIS tools are designed for representing specific inventories of spatial objects, their role in planning has typically been in the analysis of static geographic databases. This may perhaps explain the focus in landscape planning on the impacts of specific development proposals since they can be modelled in a conventional GIS/CAD environment. However, the conventional tools are not effective for modelling planning regulations because of their inability to represent generalized spatial statements. The implementation of landscape grammars provides this facility and therefore new capabilities for planners to model not only physical objects but also their own regulations as influences in the landscape. In acknowledging the impacts of plan-writing on the local landscape, planners are forced to recognize themselves as agents of change and to visualize the potential impacts of a plan in the same manner as those of singular development proposals.

Furthermore, the explicit acknowledgement in landscape grammars of imperfect knowledge and multiple possible outcomes encourages a process of outcome exploration rather than a best outcome prediction. In the continual refinement of the knowledge embodied in a landscape grammar, a role for learning in the development of planning regulations is also emphasized. In addition, this process of knowledge refinement is open to public involvement in the planning process, as members of the community can provide their own ideas of landscape vocabulary classes and syntax rules that they feel should characterize a specific environment. By combining natural, cultural and regulatory elements, landscape grammars provide one potential means for seeking to understand the different manners in which landscapes are structured especially where human influence is present.

8.3 Future Research

The introduction of the theory, implementation and use of the landscape grammar concept provided in this dissertation affords rich opportunities for new research. Future efforts are warranted in

probing further the theoretical concepts of grammars and the nature of landscape knowledge, devising methods for developing landscape grammars, providing new computer-based implementations of landscape grammar interpreters and testing the application of landscape grammars in new environments and scenarios.

The second chapter of this dissertation touched on several research areas that inform the landscape grammar concept. These and other bodies of literature potentially hold more useful insights for landscape grammar theory. It was shown in Chapter 2 that spatial grammars arose from Chomsky's phrase structure grammars in the field of linguistics. The phrase structure grammar is now a relatively old construct and has been revised and compounded since its debut. A review of the more recent advances involving generative linguistic grammars would provide the opportunity to determine whether such refinements are also applicable to the spatial realm and can enrich spatial grammar theory in general.

More specific to the fields of geography and planning, cartographic generalization research was mentioned briefly in the literature review. Closer study of the achievements of cartographic generalization is proposed here as this subject is likely to provide two areas of potential insight. First, cartographic generalization is one of the few geographic examples of a generative expert system using vector objects. Methods used in the construction and placement of objects for cartographic purposes can provide insight into ways for landscape grammars to manipulate objects in a landscape scene. The second source of potential insight involves the field's subfocus of simplifying cartographic shapes for the purposes of visual clarity in maps. For landscape grammars, the utility of these generalization methods is not so much for visual presentation as the enabling of spatial reasoning. In the Bermuda grammar, cases occurred in which a wavy polyline was simplified to a central axis, or a complex polygon to its convex hull, in order to perform further spatial calculations for placing or dimensioning other related objects. In human-based landscape reasoning, these object generalization tasks are performed quickly and easily and for computer-based grammatical mechanisms to mirror these feats, computational generalization methods would prove particularly useful. Generalization methods might also prove useful when developing a landscape grammar by revealing landscape patterns or structures that are hidden by complex object geometries.

The theoretical foundations of landscape grammars should also be revisited by pursuing a purer investigation of the nature of spatial knowledge in humans. The landscape grammar formalism provides a framework for structuring landscape knowledge as a class hierarchy and modular sets of generative decision rules, and a production mechanism for applying that knowledge to a spatial situation. Landscape grammar rules in their verbal form are generally descriptive ("all houses have white roofs"), but when translated to computational rules they become constructive ("if ?a is a house, then set the roof colour of ?a to white"). It is not clear at this point whether this is necessarily the case or whether an alternative

form of rule interpreter can be developed that interprets rules written descriptively. It is likely that descriptive forms of rules may be more restricted than constructive rules in terms of their functionality. Further research might clarify this situation, as well as explore the potential for interpreters to understand and parse rules written with natural language.

In a similar vein, the grammatical construction of landscapes could take other forms than just decision rules. In the grammar developed in this dissertation, some construction techniques were linear procedures contained within the consequent of a rule. Some took the form of optimization routines, such as finding the location of a driveway that minimized the elevation change from the road. This experience suggests that landscape knowledge may not be entirely composed of decision rules. Investigation is needed to examine the different forms or models encountered in landscape knowledge, such as procedures, heuristics, optimizations, constraints, competing agents, or other forms of directed action, and to see if they may be related to one another in terms of the circumstances in landscape design in which they are most often used. Such investigation should be informed in part by research pertaining to spatial knowledge in general, such as that of spatial cognition in psychology or that of Mark and Egenhofer (1994) which linked human interpretations of spatial relations to Egenhofer's mathematical "9-intersection" model for the intersections of spatial objects. Potential research topics of this kind that would inform landscape grammar research include empirical studies of the roles of object labelling, rule modularization, uncertainty, prior operations and shape generalization in human spatial cognition in landscape design.

In addition to studies of the structure of landscape knowledge, further research can be conducted on how to collect such knowledge and thereby develop a landscape grammar. New methods for landscape pattern analysis or new landscape metrics that are useful for grammar research may be identified. There is also potential for research on knowledge acquisition methods related to landscape design in order to devise methodologies for teasing out landscape knowledge in an explicit form from an individual or group. While the case study in this dissertation used the author as the sole landscape grammarian, it was identified earlier that, in line with a participatory approach to planning, the ideas pertaining to a landscape character from different groups should be incorporated in the grammar development. Therefore, participatory methods for knowledge acquisition should be considered as well. The incorporation of public participation practices into the use of GIS technologies is an evolving field of research and advances with conventional GIS technologies may extend to the related purposes in landscape grammars (Nyerges et al., 1997; Laurini, 1998; Al-Kodmany, 2000).

Several suggestions for computer implementations of landscape grammar interpreters were made at the end of the previous chapter. They will not be repeated here but the development of new landscape grammar software is warranted generally using newer software environments in GIS, CAD, and

knowledge representation. The Lisp programming language is still considered useful but better coupling with existing GIS and CAD software should be sought.

There are also further questions regarding the interpretation of a landscape grammar that could be clarified by further research. How may the performance of a rule set be measured in terms of its efficiency or effectiveness given that rules can be written in different ways? Can the similarity of a generated scene with an existing landscape be measured without attempting to have the grammar create an exact replica of the site? Is it possible to identify areas of the working scene in which rules conflict with each other, such as natural, cultural or regulatory rules? This latter question is particularly valuable for landscape planning since it may provide a means to identify potential problem areas in the landscape where cultural activities are competing with natural forces, or planning regulations are at odds with an existing cultural or natural milieu.

There are further applications of landscape grammars that can be explored in order to field examine the theory at work in other circumstances. Following directly from this dissertation, there is a need for closer empirical studies for the modular rulesets used here, that is, road paths, parcel configurations, building shapes, wall construction, and vegetation patterns. This case study presented a broad grammatical model of the Southcourt Avenue neighbourhood, with rulesets touching on a variety of types of landscape objects. In order to apply to the rest of Bermuda, each of these rulesets must be developed with more depth to account for further spatial situations not encountered on Southcourt Avenue. Of course, the landscape grammar theory and tools should also be applied outside of Bermuda in other regional jurisdictions with their own landscape characters. By expanding the areas of application for landscape grammars, it is then possible to examine the macro-level spatial variation of neighbourhood-level landscape regularities. In grammatical terms, the task is to determine which classes and rules are transferrable to other localities, which are unique to an area, and how similar rules vary between regions.

There is the potential also for landscape grammar concepts to be applied in circumstances that are not specific to a geographic region. Brand (1994) described various ways in which buildings grow over time. These observations can perhaps be captured as a formalized knowledge-base which can in turn be utilized in regional landscape grammars. Similarly, it may be possible to translate Lynch's (1984) site planning principles, Alexander's (1977) pattern language, or any instructive landscape design text such as Motloch (1991), into an executable form in order to explore their spatial consequences in simulated landscape scenes. In addition to the representation of design texts, possible historical and archaeological landscapes (both cultural and ecological) might be reconstructed by developing a grammar around known facts of the period as well as unsubstantiated hypotheses. Likewise, visualization of future ecological change might also be modelled using landscape grammar methods. The use of landscape grammars in

natural environments can provide a means with which to examine ideas pertaining to natural order and complexity in the succession or evolution of ecological landscapes. Darwinian grammar rules of natural selection that might inform such an evolutionary grammar would perhaps draw a link between the grammar formalism and spatial evolutionary algorithms (Krzanowski & Raper, 2001).

Finally, recall that the application of landscape grammars to planning openly acknowledges the use of imperfect knowledge, recognizes planning regulations as an active influence on the physical landscape, encourages the consideration of multiple futures and initiates a continual process of knowledge refinement. The roles of outcome exploration and learning in planning practice may well be developed further. As well, landscape grammars can be seen as a spatial tool to facilitate regulatory impact assessment for planning agencies. These approaches to planning activities should also be explored more fully and beyond the use of landscape grammar concepts.

8.4 Conclusion

A desirable visual landscape character is a valuable resource that provides communities with a cultural identity and an appealing image for tourism and other business. It is incumbent upon the planning profession in general to provide communities with policies and regulations that protect and enhance the landscape resource. In order to fulfil this role, it is essential to come to terms with the nature of landscape character and mold our tools to embrace it. The landscape-language metaphor, formalized in this dissertation as landscape grammar, recognizes the generalized nature of landscape character and provides a model for new capabilities in geographic technologies.

This dissertation has given both a normative and practical approach towards firmly establishing a theory of landscape grammar and its applicability in the planning process. A theoretical framework for formally defining landscape character and exploring its spatial consequences has been described, implemented, demonstrated and evaluated. The implementation has demonstrated a successful expansion of the capabilities of already mature geographic technologies. In addition, the work has been related successfully to other types of spatial grammars and to the general activities of landscape planners.

This research should be considered a foundation in landscape grammar theory. The diversity of further research identified in this chapter speaks not to weaknesses of the concept, but to its richness as an encompassing model that has many possible avenues of investigation. For the planning profession, the landscape grammar concept operationalizes the familiar metaphor of language and landscape and demonstrates how our ideas and observations pertaining to landscape character can be integrated with proposals that govern landscape form. Through landscape grammars, planners now have a model with which to not only examine their understanding of landscape structure and complexity, but also see the

planning profession as a medium of change in the landscape. Further, this model allows the spatial and visual impacts of regulatory influences to be revealed visually prior to policy formulation and regulatory enforcement. Planning agencies currently employ a formidable array of geographic technologies suited to the inventory and analysis of specific landscape objects. It is hoped that, as a result of the research demonstrated here, a new generation of technological tools will emerge that provides planners with the ability to work directly with the spatial generalizations that seem to form such a common part of the professional vocabulary. It is perhaps time that not only should planners learn to speak the language of landscapes, but more importantly, planning tools should learn to speak with planners' own vernacular as well.

Appendix A

Functions of the Landscape Grammar System

This appendix contains a catalogue of many of the functions used in the landscape grammar system implemented for this research. The catalogue is organized into the following sections: GIS functions, Rule-base functions and Interpreter functions. The GIS functions include those related to Classes and Objects, Coordinates, Points, Polylines, Vertices, Polygons, Topology, Intersections, Buffering, Angles, Distance, Grids, Randomization, and Vectors.

The functions generally return true/false, a number, or an object. In the following catalog, the name of the function and its arguments are shown in the left-hand column. The right-hand-column contains a description of what the function does. The term “objects” can be interpreted as Point, Polyline, or Polygon objects. For convenience, a Polyline object is abbreviated to “line” and a Point object to “p”. Most functions that use a line object will also accept an ordered list of points instead of the line.

The complete set of LGS functions contains 457 spatial functions, 145 functions for managing classes and objects, 98 functions for managing and interpreting rules, 233 functions that control the LGS interface, 95 functions that allow the import and export of objects, and 116 functions that generally enhance the standard Lisp function library. The spatial functions constitute the LGS spatial function library in Figure 4.2, while the rest constitute the LGS function library in the same figure. The set of LGS functions listed in this catalog were selected to give a meaningful representation of the functions that comprise the system. In addition to the functions identified above, each slot-definition for each class (whether a CLOS, rule, geometric-object, or landscape-object class) creates an accessory function that returns the value of the slot for a supplied object (such as the material of an object).

GIS Components

The GIS functionality of LGS is provided by a large set of functions for defining, describing and manipulating landscape object representations. Some are found in conventional GIS software but many are not. The Classes and Objects functions are significantly different from functions tied to the typically relational data models of GIS software products.

Classes and Objects

This set of functions allow the definition and query of landscape classes, instances of such classes (objects), and their slots and labels (attributes).

Define-class (superclasses class-name slots)	Creates a new landscape class.
Redefine-class (superclasses class-name slots)	Modifies the definition of an existing landscape class.
Subclasses-of (class)	Returns a list of the subclasses of the supplied class.
Superclasses-of (class)	Returns a list of the superclasses of the supplied class.

Geometric-parent-class (class)	Returns the geometric-class (Point, Polyline, Polygon) that is the superclass of the supplied class.
Slot-names (class)	Returns the slots of the supplied class.
How-many (class)	Returns the number of objects in the scene that are instances of the supplied class.
How-many-direct (class)	Returns the number of objects in the scene that are instances of the supplied class and not its subclasses.
Num-objects ()	Returns the number of objects in the scene.
Instances-of (class objects)	Returns the objects from the supplied objects that are instances of the supplied class. If no objects are supplied, the scene is used.
Direct-instances-of (class objects)	Returns the objects from the supplied objects that are instances of the supplied class but not its subclasses.
New (class geometry attributes)	Returns a new object of the supplied class with the supplied geometry and attribute data.
Class-name (object)	Returns the class of the supplied object.
Name (object)	Returns the name of the supplied object.
ID-number (object)	Returns the ID-number of the supplied object.
Slot-names-and-values (object)	Returns a list of the slot-names and their values for the supplied object.
Is-a (object class)	Returns true if the supplied object is an instance of the supplied class.
Is-strictly-a (object class)	Returns true if the supplied object is an instance of the supplied class and not its subclasses.
Is-not-a (object class)	Returns true if the supplied object is not an instance of the supplied class.
Is-not-strictly-a (object class)	Returns true if the supplied object is not an instance of the supplied class but may be an instance of a subclass.
Get-instance (object-name)	Returns the object that carries the supplied name, if one exists.
Exists (object-name)	Returns true if an object with the supplied name exists.
Any-exist (class)	Returns true if there are any instances of the supplied class (same as instances-of).
Copy (object p1 p2)	Returns a new object of the same class as the supplied object translated across the vector from p1 to p2.
Move (object from to)	Returns a new object of the same class as the supplied object translated across the vector from p1 to p2, and removes the original object.
Make (class object)	Returns an instance of the supplied class that has the same geometry and attributes (where possible) as the supplied object.
Clone (object)	Returns a C-object with the same geometry as the supplied object.
Remove-object (objects)	Removes the supplied objects from the scene.
Remove-all (class)	Removes all instances of the supplied class and its subclasses from the scene.
Remove-just-objects-of (class)	Removes all instances of the supplied class but not its subclasses from the scene.
Are-same-geometric-type (object1 object2)	Returns true if the supplied objects are instances of the same geometric class.
Set-label (label object)	Adds the supplied label to the Labels slot of the supplied object.

