Forest Growth Simulation Using L-Systems

Submitted in partial fulfilment
of the requirements of the degree
Bachelor of Science (Honours)
of Rhodes University

Kim Randleff-Rasmussen

8 November 2004

Abstract

This project creates a realistic model of forest development and growth. It models the trees in diagram form using the wide-spread technique of L-Systems. The simulation of the forest growth takes into account a number of important issues in ecosystem development including the addition of new trees to the population, the removal of old trees and the changes that occur in existing trees due to interactions with others.

The interactions aim to model the competition that occurs between individual trees within a forest for the limited resources such as light, nutrients, water and CO_2. The interactions are simulated by checking for intersections between the circles which represent the space each tree takes up. When an intersection is discovered, the stronger tree dominates the weaker and the latter begins to die. Plants also die when they reach a maximum age.

Two particular tests were conducted on the system, one which checked for the realistic depiction of the results of competition and another which compared the output to that of another forest simulator (JABOWA III). The results of both tests were positive, prompting the conclusion that, though not realistic in minute detail, this simulator produces a realistic and thus viable depiction of forest growth.

Acknowledgements

I would like to thank my parents and my dog, without whom my sanity would long ago have given up the ghost.

I would like to thank my supervisors for their unwavering support and patience in tracking down rogue pointers and those annoying segmentation faults which plagued my coding.

Thanks must also go to my digsmates for their patience and uncomplaining sympathy as I bewailed the existence of trees in general.

Finally, thanks must, traditionally, go to Google and the marvellous results of its searches.

Contents

List of Figures

Chapter 1

Introduction

In all areas of computer graphics - from CGI movies to role-playing games - fields, mountainsides, jungles and forests are needed to extend the realism and thus complete the experience. For this reason, computer scientists have conducted much research into the modelling and rendering of realistic plants and trees. However, these natural environments are more than just a collection of trees and plants; rather, they involve a number of related yet individual organic organisms interacting with each other and thus affecting one another's shape and growth.

When two plants grow too near each other, they compete for the the limited resources in that limited space. These resources include sunlight, CO_2, water and various soil nutrients. In such a situation, the stronger plant will triumph and claim the bulk of the resources. The relative strength of a tree depends on inherited characteristics and thus differs from species to species. Some tree types prefer an abundance of sunlight and direct rain, others prefer shade, still others flourish better in circumstances of increased amounts of soil nutrients. Temperature and precipitation are also environmental factors which affect tree growth.

The aim of this project is to create a simulation of the growth and development of a forest. In order to accomplish this, the basic important aspects of this development need to be modelled and the model then needs to be tested to establish which areas require more detailed modelling. There are four basic aspects to focus on:

- plant placement
- introduction or adoption of new individuals (saplings)
- removal of old and dead plants
- interaction between existing population members.

In addition to the actual simulation considerations, decisions must be made regarding the manner/form of representation of the trees themselves. This includes deciding on how to represent tree growth, neighbour interaction and plant death. Finally (or, more accurately, firstly), a programming environment must be chosen which will provide a convenient vehicle for the development of the simulator.

The success of the simulator will be measured in terms of the realism of the generated output. This will be measured by comparing the trends therein to expected results observed from nature and established forest models.

1.1 Document Structure

Background in the fields of plant and ecosystem modelling is discussed in Chapter 2. Here, choices are explored and made in terms of tree representation and some of the other aspects of forest modelling. The choices made are discussed in more specific design detail in Chapter 3. Chapter 4 looks at the particular implementation decisions that were taken in this project. This section also explores the particular programming environment of this project, focussing on the interactions between the many parts of the program. Finally, Chapter 5 documents the results that have been achieved and compares these to the expected results, as well as discussing reasons for the variations.

Chapter 2

Background

Below is a summary of the techniques available in the field of plant modelling. These techniques are analysed in terms of their potential applicability to this project. Different approaches in the field of ecosystem modelling are also looked at, with focus on the various solutions to the four main focus areas of this project. In both of these cases, specific software solutions are also discussed.

The overview is structured as follows: first a summary of existing plant modelling techniques is given. This is followed by a more in-depth examination of L-Systems, the chosen technique in this project. Software products which provide L-System translation are then discussed. Next, some models of forest development are discussed including mention of some available software products.

2.1 Plant Modelling

2.1.1 Existing Techniques

Models abound for the modelling of realistic plants. Some of these focus on plant geometry while others model trees based on behavioural aspects.

Some oft-cited models in the former class include the mathematical model developed by de Reffye *et al.* [de Reffye et al. 1988] which formalises known botanical rules. Other models focus more on specific geometric phenomena such as [Borchert and Honda 1984] which models the tendency of a tree to increase the vigour of growth after the loss of a branch. Meyerowitz [Meyerowitz 1994] used the specific plant *Aribidopsis thaliana* to model genes and mutation as they influence plant growth.

A comprehensive study of plant architectures and models which reflects the range available can be found in [Hallé et al. 1978]. This book asserts that the thousands of plant species in the world can be

modelled using a mere twenty-three models. These models take into account individual species architecture and genetic ”growth program” as well as competition for space and nutrients in a single forest. It goes one step further, too, and models the architecture of forests as separate entities.

Borchert and Slade [Borchert and Slade 1981] and Janssen and Lindenmayer [Janssen and Lindenmayer 1987] also analysed the growth of a number of tree species in order to create models. An Algorithmic Botany paper contains a summary of current plant architecture modelling techniques and mention of a number of techniques which model plant organs and tissues, genetics and physiological processes [Prusinkiewicz 2004]. It maintains that the most mathematically explicit architecture modelling technique is L-Systems (discussed later).

The second class of plant models focuses on how behaviour and the environment influence the manner in which a plant grows. [Prusinkiewicz et al. 2000], for example, models the effect of gravity on the branching behaviour of certain trees. Another paper by one of those authors looks at pruning - forced and self-imposed - and how this affects branch growth [Prusinkiewicz et al. 1994]. Yet another paper, [Prusinkiewicz et al. 2001], uses positional information to model the growth and survival of a plant. This model uses such techniques as self-thinning and competition for space to model how tree growth is affected by environmental factors.

A model designed by Lück, Lück and Bakkali [Lück et al. 1990] suggests the use of a growth potential or ”vigor” value to be assigned to branches. This value dictates which of a number of competing branches will triumph. Honda *et al.* also focuses on branch interaction between trees [Honda et al. 1981]. The value they use to determine the ”winning” branch is flow rate which dictates the rate of bifurcation. Sorrensen-Cothern, Ford and Sprugel divide a single plant into a number of modules and model the interaction between these rather than the interaction between entire plants [Sorrensen-Cothern et al. 1993].

The above models focus on botanical phenomena in tree growth and model these in detail. For this reason, they are useful for models which require detailed tree growth including nutrient use and response to certain environmental factors. This simulation does not require this level of detail but rather a model which will illustrate a living plant which can be grown to the required size and shrunk when required. Such a model can be found in L-Systems which provide a simple framework for representing plants and can be extended to incorporate more detail as required.

2.1.2 L-Systems

2.1.2.1 Definitions and Explanations

In 1968, Hungarian biologist Aristid Lindenmayer developed a formalism for multicellular development [Lindenmayer 1968]. This formalism, subsequently named L-Systems (Lindenmayer Systems), is now widely used in computer graphics for the modelling of plants and other organic structures.

An L-System is a parallel rewriting system which uses a finite set of defined production rules and a finite symbol or module alphabet to supplant symbols within a string. Production rules can be applied sequentially or in parallel, to every module or according to contextual information. There has even been the definition of a type of L-System which allows for both - the rewriting of two or more symbols with a single production [Prusinkiewicz 1986].

Production rules which have the following structure:

`lc < pred > rc : cond` → `succ : prob`

where `lc` is the left-hand context, `rc` the right-hand context, `pred` the predecessor (the symbol to be replaced), `cond` a condition and `succ` the successor (the string to replace the predecessor). The rule is only run (i.e. the symbol replaced by the successor) if the condition evaluates to *true*. `prob` is a number which defines the probability of the rule being chosen. This is only found in stochastic L-Systems where more than one rule can exist for each symbol. Only the predecessor and the successor are compulsory.

For example, an L-System may comprise an axiom `F` and a replacement rule `F` → `F[<F][>F]F`. A first pass of the L-System will yield a new string `F[<F][>F]F` where the original `F` has been replaced. A second pass will yield `F[<F][>F]F[<F[<F][>F]F][>F[<F][>F]F]F[<F][>F]F` where each `F` in the second string is replaced by the successor of the rule.

2.1.2.2 Extensions

Many different types of L-Systems exist which allow for certain changes in order to more accurately model reality. Rozenberg, for example, defines table L-Systems [Rozenberg 1973] which allow for the choosing of a set of developmental rules according to certain environmental factors. Environmentally sensitive L-Systems [Prusinkiewicz et al. 1994] provide query symbols (`?E(parameters)`) which allow L-Systems to change as a result of local environmental conditions. Open L-Systems [Prusinkiewicz and Měch 1996] extend this functionality by including bilateral communication allowing the plant to return information to the environment. This extension allows the environment to be influenced by changes

in the plant model, enabling the realistic modelling of such aspects as nutrient uptake and supply.

Other extensions provide functionality for variation of a single model to be applied to a number of individuals. Parametric L-Systems [Prusinkiewicz et al. 1995] include the facility to involve parameters in a standard L-System which allows certain aspects of the model (such as angle size and segment length) to be changed dynamically rather than statically defined. Differential L-Systems (or dL-Systems) [Prusinkiewicz et al. 1993] extend these parametric L-Systems by incorporating continuous time flow. This facility allows smooth animation of plant development rather than the rendering of discrete development steps, as is the standard practice.