Remove-label (label object)	Removes the supplied label from the Labels slot of the supplied object.
Is-labelled (object label)	Returns true if the supplied label exists in the Labels slot of the supplied object.
Is-not-labelled (object label)	Returns true if the supplied label does not exist in the Labels slot of the supplied object.
Has-all-labels (object values)	Returns true if the object has all of the supplied values in its labels slot.
Has-any-labels (object values)	Returns true if the object has any of the supplied values in its labels slot.
All-objects-are-all-labelled (values objects)	Returns t if all supplied objects contain all of the supplied values as a label.
All-objects-are-labelled (value objects)	Returns t if all supplied objects contain the supplied value as a label.
All-objects-labelled (value objects)	Returns the objects of those supplied that have the supplied value as a label.
Number-of-objects-labelled (value objects)	Returns the number of supplied objects that have the supplied value as a label.

Coordinates

These functions return data on the coordinates and dimensions of landscape objects, as well as performing tests on those coordinates.

X (p)	Returns the x or y coordinate
Y (p)	Returns the y coordinate of a point
X-length (objects)	Returns the total length of the objects in the x direction.
Y-length (objects)	Returns the total length of the objects in the y direction.
X-diff (p1 p2)	Returns the difference between the x coordinates of the two supplied points.
Y-diff (p1 p2)	Returns the difference between the y coordinates of the two supplied points.
Xy-extremes (objects)	Returns the minimum and maximum x and y values of the supplied objects.
Minimum-x (objects)	Returns minimum x-coordinate for the supplied objects.
Maximum-x (objects)	Returns minimum y-coordinate for the supplied objects.
Minimum-y (objects)	Returns maximum x-coordinate for the supplied objects.
Maximum-y (objects)	Returns maximum y-coordinate for the supplied objects.
Minimum-x-point (objects)	Returns the points with the minimum x-coordinate for the supplied objects.
Maximum-x-point (objects)	Returns the points with the minimum y-coordinate for the supplied objects.
Minimum-y-point (objects)	Returns the points with the maximum x-coordinate for the supplied objects.
Maximum-y-point (objects)	Returns the points with the maximum y-coordinate for the supplied objects.
Object-length (object)	Returns the maximum length of the supplied object (length for a line, maximum dimension in any direction for a polygon).
Inside-box (object minx miny maxx maxy)	Returns true if the supplied objects is inside a rectangle with the supplied coordinates.

Inside-or-on-box (object minx miny maxx maxy)	Returns true if the supplied objects is inside or on the perimeter of a rectangle with the supplied coordinates.
Geometrically-same (object1 object2)	Returns true if the two objects have the same coordinates.
Geometry-of (object)	If the supplied object is a point, then its coordinates are returned. Otherwise, the vertices of the object are returned.
Coords (object)	If the supplied object is a point, then its coordinates are returned. Otherwise, the coordinates of the vertices of the object are returned.
Extreme-points (points vector min/max)	Returns the points from the supplied points that are minimal or maximal in the direction of the vector.
Furthest-points (p-or-vector points)	Returns the points from the supplied points that are furthest from p. If a vector is supplied instead of p, the returned points are those whose coordinates are most extreme in the direction of the vector.
Closest-points (p-or-vector points)	Same as above, but returns the closest points.
Find-geom-same-vertex (p objects)	Returns any vertex in the supplied objects with the same coordinates as p.
Find-geom-same-points (p points)	Returns the supplied points found that have the identical geometry as the supplied point, p.
Find-geom-same-polylines (line lines)	Returns the supplied lines found that have the identical geometry as the supplied line.

Points

These functions provide tests and information relating to Point objects.

Is-list-of-points (objects)	Returns true if all of the supplied objects are points.
Left-of (p1 p2 p3)	Returns true if p1 is to the left of the line from p2 to p3.
Left-or-on (p1 p2 p3)	Returns true if p1 is to the left of or on the line from p2 to p3.
Right-of (p1 p2 p3)	Returns true if p1 is to the right of the line from p2 to p3.
Right-or-on (p1 p2 p3)	Returns true if p1 is to the right of or on the line from p2 to p3.
On (p line-or-poly)	Returns true if p lies somewhere on the supplied line or perimeter of the supplied poly.
Collinear (p1 p2 p3)	Returns true if the points all lie on a single continuous ray.
Between (p1 p2 p3)	Returns true if all three points are collinear and p3 lies between p1 and p2.
Strictly-between (p1 p2 p3)	Returns true if all three points are collinear, p3 lies between p1 and p2, and does not have the same coordinates as p1 or p2.
Compare-points (p1 p2 p3)	Returns true if $p1 < p2$, where $<$ means smaller angle from the horizontal ray at p3. If the angles are equal, then returns true if p1 is closer than p2 to p3.

Cross-product (p1 p2 p3)	Returns twice the signed area of the triangle determined by p1, p2, p3. The returned value is positive if (p1, p2, p3) are oriented counterclockwise, and p3 is to the left of the line (p1, p2).
Centre-of-MBR (objects)	Returns a C-Point at the centre of the minimum-bounding-rectangle of the supplied objects.
Mean-centre-point (points)	Returns a C-Point located at the mean location of the supplied points.
Regression-line (points)	Returns the least-squares regression line through the points.
Standard-distance-deviation (points)	Returns a numeric measure of spatial dispersion. The standard deviation of the distance of each point to the mean-centre-point.
Sum-of-variances (points)	Returns the sum of squared distances of the supplied points from the mean-centre-point.

Lines

These functions provide tests and information relating to Polyline objects. In some cases, new Polyline objects are returned.

Object-length (object)	If object is a line, the line's length is returned. If object is a polygon, the polygon's maximum length in any direction is returned.
Midpoint (line)	Returns a C-Point at the midpoint of the supplied line.
Endpoints (line)	Returns the two endpoints/nodes of the supplied line.
Is-list-of-lines (objects)	Returns true if all of the supplied objects are lines.
Calc-perpendicular (length p1 vector p2 p3)	Returns a C-Point which is at the end of a line of the supplied length, starting at the supplied point, in the supplied direction, and perpendicular to a line between the two supplied vertices.
Extend (line p distance)	Returns a C-Polyline that has the same geometry as the supplied line except that the line-segment at the supplied endpoint has been extended by the supplied distance.
Line-extension-point (line p distance)	Returns a C-Point whose location is at the end of an extension to the supplied line as described for the previous function.
Extend-to (line p objects)	Returns a C-Polyline that has the same geometry as the supplied line except that the line-segment at the supplied endpoint has been extended to the supplied objects.
Line-extension-point-to (line p objects)	Returns a C-Point whose location is at the end of an extension to the supplied line as described for the previous function.
Insert-points-along-line (distance line)	Returns a list of C-Points which are spaced at the supplied interval distance along the supplied line.
Reverse-line (line)	Reverses the order of nodes and vertices in the supplied line.
Make-perpendicular (length p direction line)	Returns a C-Polyline of the supplied length, starting from the supplied point, in the supplied direction and perpendicular to the supplied line.

Offset (object origin distance direction)	Returns a C-Object with the coordinates of the supplied object offset from the origin over the supplied distance in the supplied direction. The direction can be implied from a vector, point or left/right.
Offset-vertex (p line distance direction)	Returns a C-Point that is the result of offsetting the supplied vertex (p) away from the supplied line.
Parallel (line1 line2)	Returns true if two lines are parallel.
Slope (line)	Returns the slope of the supplied line, vector, or list of points.
Slope-between-points (p1 p2)	Returns the slope of the line between the two supplied points.
Path-of-vertices (lines p)	Returns a list of points that is a path starting at the supplied point through the set of intersecting lines.
Point-on-line (p line)	Returns true if the supplied point is within the snapping distance of the line.
Point-on-line-between-points (p1 p2 p3)	Returns true if p1 is within the snapping distance of a line between p2 and p3.
Split-line (line p)	Splits the supplied line at the supplied point and returns two lines.
Split-line-at-vertex (line p)	Splits the line at the supplied vertex.
Split-line-between-vertices (line p)	Splits the line at a point that lies between vertices.
Split-lines-with-points (lines points)	Returns the set of new lines after splitting them at every one of the supplied points.
Sine-curve (p1 p2 wavelength amplitude offset)	Returns a C-Polyline that curves around the line from p1 to p2 with the supplied wavelength and amplitude, starting at the supplied offset distance from p1.
Transect-line (p distance direction)	Returns a C-Polyline created from a vector from the supplied point over the supplied distance in the supplied direction (a vector, bearing, or point).
Unsplit-lines (line1 line2)	Returns a C-Polyline that is the concatenation of line1 and line2.

Vertices

These functions return particular points (Vertices) along a Polyline, Polygon or list of points and can be used to manipulate the geometries of these objects.

From-node (line)	Returns the first node of the supplied line.
To-node (line)	Returns the last node of the supplied line.
Endpoint (line object away-or-near)	Returns the point of the supplied line that is or is not shared by the supplied object. (line may be a list of points)
Other-endpoint (node line)	Returns the point at the other end of the supplied line from the supplied point.
Endpoints (line)	Returns the points at either end of the line.
Vertices (object)	Returns the vertices of any geometric-object.
Vertices-closed (polygon)	Returns the vertices of the polygon concatenating the first vertex to the end of the list.
Number-of-vertices (object)	Returns the number of vertices in the supplied object.
Contains-vertex (p object)	Returns true if the supplied point is in the vertices slot of the supplied object.

First-vertex (line)	Returns the first vertex of the supplied line. <i>Similar functions exist for second-vertex, third-vertex, last-vertex, second-last-vertex.</i>
Next-vertex (p line)	Returns the vertex after p in the supplied line's vertices slot.
Next-vertex-circular (p line)	Same as above, but returns the first vertex if p is the last vertex.
Previous-vertex (p line)	Returns the vertex before p in the supplied line's vertices slot.
Previous-vertex-circular (p line)	Same as above, but returns the last vertex if p is the first vertex.
Points-after (p object)	Returns the points in the supplied object that occur after p in its vertices.
Points-before (p object)	Returns the points in the supplied object that occur before p in its vertices
Between-vertices (p line)	If the supplied point falls somewhere on the supplied line, the two Vertices between which the point falls are returned.
Add-point-to-line (p line)	Makes the supplied point the last vertex of the supplied line.
Insert-vertex (p line n)	Inserts a point at the nth position of the supplied line's vertex list.
Remove-vertex (p object)	Removes the supplied vertex from the supplied object.
Replace-vertex-in-line (p1 p2 line)	Replaces p1 with p2 if p1 is a vertex in the supplied line.
Replace-vertex-in-all-lines (p1 p2)	Replaces p1 with p2 in all line objects in the scene.
Lines-with-point (p lines)	Returns those of the supplied lines which contain p in their vertices slot.
A-line-with-point (p lines)	Returns the first of the supplied lines which contains p.
Find-point-on-line (line test starting-from increment-by)	Searches the line from the supplied endpoint and at the supplied incremental distance for the first point that satisfies the supplied test, which takes a point and returns a true/false value.
Find-segments-on-line (line test starting-from increment-by)	Similar to above but returns a list of points defining the points along the line at which the supplied test alternates between being true and false.
Nearest-vertex-to (line objects)	Returns the vertex or node of the supplied line that is nearest to the supplied objects.
Is-a-closed-loop (lines start-node)	Returns a list of points that define the closed loop through the polylines, if they are a closed loop joined by their nodes.
Follow-loop (line start-node direction lines)	Returns the lines in an order that loops back to the start-node.

Polygons

These functions provide tests and information relating to Polygon objects. In some cases, new Point or Polygon objects are returned.

Area (polygon)	Returns the area of the supplied polygon. Does not work for self-intersecting polygons.
Signed-polygon-area (polygon)	Returns the signed area of the supplied polygon.

Signed-parallelogram-area (p1 p2 p3)	Returns the signed area of the four-sided parallelogram that has the three supplied points as vertices.
Signed-triangle-area (p1 p2 p3)	Returns the signed area of the triangle formed by the three supplied points.
Perimeter (polygon)	Returns the length of the perimeter of the supplied polygon.
Centroid (polygon)	Returns a C-Point which is the centroid of the supplied polygon.
Convex-hull (points)	Returns a C-Polygon which is the convex hull for the supplied list of points.
Minimum-bounding-rectangle (objects)	Returns a C-Polygon that is the minimum-area rectangle that bounds all of the supplied objects.
Minimal-rectangle (objects)	Same as above but the rectangle is not necessarily aligned with the x and y axes.
Min-rectangle-with-edge (objects p1 p2)	Returns the minimum-bounding-rectangle that has the two supplied points as an edge.
Diagonal-internal-external (p1 p2 polygon)	Returns true if the line between the supplied points does not cross the perimeter of the supplied polygon.
Diagonal-in-cone (p1 p2 polygon)	Returns true if the diagonal from point p1 to point p2 is inside the supplied polygon in the locality of p1.
Diagonal (p1 p2 polygon)	Returns true if a line between the supplied pair of points is a true internal diagonal of the supplied polygon.
Inside (object polygon)	Returns true if a supplied point, line or polygon is inside a supplied polygon.
Inside-or-on (object polygon)	Returns true if a supplied object is inside a supplied polygon or on its boundary.
Outside (object polygon)	Returns true if a supplied point, line or polygon is outside a supplied polygon.
Outside-or-on (object polygon)	Returns true if a supplied object is outside a supplied polygon or on its boundary.
Make-polygon (objects)	Returns a C-Polygon from a list of points or lines.
Polygons-containing-line (line)	Returns a list of C-Polygons that contain the supplied line in their perimeters.
Number-of-polygons-containing-line (line)	Returns the number of C-Polygons that contain the supplied line in their perimeters.
Lines-in-polygon-perimeter (lines polygon)	Returns true if the supplied lines comprise part of the perimeter of the supplied polygon.
Line-coincides-partially-with-polygon (line polygon)	Returns true if only some of the vertices of the supplied line fall on the polygon's perimeter.
Line-coincides-wholly-with-polygon (line polygon)	Returns true if all of the vertices of the supplied line fall on the polygon's perimeter.
Find-lines-in-polygon (polygon)	Returns a list of lines that form the polygon's perimeter.
New-rectangle (width height p)	Returns a rectangular C-Polygon with the supplied width and height and the supplied point as its centre.
Polygon-orientation (polygon)	Returns :clockwise or :counterclockwise according to the direction of the perimeter of the supplied polygon.
Clip (line polygon)	Returns the parts of the line that are inside the supplied polygon.

Scale (polygon factor p)	Returns a C-Polygon that is the supplied polygon scaled by the supplied factor around the supplied point.
Scale-to-fit-box (polygon rectangle p)	Returns a C-Polygon that is the supplied polygon scaled around the supplied point to fit the supplied rectangle.
Triangulate (polygon)	Returns a list of C-Polylines that are the diagonals of the supplied polygons.

Topology

The topological functions provide adjacency tests between objects. This include line-node topology as well as line-polygon topology. Some of the functions return true/false while other return objects that are connected to the supplied object(s).

Are-adjacent (object1 object2)	Returns true if two objects are adjacent to each other, that is, they are joined by a node or an edge.
Is-between (line1 line2 line3)	Returns true if the endpoints of line1 are also endpoints of both line2 and line3.
Adjacent-nodes (node lines)	Returns a list of nodes that are at the other end of the lines that are attached to the supplied node.
Is-connected-to (line1 line2)	If line1 and line2 share a node, then that node is returned.
Is-connected-to-a (line class)	Returns true if the supplied line is connected to a line of the supplied class.
Is-not-connected-to-a (line class)	Returns true if the supplied line is not connected to a line of the supplied class.
Lines-connected-to (line lines class)	Returns those of the supplied lines that share nodes with the supplied line. If class is supplied, only lines of that class are returned.
A-line-with-node (node lines)	Returns one line from the supplied lines that contains the supplied node.
A-line-between-nodes (node1 node2)	Returns the line that contains both of the supplied nodes.
All-lines-with-node (node lines)	Returns all lines from the supplied lines that contain the supplied node.
A-line-connected-to (line lines)	Returns one line from the supplied lines that shares a node with the supplied line.
Is-a-dangle (line lines)	Returns true if the supplied line is not connected to other lines at both ends. If lines are supplied the test is restricted to those lines only.
Is-bounded-by (line class)	Returns true if the supplied line shares both nodes with lines of the supplied class.
Is-circular (line)	Returns true if the supplied line connects to itself at its endpoints.
Have-common-edge (object1 object2)	Returns true if the two supplied objects share a common edge, that is, they are joined by an edge.