Perhaps the most widely used L-Systems in terms of individual variation are stochastic L-Systems [Eichhorst and Savitch 1980]. These use multiple productions for each symbol, the specific rule being chosen according to assigned probabilities and a random number generator. This functionality allows a single L-System to yield a number individual plants with the same basic shape but with noticeable differences.

Some extensions have been the addition of the functionality for L-Systems to model a specific element of plant development. Some of these have been the inclusion of a cut symbol (%) [Hanan 1992] to model self-pruning (see also [Prusinkiewicz et al. 1994]) and the development of erasing productions [Herman and Rozenberg 1975] to enable individual variation in a standard model.

A particular type of L-System has also been defined which allows a set of productions to be applied to a set of strings (individual plants) instead of to a single string. These L-Systems, called multi-set L-Systems [Prusinkiewicz and Lane 2002], mean that a dynamic plant population can be maintained and operated on with a single L-System. This technique was developed in order to manage spatial distributions of a plant population.

Much work has been done on the inclusion of genetic algorithms [McCormack 1993], mutation [Mock 1998] and evolution [Rodkaew et al. 2002] into L-Systems. This amalgamation of techniques allows for such applications as virtual landscaping and adaptation simulations. These simulations can be further aided by a technique known as extrusion in space-time [Hammel and Prusinkiewicz 1996] which extends a two-dimensional structure's development into a three-dimensional line or curve which represents the progress over time.

In order that L-System be useful to computer graphics artists, a functionality must exist to translate the final string from a set of symbols to a graphical image. The most common method of doing this is by

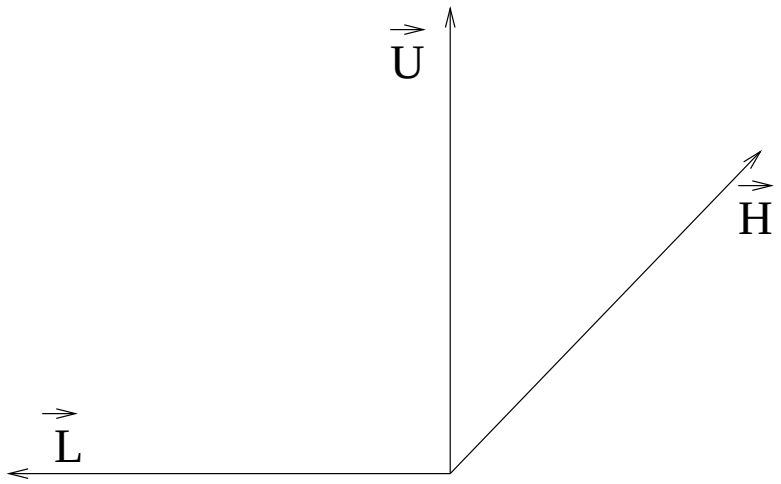

Figure 2.1: The three-dimensional space of the graphics turtle

using turtle graphics interpretation.

2.1.2.3 Graphical Interpretation

Turtle graphics interpretation of L-Systems [Szilard and Quinton 1979] involves scanning the generated L-System string (usually from left to right) and interpreting each symbol. The interpretations vary from implementation to implementation but all provide instructions to be passed to a LOGO-style turtle.

The turtle possesses a particular state which defined its appearance and position. Some symbols may change its appearance by changing the line width or the current colour. The most commonly used symbols change its orientation within the Cartesian plane by defining three vectors which indicate the turtle's heading and its direction up and to the left (see Figure 2.1).

Other symbols may cause the turtle to move forward according to its current heading by a certain number of unit steps. During this movement, the turtle may or may not draw a segment.

2.1.3 Software Solutions

One of the first L-System translation systems was developed in 1970 [Baker and Herman 1970] in order to simulate organic development for hypothesis testing. Since that time, L-System translators have been developed for a number of projects and a number of platforms. For example, CPFG [Prusinkiewicz 1993] takes a parameterised procedural model of a plant and outputs a geometric model. This product is used

in the modelling system L-Studio [Prusinkiewicz 1998] which provides a graphical user interface for the modelling of plants via L-Systems with comprehensive options for all aspects of L-Systems.

Many of these systems are available open source, such as Lparser [Lapré 1993], developed by Laurens Lapré. Another student developed product is L-Arbor developed by Marco Grubert [Grubert 1999]. This program offers simple animation and L-System modelling facilities and is an easy way to learn basic L-Systems.

The Institute of Forest Biometry and Informatics at the University of Guttingen is a special interest plant modelling group comprised of experts in a number of fields including forest science, biology and mathematics. Winfried Kurth is the computer graphics expert and he developed a modelling program called GROGRA (GROwth GRAmmar) [Kurth 1994].

GRAMPS [O'Donnell and Olson 1981] is a graphics language interpreter developed to model molecular structures and to animate their development. It is used in conjunction with an interactive vector display list processor and is focused on the provision of real-time scene manipulation facilities. A nice feature of this program is the facility to dynamically add new graphics manipulation commands through the use of macros.

2.2 Ecosystem Simulation

2.2.1 Plant Placement

There exists a number of methods by which the placement of plants in a forest can be determined. These methods can be considered to be local-to-global or global-to-local [Prusinkiewicz and Lane 2002]. The latter category contains methods which place individuals according to pre-defined and constantly updated density functions. As each plant is placed, the probability function is updated according to this new placement and the placement of the next plant will be affected.

Local-to-global plant placement methods simulate ecosystem interaction by focusing on each individual. Multi-set L-Systems are an example of this type of placement methodology. These L-Systems apply a set of production rules to a set of L-Systems strings (which denotes a plant population). This allows for the dynamic addition and removal of individual plants and the dynamic upkeep of the population as a whole. A simple method for placing plants within a modelled forest is random placement [Deussen et al. 1998]. Alternatively, a density map can be input and an error diffusion algorithm can then be applied to it.

In reality, trees produce seeds which are dispersed by various means. A percentage of these seeds then germinate and a percentage of the germinated seeds survive to grow into saplings.

2.2.2 Interaction and Competition

The complexity of the competition algorithm implemented depends on the level of reality that is required. A model whose primary aim is to predict forest growth of a real area would therefore need to model every aspect of competition. This would require calculating the exact inputs and products of every tree and how the amounts of these are affected by the surrounding tree. There is also the possibility of modelling, for example, soil nutrient levels and concentrations and how much of each nutrient is absorbed by each tree [Botkin et al. 1999]. Two trees growing close together would have to compete for the nutrient found within their shared soil.

One could then also model the transport of these nutrient through the tree to the various organs which require them. Insufficient quantities would result in the decline in health of the tree and probable eventual death.

Alternatively, these complex competitions could be simplified into a single domination algorithm. One particular such algorithm [Deussen et al. 1998] measures when two tree come into competition by measuring when they encroach on the area of one another. Once this occurs, a comparison takes place of each tree's strengths and the stronger tree dominates the weaker.

2.2.3 Testing

When a model is created, a decision must be made as to whether generality, realism or accuracy is the chief goal [Botkin 1993]. Generality refers to the applicability of the model to all observable cases, realism suggests that the model's projections follow the shapes of real, observed trajectories and accuracy infers that the model's projections are quantitatively similar to the real trajectories.

In order to assess the success of the model in achieving the desired canon, a number of standards and criteria must first be chosen.

- Variables of interest. What aspect of forest development is to be measured and compared with reality?
- Generality criterion. What range of circumstances should the model reflect?
- Determine if the goal is realism (qualitative) or accuracy (quantitative).

2.2.4 Software Products

JABOWA [Botkin et al. 1999] is a forest growth simulator developed by the Centre for the Study of the Environment in association with IBM. This simulator does not use L-Systems to model the trees but merely scaled images of the different species required. However, the simulator does model, in detail, many environmental and local factors which affect the growth rate and survival chances of the individual trees.

The Forest Ecosystem Management Simulation Group at the University of British Columbia have developed a number of forest simulation programs. Two of these are the FORCEE Individual Tree Model and the FORECAST Ecosystem Model [Kimmins 1998], both of which are detailed simulations of forest growth. These models contain provision for the numeric modelling of the following natural processes:

- photosynthesis
- nutrient cycling
- mortality
- biomass accumulation
- carbon allocation
- organic matter dynamics
- competition for light and nutrients
- various soil processes
- site quality change

They also allow for human probe effects such as pruning and harvesting and natural disasters like insect attacks and fire.

Both programs model a large number of aspects which will affect the forest growth. There are many characteristics of both the individual trees and the soil which are examined as well as attributes of the site itself.

Many of the tree characteristics which are modelled are physical production aspects such as photosynthetic rates, nutrient use and waste production. There are also the expected statistics of growth rate

and maximum height. There is also a variable called Natural Mortality, which holds information regarding the probability that an individual will die. It is activated primarily once the individual has come into competition, usually for light.

Another general ecosystem model - the GEM - has been developed by a collaboration between a number of environmental reseach groups in the United States [Fitz et al. 1996]. This model has been designed to model a variety of ecosystems on a variety of scales. It models the responses of organisms (algae and macrphytes) to changes in levels of nutrients, water and other environmental inputs. It is intended to prove the importance of the modelled factors on plant growth and ecosystem development.

2.3 Summary

A review of the available plant modelling techniques was undertaken. It was decided to use L-Systems due to their simplicity and extensibility. These were then described, with specific focus on the methods for translation to graphical images. A number of forest simulation approaches were looked at and metion made of a number of software products available.

Chapter 3

Design

The aim here is to create a diagrammatic representation of the interactions between a number of organic structures in a close environment. These interactions are modelled using a domination algorithm which stunts the growth of plants that encroach on the space of others. This same algorithm allows for the dynamic addition to and removal from the population of individual organisms. The plants themselves are modelled using the wide-spread technique of L-Systems which produce dynamic branching structures - perfect diagrams of plants for this purpose and ones that can be easily translated to realistic images should the need arise.