Intersections

The intersection functions test the spatial coincidence of Points and Polylines to each other. Some functions return a true/false result while others return an intersection point or return the supplied lines split at their intersection points.

Collinear (p1 p2 p3)	Returns true if the points all lie on a single continuous ray.
Between (p1 p2 p3)	Returns true if the points are collinear and p1 is on the line between p2 and p3.
Intersects (object1 object2)	Returns true if two lines or polygons intersect.
Segments-Intersect (p1 p2 p3 p4)	Returns true if the line segments between each pair of points intersect or one of the points is between two other points.
Segments-intersect-properly (p1 p2 p3 p4)	Returns true if an intersection point is interior to both segments and no point is between any others.
Intersection-point (p1 p2 p3 p4)	Returns a C-Point at the intersection of the lines from p1 to p2 and p3 to p4.
Line-intersection (line1 line2)	Returns a C-Point that is the intersection point of the two lines.
All-line-intersections (line1 line2)	Returns a list of all of the intersection points for the two lines.
Intersect-lines (line1 line2)	If the two lines intersect, they are split and the resultant lines are returned.
Intersect-line-with (line lines)	Splits the supplied lines at their intersections with the supplied line, and returns the new lines.
Intersect-all-lines (lines)	Intersects all of the supplied lines with each other, splitting lines and inserting nodes, and returns the list of new lines.
Intersects-ray (p1 p2 line)	Returns true if the supplied line intersects the infinite ray defined by p1-p2.
Ray-intersection (p1 p2 p3 p4)	Returns a C-Point that is at the intersection of two infinite rays defined by p1-p2 and p3-p4.
First-ray-intersection-with-line (p1 p2 line)	Returns the first intersection point of the supplied line with the infinite ray defined by p1-p2.
All-ray-intersections (p1 p2 line)	Returns a list of C-Points that define the intersections of the supplied line (or polygon) with the infinite ray defined by p1-p2.
Directed-ray-intersections (p1 p2 line)	Returns the same as the above function but only those points that are ahead of p2 on the ray.
Nearest-directed-ray-intersection (p1 p2 objects p3)	Of the ray intersection points for the supplied objects, the point that is ahead of p2 and nearest to p3 is returned.
Ip-ahead-test (p1 p2)	Returns a function that can be used to determine if any point is on the infinite ray defined by p1-p2 and is ahead of p2.

Buffering

These functions return a Polyline or Polygon object that is a constant distance from the supplied object.

Buffer (p distance)	Returns a C-Polygon that is a circle around the supplied point with a radius of the supplied distance.
Buffer (polygon distance direction)	Returns a C-Polygon whose perimeter is the supplied distance from that of the supplied polygon.

Buffer (line distance direction buffer-shape)	Returns a C-Polyline that is the supplied distance from that of the supplied line. The ends of the buffer can be square or round as designated by buffer-shape. The direction is either left or right.
Buffer-completely (line distance buffer-shape)	Same as previous function but the line is completely encircled by the buffer, and can be used as a C-Polygon.

Angles

This set of functions measures angles between Points and vectors in various ways.

Angle (vector)	Returns the angle between the horizontal and vector. The number is negative if the vector is below the x axis.
Cc-angle (vector)	Returns the counterclockwise angle between the plus-x axis and the vector. The angle is always \geq zero and $< 2\pi$.
Angle-between-vectors (vector1 vector2)	Returns the angle between the two vectors.
Angle-between-points (p1 p2 p3)	Returns the smaller angle between a ray extending from p1 through p2, and a line from p2 to p3.
Signed-angle-between-points (p1 p2 p3)	Returns a positive angle for counter-clockwise, negative for clockwise.
Cumulative-angle-on-path (points)	Returns the sum of the angles between the line-segments between the supplied points.
Cumulative-signed-angle-on-path (points)	Returns the sum of the signed angles between the line-segments between the supplied points.
Deg2rad (degrees)	Converts degrees to radians.
Rad2deg (radians)	Converts radians to degrees.
Point-to-polar	Returns the polar coordinates of the supplied point.
Polar-to-point	Returns a C-Point located at the supplied polar coordinates.
Rotate (object p angle)	Returns a C-Object which is a copy of the supplied object rotated through the supplied angle around the supplied point.
Opposite-direction (direction)	Returns the reverse of the supplied direction which may be a bearing, vector, or one of north, south, east, west, left or right.
Direction-towards (p1 p2 p)	Returns the direction from the line defined by p1-p2 to p as right, left or on.
Direction-towards-line (line p)	Returns the direction from the supplied line to p as right, left or on.
Direction-away-from (p1 p2 p)	Returns the opposite direction of direction-towards.
Direction-away-from-line (line p)	Returns the opposite direction of direction-towards-line.
Leftmost (line node lines)	If the supplied line is connected to the supplied lines by the node, then returns the line that represents the leftmost (counterclockwise) turn away from the source line at the node.
Rightmost (line node lines)	If the supplied line is connected to the supplied lines by the node, then returns the line that represents the rightmost (clockwise) turn away from the source line at the node.

Distance

These functions provide the measurement of distances between objects as well as finding or manipulating objects based on their proximity to other objects.

Distance (object1 object2)	Returns the minimum distance between the supplied objects.
Distance-along-line (distance line)	Returns a C-Point which is the supplied distance along the supplied line.
Distance-along-line-between-points (distance points)	Returns a C-Point which is the supplied distance along a line between the supplied points.
Distance-to-ray (p1 p2 p3)	Returns the perpendicular distance between p1 and the continuous ray through p2 and p3.
Is-within-distance (object1 object2 distance)	Returns true if the distance from object1 to object2 is less than or equal to the supplied distance.
Closest-points (p points)	Returns the point(s) in the supplied points that is closest to p. p can also be a vector.
Furthest-points (p points)	Returns the point(s) in the supplied points that is furthest from p. p can also be a vector.
Nearest-objects-to (object class-or-objects)	Returns a list of the nearest objects to the supplied object. The search may be limited to a class or list of objects.
Nearest-node-to (line objects)	Returns the node of the supplied line that is closest to the supplied objects.
Nearest-line-segment (p object)	Returns a list of the two vertices in the supplied line or polygon which comprise the line segment nearest to the supplied point.
Nearest-point-on-line (p object)	Returns a C-Point that is the nearest point on the line (perimeter if object is a polygon) to the supplied point.
Nearest-point-on-line-to (line objs-or-class)	Returns a C-Point that is the nearest point on the line to the supplied objects or objects of the supplied class.
Nearest-point-on-line-pts (p p1 p2)	Returns a C-Point that is the nearest point to p on the line defined by p1-p2.
Nearest-point-on-ray (p1 p2 p3)	Returns a C-Point that is the nearest point to p3 on the continuous ray through p1 and p2.
Perpendicular-distance (p object)	Returns the perpendicular distance between the point and object.
Perpendicular-distance-pts (p1 p2 p3)	Returns the perpendicular distance between p1 and the line between p2 and p3.
Point-at-fractional-distance (p1 p2 distance)	Returns a C-Point at the fractional distance along the line between p1 and p2, where p1 is at 0 and p2 is at 1.
Point-at-real-distance (p1 p2 distance)	Returns a C-Point at the distance along the line between p1 and p2.
Point-on-circle (centre-x centre-y radius-x radius-y bearing)	Returns a C-Point that sits at the supplied bearing on the circumference of the circle at (centre-x,centre-y) with the supplied radius. Radius-x and radius-y are the same for a circle and different for an ellipse.
Is-snappable (p object)	Returns true if p is within the snap-tolerance distance of the supplied object.
Snap (p object)	If the supplied point is within the snap-tolerance distance of the supplied object, then the coordinates of p are changed so that it touches object. The point is returned.

Remove-snappable-points (points)	Returns a copy of the supplied list of points after removing any consecutive vertices that are snappable.
----------------------------------	---

Grids

The grid functions return various results related to grids, grid cells and their values. They allow for the incorporation of raster spatial data within LGS.

Grid-centre (grid)	Returns the centre cell of grid.
Grid-height (grid)	Returns the height (y-length) of the grid in map units.
Grid-width (grid)	Returns the width (x-length) of the grid in map units.
Make-grid (num-cols num-rows cell-size origin layers display-layer)	Makes an instance of the Grid class with the supplied parameters.
Make-grid-layer (layername cells no-data-value)	Makes a layer in the grid with the supplied name and cell values.
Set-display-layer (layername)	Specifies which layer is to be displayed in the Scene View window.
Row (cell)	Returns the row number of the cell.
Col (cell)	Returns the column number of the cell.
Cells-eq (cell1 cell2)	Returns true if cell1 and cell2 have the same row and column numbers.
Cell-to-LL (cell)	Returns the cell to the lower-left of the supplied cell. Similar functions exist for lower-middle, lower-right, middle-left, middle-right, upper-left, upper-middle, and upper-right.
Cell-centre (cell)	Returns the xy coordinates of the centre of the supplied cell.
Cell-to-point (cell)	Returns a C-Point at the centre of the supplied cell.
Cell-LL-corner (cell)	Returns a C-Point located at the lower-left corner of the supplied cell. Similar functions exist for lower-right, upper-left and upper-right.
Mapx-to-col (x)	Converts an x-coordinate to a grid column number.
Mapy-to-row (y)	Converts a y-coordinate to a grid row number.
Distance-to-cells (distance)	Returns the number of cells covered by the supplied distance along a grid axis.
Cell-value (cell layer)	Returns the value of the cell for the given attribute layer.
Is-no-data-cell (cell layer)	Returns true if the supplied cell contains a "no-data" value for the supplied layer.
Find-unique-values (grid layer)	Returns an unordered list of the unique cell-values for a layer in the grid.
Set-cell-value (cell value layer)	Sets the value of the cell for the supplied layer to the supplied value.
Cell-at-point (p)	Returns the cell with the same location as the supplied point.
Find-cells-on-line (line)	Returns the list of cells through which the supplied line passes.
Find-cells-in-polygon (polygon)	Returns the list of cells whose centre is inside the supplied polygon.

Cell-value-at-point (p layer)	Returns the value of the cell that coincides with the supplied point for the supplied attribute layer.
Cell-values-on-line (line layer)	Returns a list of the values of the cells that coincide with the supplied line for the supplied layer.
Cell-values-in-polygon (polygon layer)	Returns a list of the values of the cells that coincide with the supplied polygon for the supplied layer.
Min-cell-value-on-line (line layer)	Returns the minimum value from the supplied layer of the cells that fall in the supplied line.
Max-cell-value-in-line (line layer)	Returns the maximum value from the supplied layer of the cells that fall in the supplied line.
Mean-cell-value-in-line (line layer)	Returns the mean value from the supplied layer of the cells that fall in the supplied line.
Sum-cell-values-in-line (line layer)	Returns the sum of the values from the supplied layer of the cells that fall in the supplied line.
Min-cell-value-in-polygon (polygon layer)	Returns the minimum value from the supplied layer of the cells that fall in the supplied polygon.
Max-cell-value-in-polygon (polygon layer)	Returns the maximum value from the supplied layer of the cells that fall in the supplied polygon.
Mean-cell-value-in-polygon (polygon layer)	Returns the mean value from the supplied layer of the cells that fall in the supplied polygon.
Sum-cell-values-in-polygon (polygon layer)	Returns the sum of the values from the supplied layer of the cells that fall in the supplied polygon.
Neighbourhood (cell nhood-type)	Returns the cells that are in the neighbourhood of the supplied cell, according to the neighbourhood-type.
Find-adjacent-cells (cells nhood-type)	Returns the cells that are adjacent to the supplied cells, according to the neighbourhood-type.
Find-cells-in-nhood-that-meet-criteria (cell criteria-function nhood-type)	Returns the cells in the neighbourhood of the supplied cell that satisfy the supplied criteria-function.
Define-zone (cell criteria-function)	Returns a list of cells that are found by starting at the supplied cell and finding all the adjacent cells that satisfy the criteria function.
Num-cells-in-nhood-with-value (test-function value cell layer)	Returns the number of cells in the neighbourhood of the supplied cell that contain the value in the supplied layer. The test-function can be applied to the cell-values before the comparison is made.

Randomization

The randomization functions incorporate stochasticity into the LGS implementation. They are provided mainly for use in rules rather than use inside the LGS system itself.

a-number-between (num1 num2)	Returns a random number greater than num1 and less than num2.
------------------------------	---

a-number-< (num)	Returns a random number less than the supplied number. There are similar functions for a-number->, a-number-<=, and a-number->=.
One-of (objects probabilities)	Returns a random object from the supplied list of objects. Also works with lists of numbers or strings. If a list of probabilities is supplied, the random selection is influenced by them.
Random-element (objects)	Same as one-of.
Random-n-elements (num objects)	Returns a list of randomly selected objects with a length of the supplied number.
With-probability (probability functions)	The supplied functions are executed only if a randomly selected number satisfies the supplied probability.

Vectors

In some situations, geometrical calculations are more easily performed using vector geometry than with coordinate geometry. The vector functions provide an array of utilities to define, query, manipulate vectors.

Add-vectors (vectors)	Returns a vector that is the sum of the supplied vectors.
Add-3D-vectors (vectors)	Same as above but with a z component.
Add-vector-to-point (vector p distance)	Returns a vector that is the result of adding the supplied vector to the position-vector of p, and scaling the vector to the supplied distance.
Subtract-vectors (vector vectors)	Returns the result of subtracting the supplied vectors from the first supplied vector.
Subtract-3D-vectors (vector vectors)	Same as above but with a z component.
Vector-product (scalar vector)	Returns a new vector by multiplying the components of the supplied vector by the scalar value.
Vector-product-3D (scalar vector)	Same as above but with a z component.
Vector-quotient (scalar vector)	Returns a new vector by dividing the vector components by the scalar value.
Vector-eq (vector1 vector2)	Returns true if all components of both vectors are equal.
Vector-length (vector)	Returns the numeric length of the vector.
Vector-3D-length (vector)	Returns the 3D length of the vector.
Extend-vector-to-length (vector distance)	Returns a new vector in the same direction as the supplied vector, but of the supplied length.
Extend-3D-vector-to-length (vector distance)	Same as above but extended in 3D.
Dot-product (vector1 vector2)	Returns the dot-product (a number) of the supplied vectors. Points may be supplied instead of vectors.
Dot-product-3D	Same as above but with a z component.
Vector-cross-product (vector1 vector2)	Returns the cross-product (a vector) of the supplied vectors.
Are-perpendicular (vector1 vector2)	Returns t if the two supplied vectors are perpendicular to each other.
Are-parallel (vector1 vector2)	Returns t if the two supplied vectors are parallel to each other.

Perpendicular-vector (vector direction)	Returns the perpendicular-vector of vector either to the :right or the :left.
Perpendicular-component-of-vector (vector)	Returns the vector which is the orthogonal component of the supplied vector, perpendicular to the unit-vector.
Parallel-component-of-vector (vector)	Returns the vector which is the orthogonal component of the supplied vector, parallel to the unit-vector.
Position-vector (p)	Returns the position-vector of the supplied point (a vector from the origin to p).
Reflect-vector (vector1 vector2)	Returns vector1 reflected in the axis defined by vector2.
Reverse-vector (vector)	Returns the supplied vector in the opposite direction.
Rotate-vector (vector angle)	Returns the supplied vector rotated through the supplied angle.
Unit-vector (vector)	Returns the supplied vector scaled to a length of one unit.
Unit-3D-vector (vector)	Same as above but with a z component.
Make-vector (p1 p2)	Returns a 2D vector from p1 to p2.
Left-unit-vector (p1 p2)	Returns the unit-vector perpendicular and to the left of a line from p1 to p2.
Right-unit-vector (p1 p2)	Returns the unit-vector perpendicular and to the right of a line from p1 to p2.
Vector-to-point (vector p distance)	Returns a C-Point whose coordinates are taken from the x and y components of the vector (a position-vector). If p is supplied, the vector is added to p. If distance is supplied, the vector is scaled to be that length.
Move-by-vector (object vector distance)	Moves the object by the supplied vector scaled to the supplied distance.
Bearing (p1 p2)	Returns the compass bearing from p1 to p2, assuming north is in the direction of the y axis.
Bearing-to-vector (bearing)	Returns a unit-vector in the direction of the supplied compass bearing.