This chapter discusses the design considerations inherent in the creation of this diagrammatic representation. First, the technique of procedural modelling is discussed and reasons given for its choice. Secondly, the L-Systems used in this project are discussed in terms of their syntax and translation within the program. The algorithm used to model the forest growth is discussed in Section 3.3, including such issues as plant placement, domination modelling and addition and removal of trees.

3.1 Program Execution

At the basis of this program is the concept of using procedures to perform all of the complex calculation and geometry generation. Figure 3.1 shows the logical progression of the program, detailing the work done in each procedure and the outputs supplied to the next one. The initial procedure (`forest`) performs the forest growth simulation calculations including the placement of new plants, the interactions between existing plants and the removal of the old plants. The actual generation of the individual trees is delegated to another procedure (`lsystem`).

This procedure parses the L-System files (see Section 4.2) in order to translate them to a graphical image. However, no geometry is generated in this procedure. Rather, the final L-System string is gener-

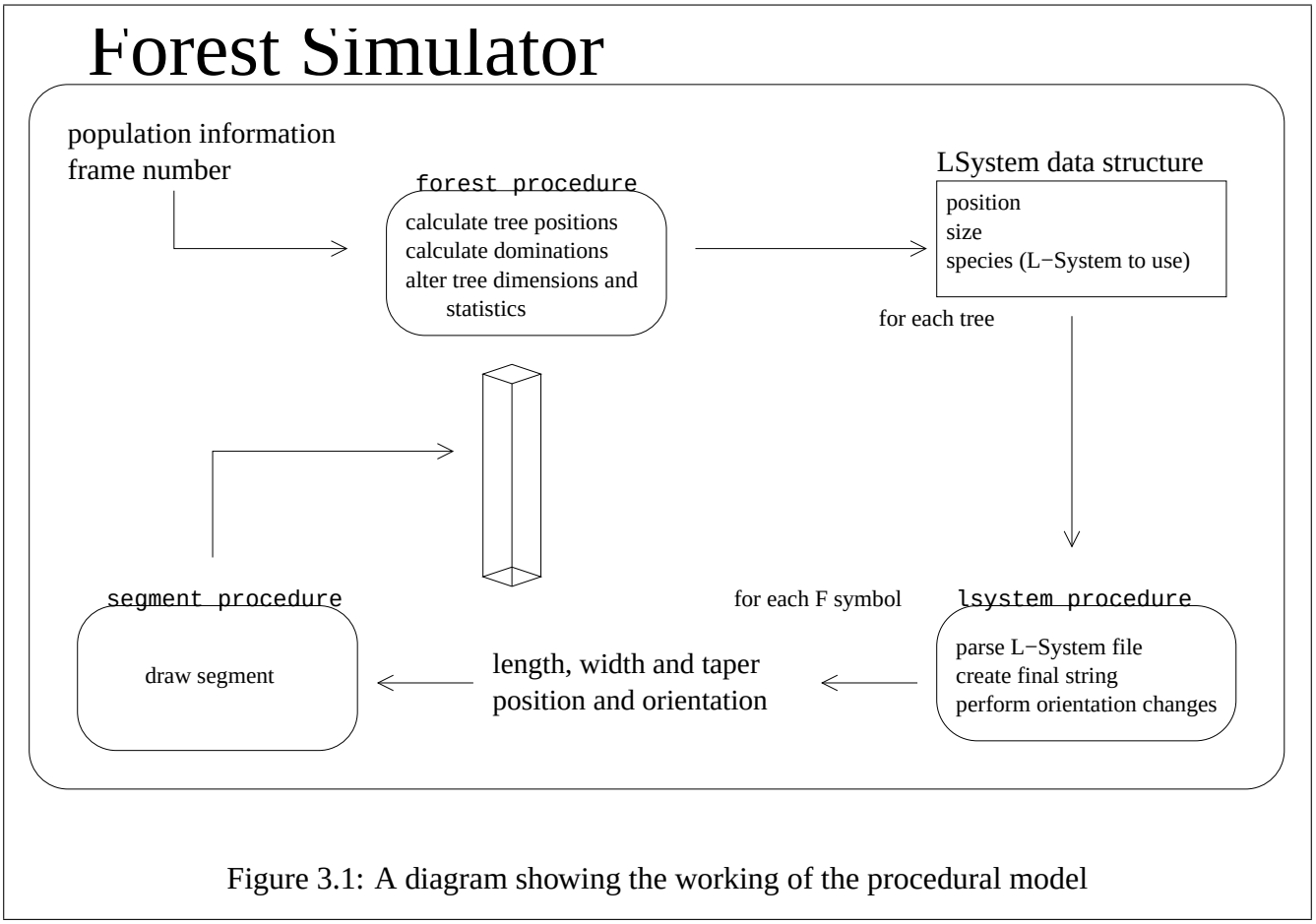

Figure 3.1: A diagram showing the working of the procedural model

ated according to the number of iterations required (which, in turn, is calculated according to the plant's age). This final string is then parsed and each symbol translated. The generation of the actual geometric segments is delegated to another procedure - `segment`.

3.2 Plants

3.2.1 L-System Representation

The `lsystem` procedure, which parses the L-System files and generates the final string, makes use of the LSystem class. This data structure has been designed to store all the information required about each L-System, namely:

- the final string to be translated
- the replacement rules available for each symbol
- the probabilities associated with each replacement rule

As can be seen in Figure 3.2, the final string is stored in a specially designed `alstr` data type which is designed to allow easy access to any symbol within the L-System string as well as the parameters associated with that symbol. This ease of access is mirrored by the presence of the `getModule` and `getParameters` operations in the LSystem class. These allow the `lsystem` procedure to access the symbols within the string without allowing access the `alstr` data structure directly.

3.3 Forest Simulation

There are a huge number of factors which influence the growth of a forest. [Botkin 1993] discusses the importance of finding "generalizations that are the simplest consistent with understanding". That is to say that one need not model every detail of a complex system but rather attempt to "determine the simplest conceptual basis consistent with the observation" of reality. For this reason, this project aims to simplify the complex and varied reality of forest growth into a general model.

3.3.1 Plant Placement

As discussed in Section 2.2.1, realistic modelling of plant placement would require complex modelling of seed production, dispersal and germination. For the purposes of this model, plants are placed randomly within the forest area. This has the effect of placing plants in the clearer areas, which is a simplified model of seeds being randomly dispersed and germinating in areas where they recieve sufficient light and nutrients.

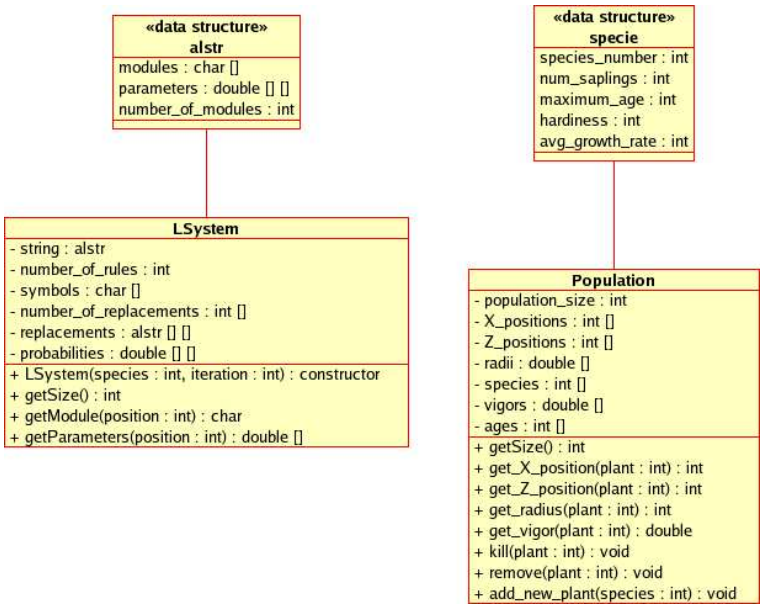

Figure 3.2: A UML representation of the LSystem and Population classes

3.3.2 Competition and Domination

In reality, the growth rates of trees in a forest are affected by a number of environmental factors including the availability of sunlight, water and nutrients. These availabilities are influenced by the surrounding trees - when two trees grow too close together, they must compete for the limited amounts of light, water and nutrients in their shared space. Ultimately, one tree will triumph over another.

This simulation does not model the competition between trees for every element necessary for their survival. Rather, the competition is simplified into a domination algorithm which states that when two trees come into competition with one another, one will triumph resulting in the decrease in growth rate of the other. This algorithm assumes that intersection between two trees results in a competition for resources which one individual will win.

For the purposes of the simulation, plants are viewed as circles on the viewing plane. These circles have dynamic radii which change according to the growth rate of the plant in question which is calculated as a function of the plant's health. When two circles intersect, the plants are said to have come into competition with one another. The vigour values of both individuals are compared and the stronger plant dominates the weaker. This means that the weaker plant's health is decreased, resulting either in stunted growth or even death. Thus, the circles can be said to represent the "footprint" of the plant in the forest.

The radii of the circles also form the basis for the calculation of the age of the plant. For this project, the age is an integer value ranging from 0 to the maximum size of the plant's species. This age value is used to determine how many iterations of the L-System rules must be completed to yield the final L-System string. An age value of 0 does not necessarily result in an empty mesh. Rather the axiom of the L-System is translated and, should it contain any `F` symbols, drawn accordingly.

3.4 Testing

Using the framework set forth in Section 2.2.3, the following must be taken into consideration when testing the final program.

- Variables of interest. This simulation is intended to model forest growth. As such, the forest size (number of plants present) is the aspect which will be compared to reality.
- Generality criterion. This model should be very general. This means that it should produce a universal growth model which could be refined for a more specific circumstance.
- The goal here is realism. The results of the model should be qualitatively similar to the observed phenomenon.