Rule-base functions

The LGS implementation allows the definition of grammar rules and rulesets together comprising a rule-base for use in the grammar interpreter. These functions define and query rules and rulesets in LGS.

Define-rule (rule-name rulesets if then explanation)	Creates a new rule in the supplied rulesets with the supplied name, antecedent (if), consequent (then), and explanation string.
Define-ruleset (ruleset-name rules)	Creates a new ruleset with the supplied name and rules.
Define-parameter (parameter-name value)	Creates a new parameter with the supplied name and value.
Rules-of (ruleset)	Returns a list of the rules in the supplied ruleset.
All-rules ()	Returns a list of all rules.
All-rulesets ()	Returns a list of all rulesets.
Get-rule (rule-name)	Returns the rule with the supplied name.
Get-ruleset (ruleset-name)	Returns the ruleset with the supplied name.
Add-to-ruleset (rule ruleset)	Registers a rule with a ruleset.

Antecedent (rule)	Returns the antecedent of the supplied rule.
Consequent (rule)	Returns the consequent of the supplied rule.
Number-of-different-variables (rule)	Returns a count of the different variable-names used in the rule's antecedent.
Sort-antecedent (antecedent)	Returns the supplied antecedent with its predicates sorted into an order efficient for processing.
Modify-antecedent (antecedent)	Returns the antecedent in an optimal form for processing by regrouping the antecedent's parts.
Sort-var-lists (var-list)	Var-list is a list of lists such as '((?a ?b ?c) (?a ?b) (?a) (?b))'. Sorts them by length and then alphabetically.
Contains-interpreter-call (rule)	Returns true if the supplied rule contains a call to the interpreter function in its consequent.
Contains-assignment (expression)	Returns the name of the variable that is the subject of an assignment within the supplied expression.
Return-assignment-expr (expression)	Returns the expression that is an assignment within the supplied expression.
Return-all-assignment-exprs (expression)	Returns a list of all assignment expressions within the supplied expression.
Remove-rule (rule)	Removes the supplied rule.
Remove-ruleset (ruleset)	Removes the supplied ruleset.
Remove-from-ruleset (rule ruleset)	Removes the supplied rule from the supplied ruleset.
Clear-ruleset (ruleset)	Removes all rules from the supplied ruleset.

Interpreter functions

The LGS interpreter functions perform the processing of scenes of objects and grammar rules. This process is represented in Figure 3.4. There are functions to compare the antecedents of rules to the working-scene of objects, to select rules from those whose antecedents match the scene, to select objects upon which the selected rule is fired, and to finally fire the consequents of the selected rule(s). Other functions in this set record the rule-firing activity and test for the achievement of pre-defined goals or stopping conditions.

Start (ruleset goal scene)	Begins a new interpretation process with the supplied ruleset, goal function, and scene objects.
Initialize-rules (rules)	Clears the environment slots of the supplied rules.
Interpret (rules objects goal)	The central controlling function for the interpretation process.
Find-rules-to-fire (rules objects)	Returns the rules whose antecedents match objects from the scene.
Match-rule (rule objects)	Returns the rule if its antecedent matches objects from the scene.
Match-antecedent (antecedent objects environment rule)	Returns the sets of matching objects from the scene for which the supplied rule's antecedent is true.
Is-a-variable (symbol)	Returns t if the supplied symbol represents a rule-interpreter variable, that is, its first character is '?'.
Variable-list (antecedent)	Returns a list of the variable-names used in the supplied antecedent.

Find-matches-for-0-vars (predicate)	Returns true if the predicate is true.
Find-matches-for-1-var (predicate objects)	Returns a list of the variable-name and matching objects from the scene.
Find-matches-for-2+-vars (predicate objects environment rule)	Returns a list of lists of matching objects corresponding to the variables stored for the rule.
Predicate-to-lambda (predicate rule)	Returns a lisp function that has the variables of the rule as its arguments and the predicate as its main body.
Find-all-combos-if (test variables matching-objects)	Returns a list of object combinations, each of which is a list of objects for which the supplied test is true.
Find-all-if (test sequence)	Returns all the items in the sequence which return true when used as arguments to the supplied test.
Assign (varname value)	Creates a variable with the supplied name and assigns the supplied value to it.
Fail (rule)	Returns true if the rule produced no matching environments.
Select-rules (matching-rules)	Returns the list of supplied rules after applying selection strategies to reduce the number of rules in the list.
Select-rule (rules strategies)	Applies selection strategies to the supplied rules until the desired number of rules remains or all strategies are exhausted.
First-rule (rules)	Returns the first rule from the supplied rules.
Random-rule (rules)	Returns a random rule from the supplied rules.
Least-recently-fired-rules (rules)	Returns the rules from the supplied rules that have been fired least recently.
Rules-with-instantiations- not-fired (rules)	Returns the rule from the supplied rules that have matching environments to which the rule has not yet been applied.
Select-environment (rule strategies)	Applies selection strategies to the supplied object environments until the desired number of environments remains or all strategies are exhausted.
First-environment (rule)	Returns the first object environment from the supplied rule.
Random-environment (rule)	Returns a random environment from the supplied rule.
Environments-not-previously- fired (rule)	Returns the environments to which the supplied rule has not yet been applied.
Instantiated-consequents (rule)	Returns a list the elements of which contain the rule's consequent with its variables instantiated with objects from each of the rule's object environments.
Fire-rules (rules)	Fires each of the supplied rules and redraws the scene.
Fire-rule (rule)	Instantiates the consequent of the supplied rule and executes the consequent.
Instantiate (consequent variables environment)	Returns the consequent with its variables substituted with the objects that are bound to those variables in the environment.
Is-bound (variable environment)	Returns true if the variable is bound to an object in the environment.
Value-of (item environment)	If the item is a variable that is bound to an object in the environment, the object is returned. Otherwise the item (object) itself is returned.

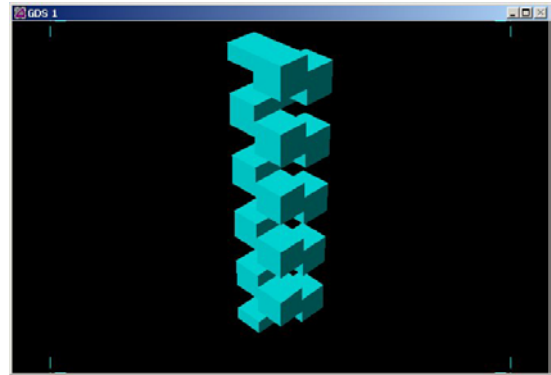
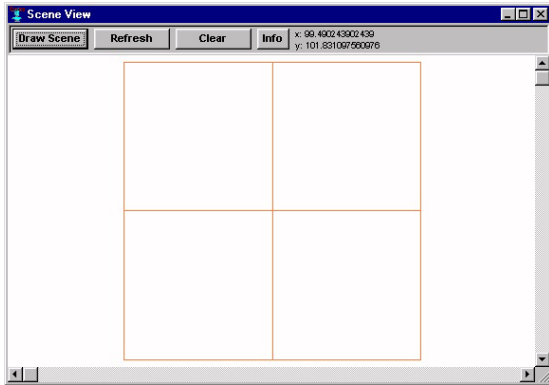
Eval-instantiation (instantiation)	Executes each expression in the supplied instantiated consequent.
Add-to-previously-fired-envts (environment rule)	Adds the environment to the previously-fired-environments slot of the rule after it has been fired.
Add-rule-to-history (rule)	Adds the rule to the rule-history-file.
Note-cycle-last-fired (rule cycle)	Adds the iteration number to the cycle-last-fired slot of the supplied rule.
Reset-rule-slots (rules)	Resets the possible-environments slot of the supplied rules to nil.
Goal-achieved (goal)	Returns true if the supplied goal expression returns true.
Halt-signalled ()	Returns true if a halt has been issued to the interpretation process.
Signal-halt ()	Adds a halt to the interpreter's messages.
working-scene ()	Returns a list of all objects in the working-scene.
Make-new-scene-table (objects)	Creates a new scene table from the supplied objects at the start of a nested interpretation process.
Install-in-scene (object table)	Adds the supplied object to the supplied scene table.
Adjust-scene-after-nested-loop ()	After a nested interpretation process has been completed for a subset of objects and rules, the objects returned from the nested process are added to the main scene.
Merge-scenes (scene1 scene2)	Merges the two supplied scenes (tables).
Add-nested-loop-to-history (rules)	Adds a note pertaining to the nested loop in the rule-history-file.
Rule-has-fired (rule)	Returns true if the supplied rule has been fired in the current session.
Rule-has-not-fired (rule)	Returns true if the supplied rule has not been fired in the current session.
Rules-have-fired (rules)	Returns true if all of the supplied rules have been fired in this session.
Rules-have-not-fired (rules)	Returns true if not all of the supplied rules have been fired in this session.
Rule-has-fired-with-objects (rule objects)	Returns true if the supplied objects have been supplied to rule and fired as an environment.
Rule-has-not-fired-with-objects (rule objects)	Returns true if the supplied objects have not been supplied to rule and fired as an environment.
Watch-rule (rule)	Signals that data from the interpreter will be printed for the supplied rule during processing.
Unwatch-rule (rule)	Halts watching of the supplied rule.

Appendix B

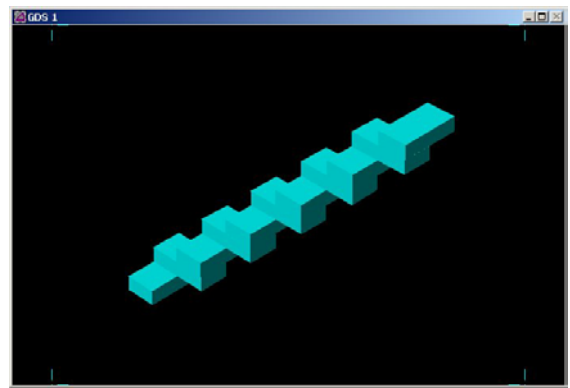
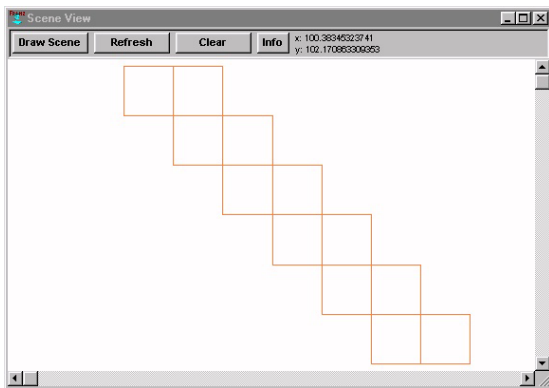
Application of the LGS System to Other Spatial Grammars

While the LGS system implementation differs from some other spatial grammars in its focus on geographic features and its incorporation of topology, grid structures, and expressive Lisp-derived language in its rules, it is still derived from the features of those spatial grammars discussed in Chapter 2 by way of its basis in the landscape grammar theory of Chapter 3. Consequently, the LGS system is able to emulate the functionality of other spatial grammars, as shown by the Scene-View displays in the following figures. In each set of screen captures, the LGS Scene-View output is shown on the left and 3D output rendered in GDS is shown on the right. Each 2D block in the landscape grammar application has elevation and height attribute data which are used to create the 3D blocks in GDS. Figure B.8.1 (i) and (ii) show a tower and stair configuration from Stiny (1980b). Figure B.8.1 (iii), a spiral staircase, is a variation of the rules that created Figure B.8.1 (ii).

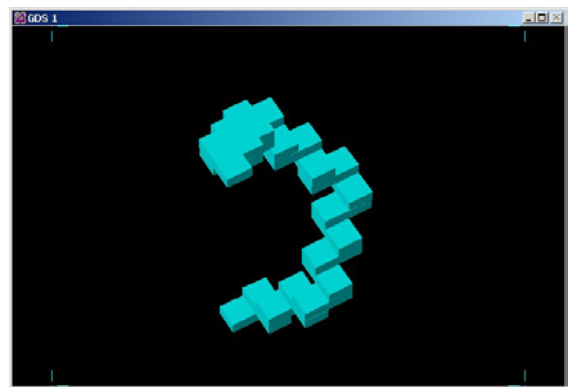
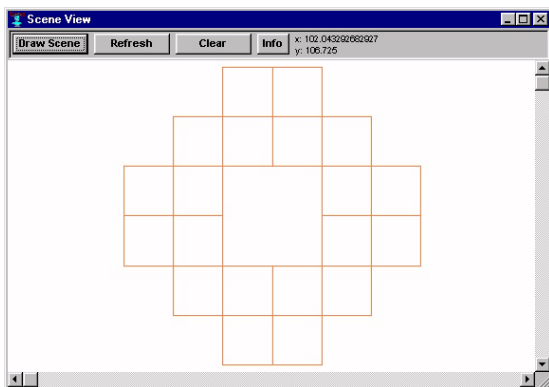
Figure B.8.2 (i) shows the application of the interpreter to the random generation of objects. The illustration on the left was generated by the iteration of a rule that inserts a random point within an area. The scene on the right is the output of rules that emulate a ‘random walk’ path-tracing program. The interpreter was switched to parallel rule-firing mode (all rules and all objects) for the designs in Figure B.8.2 (ii) and (iii). The scene on the left of Figure B.8.2 (ii) shows an emulation of an L-system, although not using the string-rewriting features of a typical L-system. The scene on the right of Figure B.8.2 (ii) demonstrates the ability of the system to generate fractal designs to a user-specified level of detail. This example is of the Koch curve. Finally, Figure B.8.2 (iii) shows the result of rules operating on a grid of cells creating spatial patterns and emulating cellular automata processes. The scene on the left is a typical pattern generated by Game of Life scenarios, while that on the right illustrates the use of cellular automata in simulating urban growth. The black area is vacant land, the grey area urbanized land and the white area water. The rules were designed to state that urban sprawl is likely to occur more rapidly near the water.