3.5 Summary

This chapter discusses the various design decisions that were taken in creating this simulation. The workings of the procedural models are discussed as are the various ways in which L-Systems are stored and read. In terms of the forest simulation, plants are randomly placed about the scene and grow according to defined growth rates. These growth rates are affected by the competition a plant receives from other plants in the scene which is modelled by checking for intersections between circles which represent the plants.

Chapter 4

Implementation

4.1 RhoVeR

The RhoVeR system is a centrally archived system which can be accessed and developed concurrently by a number of people. It is accessed via a CVS (Central Versions System) which allows for every developer to have a local copy of the code which can be added to the central version with the relevant updates. This technique means that the central system is always stable and can thus be redownloaded should the local copy be corrupted.

The source code used in this project is the `harrier` set which is the latest version of the source code. It consists of a number of classes which can be classified into five basic groups:

- Basic geometry tools
- Mesh utilities
- Movement tools
- Procedural models
- Scene viewing classes

For this project, the procedural modelling class (`ProceduralModel`) was used.

This class provides a structure for the definition of new procedural models. Each procedure is supplied with the Camera object, the Transformation matrix from the calling procedure and a long integer key. In addition to this, each procedure can define additional parameters as needed. Each procedure returns a pointer to a Mesh object.

The geometric mesh produced by each procedure is produced according to its local coordinate system which is then transposed to the correct position in the global coordinate system using the Transformation matrix supplied. The long integer key can be used to seed a pseudo-random number sequence. It is useful to use the same number to seed the sequence so that the exact same scene can be generated provided the key is known. This pseudo-random number sequence is used to determine the x and z coordinates of the plant positions, the number of saplings per species to add each frame and the vigour values of the individual trees.

The original `forest` procedure is called from the `ObjectViewer` class which is the main executable class within the `harrier` source code. This class provides definition of all of the rendering aspects of the project including camera transformations, colour, rendering engine and timing information. It is also within this class that the initial forest information is created.

4.2 L-Systems

4.2.1 `.sys` Files

An L-System is read from a `.sys` file which has the following structure:

`LSYSTEM`	file header
`r`	an integer denoting the number of replacement rules
`A`	axiom (or L-System start string)
`R`	replacement rules with the structure `A R p` (`A` is the symbol to replace, `R` the string to replace it with and `p` is a double value which signifies the probability of the rule being fired)

An example of one of these files can be seen below and the graphical interpretation of this can be seen in Figure 5.2:

```
LSYSTEM
1
F
F F [%>F]F[<&F][<&F][%F] 1
```

4.2.2 L-System Parser

The `.sys` files are read by the L-System parser which uses the information contained therein to create an LSystem data structure (see Section 3.2.1 above). The axiom string is converted to an `alstr` and stored

in the `string` variable. The rules are then read one at a time and stored in the `replacements` array. This array is a two-dimensional array to allow for the possibility of more than one rule being present for each symbol. The `symbols` array holds which symbols have an associated replacement rule and the `number_of_replacements` array contains the number of rules associated with each of these symbols.

Once the rules are stored, the parser can then generate the required L-System string. This is done by iteratively applying the replacement rules to the current string for the number of iterations provided. At each step, the new string is then stored in the `string` variable.

In order to translate the L-System string into a graphical image, the L-System alphebet needs to be defined. THe table below shows the symbol interpretations used in this program:

Geometry Definition Symbols	
`F(l,w,t)`	Draw a tapered rectangular segment of length `l`, base width `w` and tapered tip width `t`.
Orientation Change Symbols	
`<(`θ`)`	Turn right by angle θ around the z axis.
`>(`θ`)`	Turn left by angle θ around the z axis.
`&(`θ`)`	Turn (roll) right by angle θ around the y axis.
`@(`θ`)`	Turn (roll) left by angle θ around the y axis.
`%(`θ`)`	Turn right (pitch down) by angle θ around the x axis.
`#(`θ`)`	Turn left (pitch up) by angle θ around the x axis.
`|`	Turn 180 degrees around the z axis.
Positional Information Symbols	
`[..]`	Push the current state of the turtle (position, orientation and drawing attributes) onto a pushdown stack.Branch by pushing (and later popping) the current Transformation matrix to (from) a pushdown stack.

What this means is that the parser traverses the length of the final L-System string and takes action according to what symbol it encounters. The symbol is read as are its parameters in the event that they exist. If they do not, default values are applied to the relevant variables.

- Geometry Definition Symbols

Upon encountering an `F` symbol, the parser defines three variables which represent the length, width and taper width of the segment to be drawn. The default values are 10.0, 2.0 and 1.0 respectively. These

values are then sent to the `segment` procedure which will generate a segment with the appropriate dimensions.

The position and orientation of the segment to be drawn are determined by the current Transformation matrix. This is updated according to which orientation change symbols have been encountered previously in the string. This Transformation matrix is then sent to the `segment` procedure for alignment with the object coordinate system of the segment (see Section 4.1 for details).