(i) Tower (Stiny, 1980b; Figure 40a)

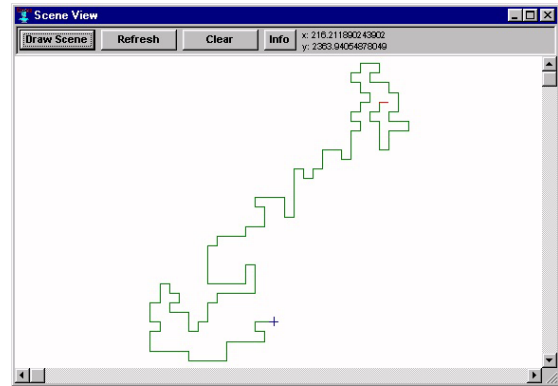
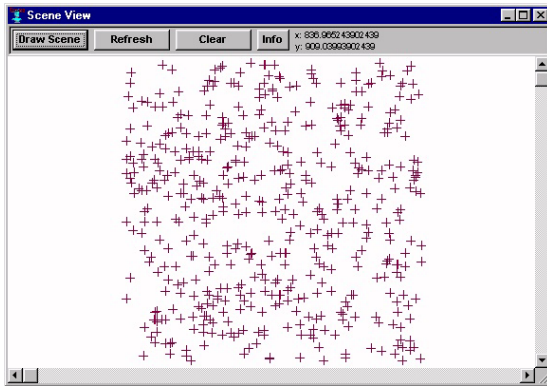


(ii) Stairs (Stiny, 1980b; Figure 40b)

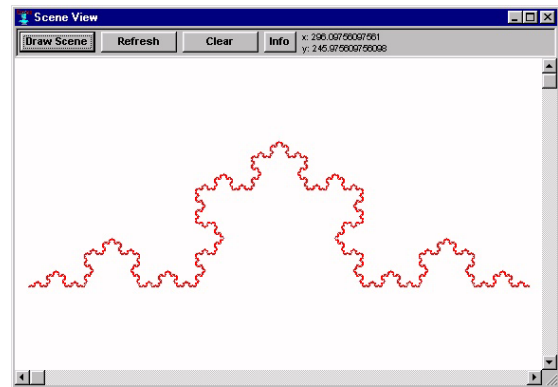
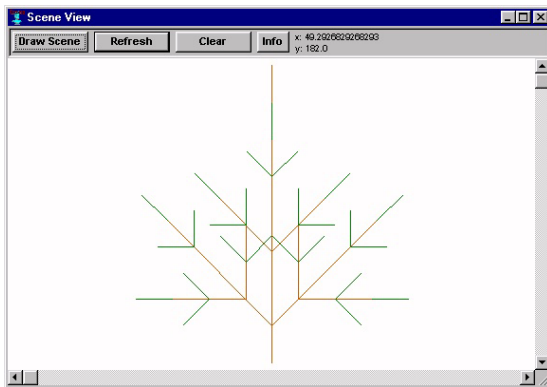


(iii) Spiral Stairs

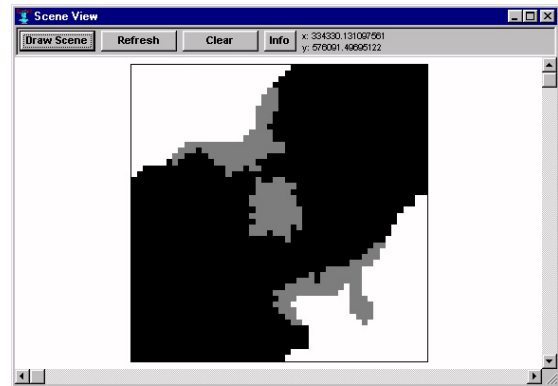
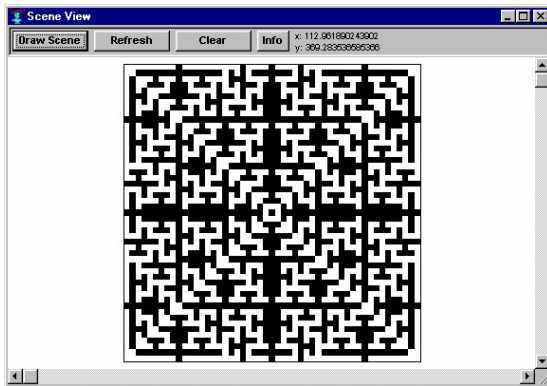
FIGURE B.8.1 APPLICATION OF LGS TO FROEBEL BLOCK DESIGNS



(i) Allocation of Random Points and Paths



(ii) Lindenmayer-Systems and Fractal Designs



(iii) Cellular Automata

FIGURE B.8.2 APPLICATION OF LGS TO OTHER SPATIAL GRAMMARS

Appendix C

Sections relevant to the “Details of Planning” from the 1992 Bermuda Planning Statement

Section 3 – Details of Planning

3.1 General direction to the Board

The Board shall apply the details of planning provisions and other relevant provisions of the Statement in a manner which –

- (a) is consistent with the provisions of the Statement which give the Board certain powers of discretion with respect to details of planning;
- (b) will achieve and satisfy the provisions set out in those paragraphs headed “General Direction to the Board”; and
- (c) ensures that all development is sensitive to, and compatible with, the Bermuda Image.

3.2 Bermuda Image

“The Bermuda Image” means the appearance of Bermuda resulting from a harmonious mix of natural features and man-made elements which produce a visual quality and a character of development which are distinctively Bermudian, and which includes –

- (a) a scale of building which is compatible with the landform and which sits comfortably in its setting;
- (b) the balance and proportions of the traditional building form as exemplified in sturdy residential structures with white pitched roofs, and features and embellishments which distinguish local architecture;
- (c) plentiful lush and colourful sub-tropical vegetation;
- (d) gently rolling hillsides and dense vegetation which effectively blend to screen development and to maintain the illusion of open space and a natural appearance;
- (e) Bermuda stone walls, weathered rock cuts, hedging and planting alongside roads; and
- (f) natural coves, bays, the rocky coastline and islands, with views and glimpses of vividly coloured waters and the ocean.

3.3 Details of Planning

1) “Details of planning” include:

- (a) characteristics which determine the suitability of the site for the use and the form of development proposed, such as –

- (i) the location of the site;
 - (ii) the size and configuration of the site;
 - (iii) the topography and physical features of the site;
 - (iv) the means of a vehicular access to the site; and
 - (v) the potential impact of development on the human, natural and built environments of the area; and
- (b) the way in which the development is arranged and set out on the site, such as–
- (i) the layout and siting of development and the protection given to the natural features of the site;
 - (ii) the extent of excavation and/or filling required in site preparation and the size and design of retaining walls;
 - (iii) the massing, scale and height of development;
 - (iv) the site coverage and the extent and visual impact of hard-surfaced areas;
 - (v) the provision made for vehicular access, parking and servicing;
 - (vi) the arrangements made for pedestrian access and circulation; and
 - (vii) the overall appearance and visual impact of development in the context of its surroundings, including the impact on the natural profile of any visually prominent ridgeline; and
- (c) the character and appearance of development, such as –
- (i) the design and external appearance of buildings;
 - (ii) the building materials used;
 - (iii) the provision of adequate natural light, ventilation and privacy to residential units;
 - (iv) the provision made for private and communal outdoor living and recreation space; and
 - (v) the extent and nature of landscaping proposals.
- 2) The details of planning specified in sub-paragraph (1) are relevant considerations for the Board in the determination of any planning application, whether or not specific reference is made to details of planning in the provisions of the Statement which apply to the subject application, and, for the avoidance of doubt, an application may be refused if the Board is not satisfied with the details of planning.

Appendix D

Rules of the Southcourt Avenue Grammar

The rules of the Southcourt Avenue landscape grammar are listed below. The rules are listed by their thematic rulesets as discussed in detail in Chapter 7.

World Ruleset

`identify-mainroad-and-shore`

At the beginning of the process, identify the main road and the shoreline.

`define-areal-extents`

If the main road and shoreline have been identified, then define the area of the neighborhood extending from the main road to the shore. Assumes the lot-lines are topologically connected to the road and shore.

`start-road-ruleset`

If the initial parameters have been established, begin plotting the road (interpret the plot-roads ruleset).

`start-lotline-ruleset`

If the road has been plotted, begin inserting lot-lines along the road-edges (interpret the adding-lotlines ruleset).

`start-building-parcels`

If lot-lines have been inserted, begin building polygonal parcel objects from the lot-lines (interpret the build-parcels ruleset).

`start-preparing-lots-for-development`

If parcels have been created and are not labelled developable/undevelopable, then begin labelling the parcels as such (interpret the prepare-parcels ruleset).

`start-constructing-houses`

If a developable parcel has not been developed, then insert a house on it (interpret the construct-houses ruleset).

`start-extending-houses`

If a parcel has a house with no additions, then begin creating additions to the house (interpret the extend-houses ruleset).

`associate-houses-with-parcels`

Associate a finished house with the parcel (so that it does not have to be determined by spatial calculations in the future).

`paint-houses`

If a house does not have a material (colour), then create a random material and assign it to the house and its chimney.

start-adding-driveways

If a developed parcel does not have a driveway, then create a driveway (interpret the add-driveways ruleset).

start-adding-walls-hedges

If a developed parcel has a driveway but no walls or hedges, then begin inserting walls and hedges (interpret the plot-walls-hedges ruleset).

start-cleaning-shoreline-walls

If all of the waterfront lots have walls/hedges, then begin cleaning the walls near the shoreline (interpret the shoreline-wall-cleanup ruleset).

start-finishing-walls

If all wall-lines have been created, then create wall objects and set materials (interpret the finish-walls ruleset).

start-adding-trees

If a parcel has not been planted with trees, then begin inserting trees (interpret the add-trees ruleset).

Plot-Roads Ruleset

create-road-centreline

If there is no centreline for the neighborhood, then create it extending straight from the road to the shore.

define-road-centreline-slope-segments

If the road centreline has been created, then split the centreline according to the slope of the underlying land (steep or gentle).

start-clean-up-road-centreline-segments

If the road centreline has been split but not cleaned, then begin cleaning the segments (interpret the clean-road-centreline-segments ruleset)

Clean-Road-Centreline-Segments Ruleset

eliminate-small-road-segments-in-middle

If the length of an interior road centreline is short (< 10m) then dissolve it into the surrounding centrelines.

eliminate-small-road-segments-at-ends

If the length of an end centreline is short (< 10m) then dissolve it into its adjacent centreline.

label-steep-road-segments

If a road centreline has a slope greater than or equal to 17 degrees, then label it as 'steep-slope'.

label-gentle-road-segments

If a road centreline has a slope less than 17 degrees, then label it as 'gentle-slope'.

truncate-road-near-shoreline

If a steep road centreline is connected to the shoreline, then truncate it so that it does not meet the shoreline.

remove-roadlines-outside-area

If any road centrelines or road edges fall outside the site area, then remove the parts outside the area.

start-steep-road-curves

If all road centrelines are labelled as 'gentle-slope' or 'steep-slope', then begin curving the road on steep segments (interpret the steep-road-curves ruleset).

Steep-Road-Curves Ruleset

curve-road-on-steep-slopes

If a road centreline is labelled 'steep-slope' then replace it with an approximation of a sine-curve.

start-modify-road-curves

If a curved road centreline has not been modified, then begin modifying its geometry (interpret the modify-road-curves ruleset).

Modify-Road-Curves Ruleset

extend-curved-road-on-steep-slopes

If a road centreline is labelled 'curved', then extend the final bend near the straighter adjacent centreline.

truncate-road-curve-near-entrance

If any bend in the centreline's path is within 10 metres of the main road, then detour that centreline to be the shortest route to the main road.

create-road-edges

If a road edge has not been created around a road centreline, then buffer the centreline by half of the *road-width*.

create-road-area

If a road edge is not associated with a road area, then create a road area (polygon) from the chain of road edges. Also calculate the elevation of each vertex of the road-area.

Construct-Lot-Lines Ruleset

add-initial-lot-line

If a road edge is longer than the lot-frontage, then add a lot line perpendicular to the road edge, extending it to the boundary of the area.

start-adding-perp-lot-lines

If initial lot-lines have been created on a road edge, then start adding further lot-lines (interpret the adding-perp-lot-lines ruleset).

Adding-Perp-Lot-Lines Ruleset

add-perpendicular-lot-line

If a road edge is longer than the lot-frontage, then add a lot-line perpendicular to the road edge, extending it to the next lot-line, road edge or shoreline.

remove-lot-lines-from-road

If a lot-line is located inside a road area, then delete the lot-line.

`consolidate-centreline-segments`

If no more lot-lines can be added, then reconnect road centrelines that have been split by lot-lines.

`half-the-waterfront-area`

If a road-edge is proximal to the shoreline, create a lot-line there to divide land at that point.

`rotate-lot-lines-near-shoreline`

If a lot-line is within the lot-frontage distance of the shoreline, then rotate the lot-line to be perpendicular to the shore.

`label-remnant-centreline`

If no more lot-lines can be added or rotated, then label the last road centreline so that it will be ignored by the interpreter.

`label-curved-road-edges`

If a road edge is associated with a curved road centreline, then label the road edge as 'curved'.

`start-adding-lotlines-to-road-curves`

If a road edge is labelled 'curved' and does not have any lot-lines, then begin creating lot-lines around the road curves (interpret the adding-lot-lines-to-road-curves ruleset).

Adding-Lot-Lines-To-Road-Curves Ruleset

`add-lot-lines-on-concave-road-curves`

If a road edge is curved and concave, then create a lot-line that is perpendicular to the bisector of the road curve.

`ignore-tiny-concave-corners`

If the concave vertex of a road edge is very close to other vertex, then label it so it is ignored.

`start-cleaning-concave-corner-lots`

If a lot-line has been created for a concave corner then begin to clean the lot-line (interpret the cleaning-concave-corner-lots ruleset).

Cleaning-Concave-Corner-Lots Ruleset

`remove-sketch-for-short-lots`

If the concave vertex of a road edge is within 20 metres of the lot-line created, then remove the lot-line (it would create a nonsensical lot).

`align-sketch-with-neighborhood-axis`

If the created lot-line is valid, then rotate it to be aligned with the central axis of the area.

`create-lot-line-from-sketch`

If the lot-line associated with a concave vertex of a road edge has been cleaned, then extend it so that it joins the nearest lot-lines or road edges.

`add-lot-lines-on-convex-road-curves`

If a road edge is curved and convex, then create a lot-line from the peak of the curve to the site boundary.

start-cleaning-convex-corner-lots

If a lot-line has been created for a convex road edge, then begin to clean the lot-line (interpret the cleaning-convex-corner-lots ruleset).

Cleaning-Convex-Corner-Lots Ruleset

remove-bad-lot-lines-on-convex-road-curves-1

If two lot-lines are very close to each other and nearly parallel, then remove one of them.

remove-bad-lot-lines-on-convex-road-curves-2

If a lot-line and a road edge are very close to each other and nearly parallel, remove the lot-line (since it will create a thin nonsensical lot shape).

Build-Parcels Ruleset

build-parcel-polygons

If a lot-line, road edge or shoreline is not part of a land parcel, then construct parcel polygons from the chain of polylines connected to the selected line.

select-parcels-to-subdivide

If a land parcel is over 900 square metres and has not been subdivided or amalgamated, then subdivide the parcel (interpret the subdivide-parcels ruleset).

Subdivide-Parcels Ruleset

subdivide-large-parcels

If a parcel is not elongated and is approximately rectangular, then subdivide it into regular sized parcels.

subdivide-long-large-parcels

If a parcel is elongated, then subdivide it into parcels that are slightly smaller than normal.

select-parcels-to-amalgamate

If land parcels have not been tested for amalgamation, then begin amalgamating parcels (interpret the amalgamate-parcels ruleset) until 20% of the site has been amalgamated.

Amalgamate-Parcels Ruleset

amalgamate-two-adjacent-parcels

If two adjacent parcels each have areas less than 700 square metres, then amalgamate the two parcels into one.

amalgamate-three-adjacent-parcels

If three adjacent parcels each have areas less than 700 square metres, then amalgamate the three parcels into either one or two parcels.

amalgamate-three-adjacent-parcels-at-shoreline

If a waterfront parcel that is not adjacent to a road edge, but is adjacent to two other parcels of less than 700 square metres, then amalgamate the three parcels into one (if it were adjacent to the road, it would cut off access to another waterfront lot).

Prepare-Parcels-For-Development Ruleset

label-developable-parcels

If a parcel has an area greater than 400 square metres, then label it as ‘developable’.

label-undevelopable-parcels

If a parcel has an area less than or equal to 400 square metres, then label it as ‘undevelopable’.

label-corner-lots

If a parcel is adjacent to a curved road edge, then label it as a ‘corner-lot’.

label-waterfront-lots

If a parcel is adjacent to a shoreline, then label it as a ‘waterfront-lot’.

regrade-parcels

If a developable parcel has a mean slope value of greater than 3 percent, then adjust the elevation values within that parcel to be within 0.5 metres from the mean elevation of the parcel (level the parcel without completely flattening it).

Construct-House Ruleset

insert-house

If a developable parcel does not have a house inside of it, then insert a house in the centre of the parcel.

align-house-with-lot-line

If a house is not on a waterfront lot, align the long axis of the house with the nearest lot-line.

align-house-with-shoreline

If a house is on a waterfront lot, then align the long axis of the house with the shoreline axis.

align-house-with-corner-axis

If a house is on a concave road corner lot, then align the house to make best use of the length of the parcel (along the angular bisector of the road corner).

associate-front-of-house-to-road

If a house does not have a frontal direction, then assign its front (a vector) to the side facing the road.

associate-front-of-house-to-shore

If a house on a waterfront lot does not have a frontal direction, then assign its front (a vector) to the side facing the road.

relocate-house-on-large-lots

If a house is centred in an amalgamated lot, then move the house towards the point of maximum elevation in the parcel (for best vantage).

relocate-house-on-waterfront-lots

If a house is centred in a waterfront lot, then move the house away from the direction of the shoreline (away from the danger of cliffs).

move-house-from-main-road

If a house is too close to the main road (as opposed to the road bisecting the site), then move it away from the road edge (away from busy traffic).

`move-house-from-avenue`

If a house is too close to the edge of the central road, then move it away from the road edge.

Extend-Houses Ruleset

`start-additions-ruleset`

If a house has no additions, then begin creating additions (interpret the house-additions ruleset) until between zero and four additions have been generated.

House-Additions Ruleset

`create-addition`

If the maximum number of additions for a house has not been reached, then insert an addition centred on a random point on the perimeter wall of the house. Align the addition to be parallel/perpendicular to the house. Begin determining whether the addition is valid (interpret the removing-bad-house-additions ruleset).

Removing-Bad-House-Additions Ruleset

`reduce-narrow-addition-on-short-segment`

If a long and narrow addition is located on a short wall of a house, then reduce the length of the addition.

`reduce-narrow-addition-on-long-segment`

If a long and narrow addition is located on a long wall of a house, then reduce the length of the addition (using different reductions than above).

`remove-addition-near-lot-line`

If an addition is within two metres of a parcel's edge, then remove the addition.

`remove-addition-with-narrow-gap`

If two additions are less than one metre apart (and not overlapping), then remove one of them.

`remove-proximal-additions`

If the centroids of two additions are within three metres of each other, then remove one of them.

`build-chimneys`

If a house does not have a chimney, then insert a chimney on a wall (but not near a corner).

Add-Driveways Ruleset

`place-driveways`

If a developed parcel does not have a driveway, then place a potential driveway at each end of the road-side of the parcel.

`align-and-move-driveways`

If a driveway has not been aligned, then align it to the edges of the parcel and move its centre point further inside the parcel area.

`remove-dway-over-lot-line`

If a driveway intersects a lot-line (as is possible in the more awkward lots at the road curves) then remove the driveway.

`remove-steeply-sloping-dway`

If the average slopes of some (but not all) driveways are too steep, then remove the steepest driveway.

`remove-furthest-dway-from-house`

If a driveway is one of the furthest driveways from a house, then remove it (a less convenient driveway).

`remove-random-driveway`

If the above rules have not reduced the number of potential driveways to one then remove a driveway randomly.

`ensure-driveway-joins-avenue`

If the driveway does not overlap the adjacent road polygon, then extend it so that it does.

`extend-driveway-past-house`

If there is not a house in the way, and the slope of the land is gentle enough, then extend the driveway to either half-way along the house or past the house.

`extend-driveway-to-house`

If there is a house in the way, and the slope of the land is gentle enough, then extend the driveway to the house.

`place-driveway-for-waterfront-lots`

If a developed parcel is a waterfront lot that is adjacent to the road, then place the driveway at the road-edge furthest from the shore.

`share-driveway-for-waterfront-lots`

If a developed parcel is a waterfront lot that is not adjacent to the road, then extend the driveway of the neighboring parcel into the current parcel.

`level-driveway`

If a driveway has been completed, then set the elevation values within the driveway to the mean elevation value (flatten it).

`create-garage`

If a driveway is extended past a house, then, with a 60% probability, insert a garage at the end of the driveway.

`paint-garage`

If a garage does not have a material (colour), then set the material to that of the relevant house.

Plot-Walls-Hedges Ruleset

`start-creating-boundary-structures`

If the locations of boundary structures have not been determined for a parcel, then begin creating them (interpret the boundary-structures ruleset).

Boundary-Structures Ruleset

`boundary-marker-between-parcels`

If a lot-line (except for the boundary of the site) does not have a boundary-marker, then create one along the length of the line.

`boundary-marker-at-roadside`

If a road edge does not have a boundary-marker, then create one along the chain of road edges that borders the adjacent parcel.

`visual-barrier-from-neighbours`

If a boundary marker is on a lot-line, then create visual barrier from the boundary marker.

`visual-barrier-in-backyard`

If a boundary marker is on a lot-line, then split it at the back of the house and create a visual barrier from the back yard portion of the line.

`visual-barrier-from-roadside`

If a boundary marker is on a road edge, then create a visual barrier from the boundary marker.

`retaining-structures`

If there is a sharp difference in elevation on either side of a boundary marker, create a retaining wall along the length of the boundary marker.

`lotline-wall-from-visual-barrier`

If a visual barrier is on a lot-line, then create either a sloping or flat wall-line from the visual barrier and calculate its elevations and heights.

`roadside-wall-from-visual-barrier`

If a visual barrier is on a road edge, then create either a sloping or flat wall-line from the visual barrier and calculate its elevations and heights.

`main-road-wall-from-visual-barrier`

If a visual barrier is on a road edge that is labelled as the main road, then create a flat wall-line from the visual barrier and calculate its elevations and heights (higher than other roadside walls to block traffic).

`hedge-from-visual-barrier`

If a visual barrier is on a lot-line, then create a hedge from the visual barrier and calculate its elevations and height.

`wall-or-hedge-from-boundary-marker`

If a boundary marker exists, then create either a (sloping or flat) wall-line or hedge from the boundary marker and calculate its elevations and heights.

`detach-hedges-from-walls`

If a hedge and a wall are connected, then detach the end of the hedge from the wall (so that they have separate nodes and can thus have different elevations and heights at the node).

`tie-ret-road-wall-heights`

If a roadside wall-line is connected to a retaining wall-line, then adjust the height of the roadside wall such that the top is level with that of the retaining-wall.

tie-nonret-road-wall-heights

If a roadside wall-line is connected to a wall-line (but not a retaining wall), then adjust the height of the end of the latter wall to that of the roadside wall.

Shoreline-Walls Ruleset

create-shoreline-wall

If a parcel is a waterfront lot and does not have a shoreline wall, then create a wall-line that approximates a boundary between the flat usable land and the steeper rocky shore. Set the elevations and heights of the wall-line to reflect a low sloping wall.

move-shoreline-wall-near-house

If a shoreline wall intersects or is near a house, then create a new wall-line that is three metres from the house (creating a yard area), extend it to the parcel boundaries, and set the elevations and heights as above.

Shoreline-Wall-Cleaning Ruleset

join-shoreline-wall-to-boundary

If a shoreline wall-line does not topologically intersect the boundary of a parcel, and it can be extended to the boundary without crossing other objects, then extend the wall-line and intersect the nearest hedge, wall-line, or lot-line.

split-boundary-at-shoreline-wall

If a shoreline wall-line intersects another wall-line or hedge that is connected to the shoreline, then split the latter line at the intersection point with the shoreline wall-line.

remove-shoreline-wall-dangles

If the above rule has created remnant ends of the shoreline wall-line then remove them.

remove-wall-hedge-on-shore

If a wall-line or hedge is connected to a shoreline, then remove the wall-line or hedge.

remove-wall-hedge-in-driveway

If a wall-line or hedge traverses a driveway, then truncate the wall-line or hedge at the edge of the driveway.

Finish-Walls Ruleset

create-wall-from-wall-line

If a wall-line has not been used to create a wall, then create a sloping or flat wall by buffering the wall-line by half the width of a wall.

create-shoreline-wall-from-wall-line

If a shoreline wall-line has not been used to create a wall, then create a sloping wall by buffering the wall-line by half the width of a wall.

colour-roadside-wall

If a wall is on a road edge and does not have a material, then set the material to that of the nearest house towards the interior of the parcel.

colour-boundary-wall

If a wall is on a lot-line and does not have a material, then set the material to that of the nearest house.

colour-shoreline-wall

If a shoreline wall does not have a material, then set the material to grey (stone).

Tree-Rules

area-boundary-trees

If a line is labelled as the site boundary and does not have trees, then create a buffer polygon of two metres around the line and generate random tree points within the polygon to a density of 0.15 and a minimum distance of two metres between trees. Set the tree's species to be randomly selected with a higher probability of being a Bay-Grape if the tree is near the shoreline.

shoreline-area-boundary-trees

If a line is as above and connected to the shoreline, then generate trees as above except that tree points between a shoreline-wall and the shoreline are removed. The species are set as above.

front-yard-trees

If a developed parcel (not a waterfront lot) has not had trees planted in the front yard, then plant either a row of zero, one or two trees or (if the parcel is large) create a linear polygonal area and create random trees inside that area. Set the species probabilistically according to frequencies of occurrence on site.

front-yard-trees-near-main-road

If a developed land parcel borders the main road, then generate front-yard trees as above but relative to the main road rather than the central avenue (creates a vegetative buffer from traffic).

define-back-yard

If a developed parcel (not a waterfront lot) borders a site boundary, then define a back yard as the area between the rear wall of the house (away from the defined frontal vector) and the site boundary.

define-back-yard-near-main-road

If a developed parcel borders a main road edge, then define a back yard as the area between the rear wall of the house (away from the main road) and the site boundary.

define-back-yard-at-waterfront

If a developed parcel is a waterfront lot, then define a back yard as the area between the rear wall of the house (away from the shoreline) and the site boundary.

back-yard-trees

If a back yard has a slope of less than five degrees, then (with a probability of 20%) generate tree points randomly within the back yard area with a minimum distance of five metres between trees. Set the species probabilistically according to frequencies of occurrence on the site.

create-banana-patch

If a back yard has a large area (> 80 metres) and a low average slope (< 4 degrees) then, with a 50% probability, create a rectangular banana patch in the corner nearest the site boundary, with an intermediate distance of at least three metres. Align the patch with the lot-lines and generate six to ten random banana tree points inside the patch.

remove-trees-from-banana-patch

If a tree other than a banana tree is located inside of a banana patch, then remove the tree.

remove-trees-from-road

If a tree is located inside or within 0.5 metre of a road area (including driveways), then remove the tree.

remove-trees-from-buildings

If a tree is located inside or within 0.5 metre of a building, then remove the tree.

remove-trees-from-walls

If a tree is located inside or within 0.5 metre of a wall, then remove the tree.

References

- Agarwal, M. and Cagan, J. (1998). "A Blend of Different Tastes: The Language of Coffeemakers", *Environment and Planning B: Planning and Design*, 25(2), 205-226.
- Alexander, C. (1977). *A Pattern Language: Towns, Buildings, Construction*. New York: Oxford University Press. See also <http://patternlanguage.com>
- Alexander, C. (1979). *A Timeless Way of Building*. New York: Oxford University Press.
- Alexander, E.R. (1998). "Doing the 'Impossible': Notes for a General Theory of Planning", *Environment and Planning B: Planning and Design*, 25, 667-680.
- Al-Kodmany, K. (2000). "Extending Geographic Information Systems (GIS) to Meet Neighbourhood Planning Needs: Recent Developments in the Work of the University of Illinois at Chicago", *URISA Journal*, 12(3), 19-37.
- Allen, H. (1936). *Residence in Bermuda*. No publisher specified.
- ANSI, American National Standards Institute. (1994). "X3.226:1994 American National Standard for Programming Language Common LISP (X3J13)". New York, New York: American National Standards Institute.
- Bartuska, T.J., and Young, G.L. (1994). *The Built Environment: Creative Inquiry into Design and Planning*. Menlo Park, California: Crisp Publications.
- Batty, M., Couclelis, H. and Eichen, M. (1997). "Urban Systems as Cellular Automata", *Environment and Planning B: Planning and Design*, 24, 159-164.
- Batty, M. and Longley, P. (1994). *Fractal Cities*. London, Academic Press.
- Bench-Capon, T.J.M. (1990). *Knowledge Representation: An Approach to Artificial Intelligence*. The APIC Series, No. 32. Toronto: Academic Press.
- Bermuda Biodiversity Project, The (2001). Personal Communication. Bermuda: Bermuda Aquarium Museum and Zoo.
- Bermuda Census Office. (1992). "The 1991 Census of Population and Housing - An Executive Report", Hamilton, Bermuda: Government of Bermuda, Census Office.
- Bermuda Department of Planning, The (1974). The Bermuda Development Plan 1974. Hamilton, Bermuda: Government of Bermuda, Department of Planning.
- Bermuda Department of Planning, The (1983). *The Bermuda Development Plan 1983*, Hamilton, Bermuda: Government of Bermuda, Department of Planning.
- Bermuda Department of Planning, The (1989). "Bermuda 2000: The Changing Face of an Island", Bermuda Development Plan Discussion Paper Number 2, Hamilton, Bermuda: Government of Bermuda, Department of Planning.

- Bermuda Department of Planning, The (1990). "Visual Quality in Bermuda", Bermuda Development Plan Discussion Paper Number 6, Hamilton, Bermuda: Government of Bermuda, Department of Planning.
- Bermuda Department of Planning, The (1991). *Bermuda 2000: Facing the Future*, Hamilton, Bermuda: Government of Bermuda, Department of Planning.
- Bermuda Department of Planning, The (1992a). *The Bermuda Plan 1992*, Hamilton, Government of Bermuda, Bermuda: Department of Planning.
- Bermuda Department of Planning, The (1992b). *The Bermuda Plan 1992 - Planning Statement*, Hamilton, Government of Bermuda, Bermuda: Department of Planning.
- Bermuda Department of Planning, The (2000). Personal communication.
- Bermuda Department of Statistics, The (1999) "Bermuda Facts and Figures - 1992 Edition", Pamphlet. Hamilton, Bermuda: Government Statistical Department.
- Bermuda Department of Tourism, The (1996-1999). Bermuda Airport Studies – Visitor Surveys.
- Bermuda Land Valuation Office, The (2000). Personal communication.
- Bermuda Ministry of Works & Engineering, The (2000). Personal communication.
- Bermuda National Trust, The (1995). Bermuda's Architectural Heritage: Devonshire. Bermuda: Bermuda National Trust.
- Bermuda National Trust, The (1998). Bermuda's Architectural Heritage: St. George's. Bermuda: Bermuda National Trust.
- Bermuda National Trust, The (2000). Bermuda's Architectural Heritage: Sandys. Bermuda: Bermuda National Trust.
- Boots, B. and South, R. (1997). "Modelling Retail Trade Areas Using Higher-Order, Multiplicatively Weighted Voronoi Diagrams", *Journal of Retailing*, 73(4), 519-536.
- Bourassa, S.C. (1991). *The Aesthetics of Landscape*. New York: Belhaven Press.
- Brail, R.K. (1990). "ERS: Prolog to an Expert System for Transportation Planning", in *Expert Systems: Applications to Urban Planning*, Kim, Wiggins and Wright (eds.), 87-104.
- Brand, S. (1994). *How Buildings Learn: What Happens After They're Built*. New York: Viking Penguin.
- Brassel, K.E. and Weibel, R. (1988). "A Review and Framework of Automated Map Generalization", *International Journal of Geographical Information Systems*, 2(3), 229-244.
- British Columbia Ministry of Forests. (1996). British Columbia Ministry of Forests Policy Manual, Policy 4.2 – Forest Landscape Management. Available at <http://www.for.gov.bc.ca/tasb/manuals/policy/resmngmt/rm4-2.htm>

- Buttenfield, B.P. and McMaster, R.B. eds. (1991). *Map Generalization: Making Rules for Knowledge Representation*. New York, New York: Wiley.
- Carlisle Scott, A., Clayton, J.E., and Gibson, E.L. (1991). *A Practical Guide to Knowledge Acquisition*. Reading, Massachusetts: Addison-Wesley Publishing Company.
- Carlson, C. (1993). "Describing Spaces of Rectangular Dissections via Grammatical Programming", in *CAAD Futures '93*, U. Flemming and S. Van Wyk (eds.), Amsterdam: Elsevier Science Publications, 143-158.
- Carlson, C., Woodbury, R., and McKelvey, R. (1991). "An Introduction to Structure and Structure Grammars", *Environment and Planning B: Planning and Design*, 18, 417-426.
- Chase, S.C. (1989). "Shapes and Shape Grammars: From Mathematical Model to Computer Implementation", *Environment and Planning B: Planning and Design*, 16, 215-242.
- Chase, S.C. (1999). "Supporting Emergence in Geographic Information Systems", *Environment and Planning B: Planning and Design*, 26, 33-44.
- Chiou, S-C. and Krishnamurti, R. (1995). "The Grammar of Taiwanese Traditional Vernacular Dwellings", *Environment and Planning B: Planning and Design: Planning and Design*, 22, 689-720.
- Chomsky, N. (1957). *Syntactic Structures*. The Hague: Mouton & Co.
- Chomsky, N. (1965). *Aspects of the Theory of Syntax*. Cambridge, Massachusetts: The MIT Press.
- Chomsky, N. (1966). *Cartesian Linguistics*. New York: Harper & Row.
- Chomsky, N. (1981). *Lectures on Government and Binding*. Dordrecht, Holland: Foris Publications.
- Chomsky, N. (1982). *Some Concepts and Consequences of the Theory of Government and Binding*. Cambridge, Massachusetts: The MIT Press.
- Chomsky, N. (1995). *The Minimalist Programme*. Cambridge, Massachusetts: The MIT Press.
- Clarke, K., Brass, J., and Riggan, P. (1994). "A Cellular Automaton Model of Wild Fire Propagation and Extinction", *Photogrammetric Engineering and Remote Sensing*, 60, 1355-67.
- Cosgrove, D. (1984). *Social Formation and Symbolic Landscape*. London: Croom Helm.
- Coyne, R.D., and Gero, J.S. (1985). "Design Knowledge and Sequential Plans", *Environment and Planning B: Planning and Design*, 12, 401-418.
- Cullen, G. (1971). *The Concise Townscape*. New York: Van Nostrand Reinhold.
- Cycorp (2000). "The Cyc Knowledge Server". Available at <http://www.cyc.com/products2.html>
- Davidoff, P. and Reiner, T.A. (1973). "A Choice Theory of Planning" in *A Reader in Planning Theory*, A. Faludi (ed.). Oxford: Pergamon.

- Davis, J.R. and Grant, I.W. (1990). "ADAPT: A Knowledge-Based Decision Support system for Producing Zoning Schemes", in *Expert Systems: Applications to Urban Planning*, Kim, Wiggins and Wright (eds.), 67-85.
- Davis, J.R., and McDonald, G. (1993). "Applying a Rule Based Decision Support System to Local Government Planning", in *Expert Systems in Environmental Planning*, J.R. Wright, L.L. Wiggins, R. Jain, and T.J. Kim (eds.). New York: Springer-Verlag.
- de Reffye, P. (1994). "Computer Simulation of Plant Growth", in *Frontiers of Scientific Visualization*, C.A. Pickover and S.K. Tewkesbury (eds.), Toronto: John Wiley and Sons.
- de Reffye, P., Edelin, C., Francon, J, Jaeger, M., and Puech, C. (1988). "Plant Models Faithful to Botanical Structures and Development", *Computer Graphics*, Anaheim, CA: ACM SIGGRAPH.
- DETR, Department of Environment, Transportation and Regions. (2000). "Countryside and Rights of Way Act 2000: Fact Sheets". Available at: <http://www.wildlife-countryside.detr.gov.uk/cl/bill/factsheet>
- Dilley, R.S. (1986). "Tourist Brochures and Tourist Images", *Canadian Geographer*, 30(1), 59-65.
- Eckbo, G. (1969). *The Landscape We See*. New York: McGraw-Hill.
- Egenhofer, M. and Herring, J. (1990). "A Mathematical Framework for the Definition of Topological Relationships", *Proceedings of the Fourth International Symposium on Spatial Data Handling*, Zurich, Switzerland, 2: 803-813.
- Egenhofer, M. and Herring, J. (1991). "Categorizing Topological Spatial Relations Between Point, Line and Area Objects". Technical Report, University of Maine.
- ESRI, Environmental Systems Research Institute. (1998). "ESRI Shapefile Technical Description, An ESRI White Paper – July 1998", Redlands, California: ESRI. Available at: <http://www.esri.com/>
- ESRI, Environmental Systems Research Institute. (1999). "Arc News: Special Insert on ArcInfo 8", Redlands, California: ESRI. Available at: <http://www.esri.com/>
- ESRI, Environmental Systems Research Institute. (2000). "Map Generalization in GIS: Practical Solutions with Workstation ArcInfo Software, An ESRI White Paper – July 2000", Redlands, California: ESRI. Available at: <http://www.esri.com/>
- Faludi, A. (1973). "What is Planning Theory?" in *A Reader in Planning Theory*, A. Faludi (ed.). Oxford: Pergamon.
- Fang, H., Mikroudou, G.K. and Pamukcu, S. (1993). "Multi-domain Expert Systems for Hazardous Waste Site Investigations", in *Expert Systems in Environmental Planning*, J.R. Wright, L.L. Wiggins, R. Jain, and T.J. Kim (eds.). New York: Springer-Verlag.
- Fawcett, W. (1986). "An Elementary Rule Interpreter for Architectural Design", in *Knowledge Engineering and Computer Modelling in CAD*, A. Smith (ed.). London: Butterworths, 259-269.
- Findikaki, I. (1990). "SISES: An Expert System for Site Selection", in *Expert Systems: Applications to Urban Planning*, Kim, Wiggins and Wright (eds.), 125-132.

- Flemming, U. (1981). "The Structure of Bungalow Plans", *Environment and Planning B: Planning and Design*, 8(4), 393-404.
- Flemming, U. (1987). "More Than the Sum of Parts: The Grammar of Queen Anne Houses", *Environment and Planning B: Planning and Design*, 14, 323-350.
- Flemming, U., Gindroz, R., Coyne, R., and Pitharadian, S. (1985). "A Pattern Book for Shadyside", Technical Report, Department of Architecture, Carnegie-Mellon University, Pittsburgh, Pennsylvania.
- Flemming, U., and Van Wyk, S. (eds.) (1993). *Proceedings of CAAD Futures '93*. Amsterdam: Elsevier Science Publications.
- Forberg, A., and Mayer, H. (2002). "Generalization of 3D Building Data Based on Scale-Spaces", *Proceedings of the Joint International Symposium on Geospatial Theory, Processing and Applications*, July 2002, Canadian Institute of Geomatics, Ottawa, Canada.
- Foulkes, D. (1978). *A Grammar of Dreams*. New York: Basic Books.
- Friedell, M., and Shulmann, J.L. (1990). "Constrained, Grammar-Directed Generation of Landscapes", *Proceedings of Graphics Interface '90*, 244-251.
- Gero, J.S., and Tyugu, E. (eds.). (1994). *Formal Design Methods for CAD*. Amsterdam: Elsevier Science Publications.
- Gips, J. (1975). *Shape Grammars and Their Uses: Artificial Perception, Shape Generation and Computer Aesthetics*. Basel: Birkhauser.
- Gips, J. (1999). "Computer Implementation of Shape Grammars", invited paper, Workshop on Shape Computation, MIT, 1999. Available at <http://www.cs.bc.edu/~gips>
- Gurr, T. (1984). "The Quality of Life and Prospects for Change in Bermuda: A Report to the Government of Bermuda on a Sample Survey", Consultant's Report for the Cabinet Office, Government of Bermuda. Available from the Bermuda Archives.
- Gussow, A. (1979). "Conserving the Magnitude of Uselessness: A Philosophical Perspective", in *Proceedings of Our National Landscape: A Conference on Applied Techniques for Analysis and Management of the Visual Resource*, by G.E. Elsner and R.C. Smardon, compilers. General Technical Report PSW-35, United States Department of Agriculture, Forest Service, Pacific Southwest Forest and Range Experiment Station, Berkeley, California, 6-11.
- Hasemer, T., and Domingue, J. (1989). *Common LISP Programming for Artificial Intelligence*. Reading, Massachusetts: Addison-Wesley Publishing Company.
- Hayward, S. and Rowlinson, B. (1981). "Land", in S. Hayward, V. Holt Gomez, and W. Sterrer (eds.), *Bermuda's Delicate Balance: People and the Environment*, Hamilton, Bermuda: Island Press, pp. 71-92.
- Heisserman, J. (1991). "Generative Geometric Design and Boundary Solid Grammars", PhD Dissertation, Department of Architecture, Carnegie Mellon University, Pittsburgh.

- Heisserman, J. (1994). "Generative Geometric Design", *IEEE Computer Graphics & Applications*, 4(2), 37-45.
- Heisserman, J. and Woodbury, R. (1994). "Geometric Design with Boundary Solid Grammars", in *Formal Design Methods for CAD*, J.S. Gero and E. Tyugu (eds.). Amsterdam: Elsevier Science Publications, 85-105.
- Herbert, T., Sanders, I., and Mills, G. (1994). "African shape grammar: a language of linear Ndebele homestead", *Environment and Planning B: Planning and Design*, 21, 453-476.
- Higuchi, T. (1983). *The Visual and Spatial Structure of Landscapes*. Translated by Charles S. Terry. Cambridge, Massachusetts: The MIT Press.
- Hiller, B. (1996). *Space is the Machine*. Cambridge, England: Cambridge University Press.
- Hiller, B. (1999). "The Hidden Geometry of Deformed Grids", *Environment and Planning B: Planning and Design*, 26, 169-191.
- Hiller, B. (2001). "The Common Language of Space". Book forthcoming. Paper available at <http://www.spacesyntax.com/publications/commonlang.html>
- Hillier, B. and Hanson, J. (1984). *The Social Logic of Space*. Cambridge, England: Cambridge University Press.
- Hough, M. (1990). *Out of Place*. New Haven, Massachusetts: Yale University Press.
- Humphreys, J.S. (1923). *Bermuda Houses*. Facsimile Edition (1993) edited by E.C. Harris and S. Frith-Brown, The Royal Naval Dockyard, Bermuda: Bermuda Maritime Museum Press.
- Informatix Software International. (1993). "Classic GDS Reference Manuals". London, U.K.
- Informatix Software International. (2000). "Classic GDS Central". Available on the Internet at <http://www.informatix.co.uk/classicgds.htm>
- IEA/LI, The Institute of Environmental Assessment and the Landscape Institute. (1995). *Guidelines for Landscape and Visual Impact Assessment*. London, UK: E. & F.N. Spon Press.
- ISI, The University of Southern California's Information Sciences Institute. (1999). "The Loom Project Home Page". Available at <http://www.isi.edu/isd/LOOM/>
- ISI, The University of Southern California's Information Sciences Institute. (2000). "PowerLoom Knowledge Representation System". Available at: <http://www.isi.edu/isd/LOOM/PowerLoom/index.html>
- Jackson, P. (1999). *Introduction to Expert Systems*. Reading, Massachusetts: Addison-Wesley.
- Jacobs, A.B. (1993). *Great Streets*. Cambridge, Massachusetts: The MIT Press.
- Jakle, J.A. (1987). *The Visual Elements of Landscape*. Amherst, MA: University of Massachusetts Press.

- Kada, M. (2002). "Automatic Generalization of 3D Building Models", *Proceedings of the Joint International Symposium on Geospatial Theory, Processing and Applications*, July 2002, Canadian Institute of Geomatics, Ottawa, Canada.
- Keene, Sonya E. (1989). *Object-oriented Programming in Common LISP: A Programmer's Guide to CLOS*. Reading, Massachusetts: Addison-Wesley Publishing Company.
- Kim, T.J., Wiggins, L.L., and Wright, J.R. (1990). *Expert Systems: Applications to Urban Planning*. New York: Springer-Verlag.
- Knight, T.W. (1980). "The Generation of Heppelwhite-style Chair-back Designs", *Environment and Planning B: Planning and Design*, Vol. 7(2), 227-238.
- Knight, T.W. (1989a). "Transformations of De Stijl Art - the Paintings of Georges Vantongerloo and Fritz Glarner", *Environment and Planning B: Planning and Design*, Vol. 16(1), 51-98.
- Knight, T.W. (1989b). "Color Grammars: Designing with Lines and Colors", *Environment and Planning B: Planning and Design*, Vol. 16(4), 417-449.
- Knight, T.W. (1990). "Mughul Gardens Revisited", *Environment and Planning B: Planning and Design*, Vol. 17(1), 73-84.
- Knight, T.W. (1991). "Designing with Grammars", in *CAAD Futures '91*, G.N. Schmitt (ed.). Wiesbaden: Vieweg, Bertelsmann Publishing Group International.
- Knight, T. (1999a). "Shape Grammars: Six Types", *Environment and Planning B: Planning and Design*, 26(1), 15-32.
- Knight, T. (1999b). "Shape Grammars: Five Questions", *Environment and Planning B: Planning and Design*, 26(4), 477-502.
- Koning, H. and Eizenberg, J. (1981). "The Language of the Prairie: Frank Lloyd Wright's Prairie Houses", *Environment and Planning B: Planning and Design*, Vol. 8, 295-323.
- Krishnamurti, R. (1980). "The Arithmetic of Shapes", *Environment and Planning B: Planning and Design*, 7, 463-484.
- Krishnamurti, R. (1981). "The Construction of Shapes", *Environment and Planning B: Planning and Design*, 8, 5-10.
- Krishnamurti, R. (1982). "SGI: A Shape Grammar Interpreter", Centre for Configurational Studies, Open University, and University of Edinburgh.
- Krishnamurti, R. and Giraud C. (1986). "Towards a Shape Editor: The Implementation of a Shape Generation System", *Environment and Planning B: Planning and Design*, 13, 391-404.
- Krishnamurti, R. and Stouffs, R. (1993). "Spatial Grammars: Motivation, Comparison, and New Results", in *CAAD Futures '93*, U. Flemming and S. Van Wyk (eds.), Amsterdam: Elsevier Science Publications, 57-74.

- Krzanowski, R., and Raper, J. (2001). *Spatial Evolutionary Modeling*. Spatial Information Series. Oxford, UK: Oxford University Press.
- Laurini, R. (1998). "Groupware for Urban Planning: An Introduction", *Computers, Environment and Urban Systems*, 22(4), 317-333.
- Lawless, J.A., and Miller, M.M. (1991). *Understanding CLOS: The Common LISP Object System*. Woburn, Massachusetts: Digital Press.
- Lee, Y.C. and Wiggins, L.L. (1990). "MEDIATOR: An Expert System to Facilitate Environmental Dispute Resolution", in *Expert Systems: Applications to Urban Planning*, Kim, Wiggins and Wright (eds.), 197-221.
- Lenat, D.B. (1995). "Cyc: A Large-Scale Investment in Knowledge Infrastructure", *Communications of the ACM*, 38(11). Also see <http://www.cycorp.com>
- Levinsohn, A. (2000a). "Use Spatial Data to Model Geographic Knowledge", *GEOWorld*, April 2000, 28.
- Levinsohn, A. (2000b). "Feature Definitions Replace Maps as the Principal Model of Geography", *GEOWorld*, June 2000, 30.
- Lindenmayer, A. (1968). "Mathematical Models for Cellular Interaction in Development, Parts I and II", *Journal of Theoretical Biology*, 18, 280-315.
- Litton, R.B. (1982). "Visual Assessment of Natural Landscapes", in *Environmental Aesthetics: Essays in Interpretation*, by B. Sadler and A. Carlson, eds., Western Geographical Series. Victoria, British Columbia: Department of Geography, University of Victoria, 20.
- Liu, Y. (1993). "A Connectionist Approach to Shape Recognition and Transformation", in *CAAD Futures '93*, U. Flemming and S. Van Wyk (eds.), Amsterdam: Elsevier Science Publications, 19-36.
- Lynch, K. (1960). *The Image of the City*. Cambridge, Massachusetts: The MIT Press.
- Lynch, K. (1976). *Managing the Sense of a Region*. Cambridge, Massachusetts: The MIT Press.
- Lynch, K. (1984). *Site Planning*. Cambridge, Massachusetts: The MIT Press.
- MacGregor, R. (1999). "Retrospective on Loom". Available at http://www.isi.edu/isd/LOOM/papers/macgregor/Loom_Retrospective.html
- Maidment, D.R., and Evans, T.A. (1993). "Regulating the Municipal Environment Using an Expert Geographic Information System", in *Expert Systems in Environmental Planning*, J.R. Wright, L.L. Wiggins, R. Jain, and T.J. Kim (eds.). New York: Springer-Verlag.
- Mark, D.M. and Egenhofer, M.J. (1994). "Modeling Spatial Relations Between Lines and Regions: Combining Formal Mathematical Models and Human Subjects Testing", *Cartography and Geographical Information Systems*, 21(3), 195-212.
- Mayall, K. (1993). "Visualization of Landscape: Towards the Integration of Geographic Information Systems and Computer-Assisted Design", Unpublished Masters thesis, School of Urban & Regional Planning, University of Waterloo, Ontario, Canada.

- Mayall, K., Hall, G.B., Seebohm, T. (1994). "Integrating GIS and CAD to Visualise Landscape Change", *GIS World*, Vol. 7(9), 46-49. Also available at <http://www.fes.uwaterloo.ca/u/kmayall/Research>
- Mayall, K., and Hall, G.B. (1994). "Information Systems and Three-Dimensional Modelling in Landscape Visualization", *Proceedings of the Urban and Regional Information Systems Association (URISA) International Conference 94*, pp. 796-804.
- Mayall, K., Seebohm, T., and Hall, G.B. (1994). "From 2-D to 3-D: Integrating GIS and CAD to Visualize Landscape Change", *Proceedings of the GDS Users World Conference 1994*.
- McCarthy, J. (1960). "Recursive Functions of Symbolic Expressions and Their Computation by Machine", *Communications of the Association for Computing Machinery*, 184-195.
- McCullough, M., Mitchell, W.J., and Purcell, P. (eds.). (1990) *The Electronic Design Studio*, Cambridge, MA: MIT Press.
- Meinig, D.W. (1979). *The Interpretation of Ordinary Landscapes*. New York: New York: Oxford University Press.
- Meyer, F.S. (1888/1957). *Handbook of Ornament*. New York: Dover.
- Mitchell, W.J. (1990). *The Logic of Architecture: Design, Computation, and Cognition*. Cambridge, Mass.: MIT Press.
- Mitchell, W.J., Liggett, R.S., Pollalis, S., and Tan, M. (1991). "Integrating Shape Grammars and Design Analysis", in *CAAD Futures '91*, G.N. Schmitt (ed.). Wiesbaden: Vieweg, Bertelsmann Publishing Group International.
- Mitchell, W.J., Liggett, R.S., and Tan, M. (1990). "Top-Down Knowledge-Based Design", in *The Electronic Design Studio*, M. McCullough, W.J. Mitchell and P. Purcell (eds.). Cambridge, MA: MIT Press, 137-148.
- Morelli, R.A., Walde, R.E., Akstin, E., and Schneider, C.W. (1991). "L-system Representation of Speciation in the Red Algal Genus *Dipterosiphonia* (Ceramiales, Rhodomelaceae)", *Journal of Theoretical Biology*, 149, 453-465.
- Motloch, J.L. (1991). *Introduction to Landscape Design*. New York: Van Nostrand Reinhold.
- Muller, J.C., Lagrange, J.P., and Weibel, R. (eds.). (1995). *GIS and Generalization: Methodology and Practice*. London: Taylor & Francis.
- Muller, J.C., and Mouwes, P.J. (1990). "Knowledge Acquisition and Representation for Rule-based Map Generalization: An Example from the Netherlands", *Proceedings of GIS/LIS 1990*. Falls Church, Virginia: American Society for Photogrammetry and Remote Sensing, 58-67.
- Navinchandra, D. (1993). "Observations on the Role of Artificial Intelligence Techniques in Geographic Information Processing", in *Expert Systems in Environmental Planning*, J.R. Wright, L.L. Wiggins, R. Jain, and T.J. Kim (eds.). New York: Springer-Verlag.
- NEPA, *National Environmental Policy Act* (United States). (1970).

- National Parks Service (United States). (2001). "The National Park System: Caring for the American Legacy". Available at: <http://www.nps.gov/legacy/organic-act.htm>
- Norvig, P. (1992). *Paradigms of Artificial Intelligence Programming*. San Mateo, California: Morgan Kaufmann Publishers. See also <http://www.norvig.com/>
- Nyerges, T.L., Montejano, R., Oshiro, C., and Dadswell, M. (1997). "Group-based Geographic Information Systems for Transportation Improvement Site Selection", *Transportation Research – C*, 5(6), 349-369.
- Okabe, A., Boots, B., Sugihara, K., and Chiu, S. (2000). *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. Chichester: John Wiley & Sons.
- Olmsted, Frederick Law. (1936). "As a Landscape Architect Sees Bermuda", pp89-91 in *Residence in Bermuda*, by Hervey Allen. No publisher specified.
- Openshaw, S. and Openshaw, C. (1997). *Artificial Intelligence in Geography*. New York: John Wiley & Sons.
- Oracle Corporation. (2002). Oracle Spatial User's Guide and Reference. Available at <http://otn.oracle.com/docs/products/spatial/content.html>
- O'Rourke, J. (1993). *Computational Geometry in C*. Cambridge, U.K.: Cambridge University Press.
- Owen, S. (1995). "Local Distinctiveness in Villages", *Town Planning Review*, 66(2), 143-161.
- Paepcke, A. (ed.) (1993). *Object-oriented Programming: The CLOS Perspective*. Cambridge, Massachusetts: The MIT Press.
- Paoluzzi, A., Sansoni, V., and Vincentino, M. (1993). "PLASM Functional Approach to Design: Representation of Geometry", in *CAAD Futures '93*, U. Flemming and S. Van Wyk (eds.), Amsterdam: Elsevier Science Publications, 127-142.
- Parks Canada (2000). "Introducing the National Parks of Canada". Available at: http://www.parkscanada.gc.ca/np/english/nptxt_e.htm
- Phillips-Watlington, C. (1996). *Bermuda's Botanical Wonderland*. London: MacMillan Education Ltd.
- Piazzalunga, U., and Fitzhorn, P. (1998). "Note on a Three-Dimensional Shape Grammar Interpreter", *Environment and Planning B: Planning and Design*, 25, 11-30.
- Porteous, D. (1982). "Approaches to Environmental Aesthetics", *Journal of Environmental Psychology*, 2, 53-66.
- Prusinkiewicz, P., Hammel, M., and Mech, R. (1997). "Visual Models of Morphogenesis: A Guide Tour", University of Calgary, Department of Computer Science. Available at <http://www.cpsc.ucalgary.ca/projects/bmv/vmm-deluxe/index.html>
- Prusinkiewicz, P., and Lindenmayer, A. (1990). *The Algorithmic Beauty of Plants*. New York: Springer-Verlag.
- Raban, J. (2001). "Battleground of the Eye", *The Atlantic Monthly*, 287(3), 40-52.

- Radinsky, L.B. (1987). *The Evolution of Vertebrate Design*. Chicago: University of Chicago Press.
- Raine, D.F. (1989). *Architecture Bermuda Style*. Bermuda: Pompano Publications.
- Raitz, K. and Van Dommelen, D. (1990). "Creating the Landscape Symbol Vocabulary for a Regional Image: The Case of the Kentucky Bluegrass", *Landscape Journal*, 9(2), 109-121.
- Rapoport, A. (1969). *House Form and Culture*. Englewood Cliffs, New Jersey: Prentice-Hall.
- Relph, E. (1973). "The Phenomenon of Place: An Investigation of the Experience and Identity of Places", Unpublished PhD Dissertation, Department of Geography, University of Toronto.
- Relph, E. (1976). *Place and Placelessness*. London: Pion.
- Riley, G. (1998). "CLIPS: A Tool for Building Expert Systems". Available at: <http://www.ghgcorp.com/clips/CLIPS.html>
- Roberts, S., Hall, G.B., and Calamai, P.H. (2000). "Analysing Forest Fragmentation Using Spatial Autocorrelation, Graphs and GIS", *International Journal of Geographical Information Science*, 14(2), 185-204
- Roberts, S.A., Hall, G.B. and Calamai, P.H. (2002). "Shape-based Properties of the Boundaries Between Landscape Feature Types", forthcoming, in P. Forer, A. G. O. Yeh and Jianbang He (eds.), *Spatial Data Handling*, New York: Springer Verlag.
- Rowlinson, B. (2000). Personal communication. Permanent Secretary, Ministry of the Environment, Bermuda.
- Royal Gazette (1958). "Lament Loss of View From 'Old Baldy'", *The Royal Gazette*, Hamilton, Bermuda, October 28th, 1958.
- Russ, T.A., MacGregor, R.M., Salemi, B., Price, K., and Nevatia, R. (1996). "VEIL: Combining semantic knowledge with image understanding", [*Postscript document from www.isi.edu*].
- Schmitt, G.N. (ed.). (1991). *Proceedings of CAAD Futures '91*. Wiesbaden: Vieweg, Bertelsmann Publishing Group International.
- Seebohm, T. (2000). Personal communication. Professor, School of Architecture, University of Waterloo, Waterloo, Ontario, Canada.
- Seebohm, T. and Wallace, W. (1998). "Rule-based Representation of Design in Architectural Practice", *Automation in Construction*, 8, 73-85.
- Sheppard, S. (1989). *Visual Simulation: A User's Guide for Architects, Engineers and Planners*. New York: Van Nostrand Reinhold.
- Silk, J. (1979). *Statistical Concepts in Geography*. London: George Allen & Unwin.
- Sardon, R.C., Palmer, J.F., and Felleman, J.P. (eds.). (1986). *Foundations for Visual Project Analysis*. New York: John Wiley and Sons.

- Southworth, F., Chin, S., and Cheng, P.D. (1990). "RTMAS: An Expert System for Real Time Monitoring and Analysis of Traffic During Evacuations", in *Expert Systems: Applications to Urban Planning*, Kim, Wiggins and Wright (eds.), 105-120.
- Spirn, A.W. (1998). *The Language of Landscape*. New Haven: Yale University Press. Also see <http://www.thewolfree.com>
- Steele, G. (1990). *Common Lisp the Language*, 2nd Edition. Woburn, Massachusetts: Digital Press. Also available at <http://www.cs.cmu.edu/afs/project/ai-repository/ai/html/cltl/cltl2.html>
- Stefik, M. (1995). *Introduction to Knowledge Systems*. San Francisco: Morgan Kaufmann Publishers.
- Sterrer, W. (1998). "How Many Species are There in Bermuda?", *Bulletin of Marine Science*, 62(3), 809-840.
- Stiny, G. (1975). *Pictorial and Formal Aspects of Shape and Shape Grammars: On Computer Generation of Aesthetic Objects*. Basel: Birkhauser.
- Stiny, G. (1977). "Ice-ray: A Note on the Generation of Chinese Lattice Designs", *Environment and Planning B: Planning and Design*, Vol. 4, 89-98.
- Stiny, G. (1980a). "Introduction to Shape and Shape Grammars", *Environment and Planning B: Planning and Design*, 7, 343-351.
- Stiny, G. (1980b). "Kindergarten Grammars: Designing with Froebel's Building Gifts", *Environment and Planning B: Planning and Design*, 7, 409-462.
- Stiny, G. (1993). "Emergence and Continuity in Shape Grammars", *CAAD Futures '93*, U. Flemming and S. Van Wyk (eds.), Amsterdam: Elsevier Science Publications, 37-54.
- Stiny, G. (1994). "Shape Rules: Closure, Continuity and Emergence", *Environment and Planning B: Planning and Design*, 21, 49-78.
- Stiny G. (1999). "Commentary: Shape", *Environment and Planning B: Planning and Design*, 26(1), 7-14.
- Stiny, G., and Gips, J. (1978). "An Evaluation of Palladian Plans", *Environment and Planning B: Planning and Design*, Vol. 5, 199-206.
- Stiny, G. and Mitchell, W.J. (1978a). "The Palladian Grammar", *Environment and Planning B: Planning and Design*, Vol. 5, 5-18.
- Stiny, G. and Mitchell, W.J. (1978b). "Counting Palladian Plans", *Environment and Planning B: Planning and Design*, Vol. 5, 189-198.
- Stiny, G. and Mitchell, W.J. (1980). "The Grammar of Paradise: On the Generation of Mughul Gardens", *Environment and Planning B: Planning and Design*, Vol. 7, 209-226.
- Struckmeyer, K. (1994). "Visual Resource Management", in *The Built Environment: Creative Inquiry into Design and Planning*, Bartuska and Young (eds.), 231-242.
- Tapia, M.A. (1992). "Chinese Lattice Designs and Parametric Shape Grammars", *The Visual Computer*, 9, 47-56.

- Tapia, M.A. (1999). "A Visual Implementation of a Shape Grammar System", *Environment and Planning B: Planning and Design*, 26(1), 59-74.
- Thomas, M.L.H. (1998). *Marine and Island Ecology of Bermuda*. Bermuda: Bermuda Aquarium, Museum and Zoo, The Bermuda Zoological Society, and the Friends of the Bermuda Aquarium.
- Tuan, Y.F. (1974). *Topophilia: A Study of Environmental Perception, Attitudes and Values*. Englewood Cliffs, New Jersey: Prentice Hall.
- White, R. and Engelen, G. (1997). "Cellular Automata as the Basis of Integrated Dynamic Regional Modelling", *Environment and Planning B: Planning and Design*, 24, 235-246.
- Wilensky, R. (1986). *Common LISPcraft*. New York City, New York: W.W. Norton & Company.
- Wolfram, S. (1994). *Cellular Automata and Complexity: Collected Papers*. Reading, Massachusetts: Addison-Wesley.
- Woodbury, R., and Griffith, E. (1993). "Layouts, Solids, Grammar Interpreters and Fire Stations", in *CAAD Futures '93*, U. Flemming and S. Van Wyk (eds.), Amsterdam: Elsevier Science Publications, 75-90.
- Woodbury, A.C. (1999). "Counting Eskimo Words for Snow: A Citizen's Guide". Available at <http://www.princeton.edu/~browning/snow.html>
- Wright, J.R., and Buehler, K.A. (1993). "Probabilistic Inferencing and Spatial Decision Support Systems", in *Expert Systems in Environmental Planning*, J.R. Wright, L.L. Wiggins, R. Jain, and T.J. Kim (eds.). New York: Springer-Verlag.
- Wright, J.R., Wiggins, L.L., Jain, R., and Kim, T.J. (1993). *Expert Systems in Environmental Planning*. New York: Springer-Verlag.