- Orientation Change Symbols

Each of these symbols has one parameter which may be associated with it. On encountering one of these symbols, the parser modifies the current Transformation matrix by applying a rotation transformation to it. The angle of rotation is defined by the parameter of the symbol (the default value is 45 degrees or $\Pi/4.0$ radians) and the axis about which to rotate is defined by the symbol itself (see the alphabet table above).

- Positional Information Symbols

These symbol represent a branch in the L-System. An open bracket symbol (`[`) means that the current position and orientation (i.e. the Transformation matrix) is to be remembered while the symbols within the brackets are interpreted. Upon encoutering a close bracket symbol (`]`), the last remembered Transformation matrix must be restored.

This parser maintains a `count` value which counts the number of levels of branching which have been encountered. An array of Transformation matrices is maintained along with this and thus the relevant matrix can be saved or restored in the correct place.

4.3 Summary

This chapter describes the RhoVeR programming environment in which this simulator was created. There is also a description of the structure of the `.sys` files which are used to house the L-Systems to be used within the program. This discussion includes the syntax of the file and the L-System strings themselves. Finally, the L-System parser is examined and various implementation isuues are discusses in terms of translating L-System strings into graphical images.

Chapter 5

Results

5.1 L-Systems

L-Systems are parsed and translated as discussed in Section 4.2. Figure 5.1 shows an example of an L-System image where the originating string is (ignore '-' symbols in this text):

```
F[>F][<F]F[>F[>F][<F]F][<F[>F][<F]F]F[>F][<F]F[>F[>F][<F]F[>F[>F][-
<F]F][<F[>F][<F]F]F[>F][<F]F][<F[>F][<F]F[>F[>F][<F]F][<F[>F][<F]F-
]F[>F][<F]F]F[>F][<F]F[>F[>F][<F]F][<F[>F][<F]F]F[>F][<F]F.
```

Figure 5.2 shows another L-System example, illustrating the ability of the simulator to create three-dimensional images as well as two-dimensional. The L-System file describing this figure can be seen in Section 4.2.

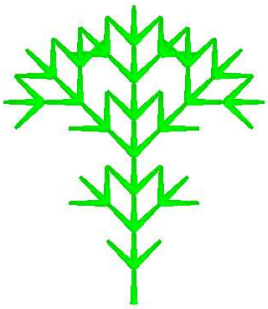

Figure 5.1: An example of an L-System tree

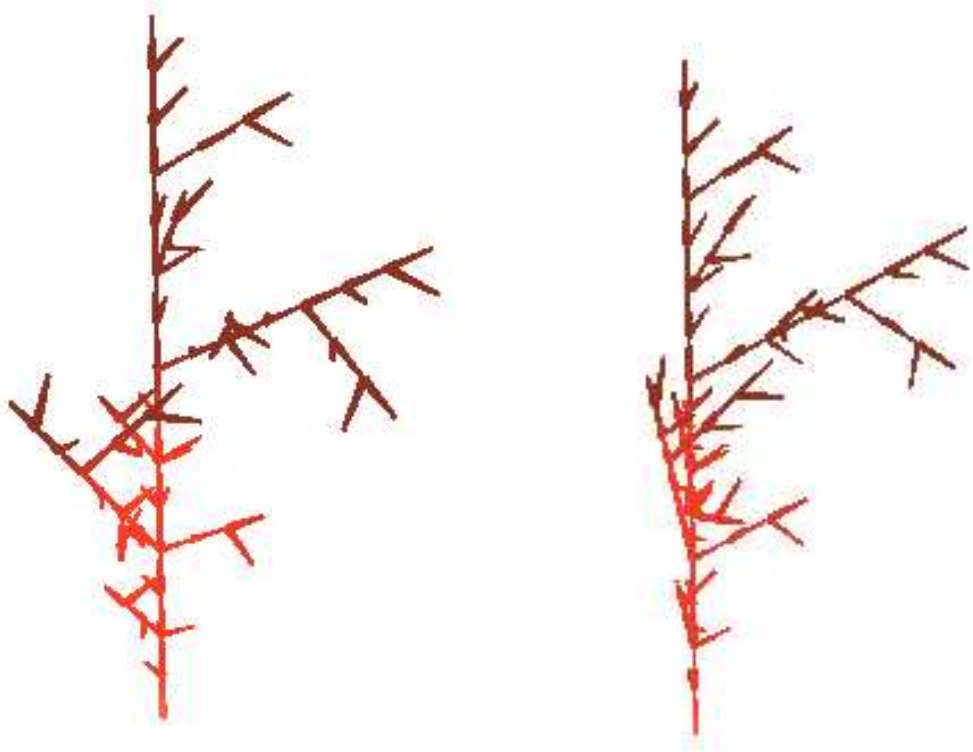

Figure 5.2: Two views of an example of a three-dimensional L-System tree

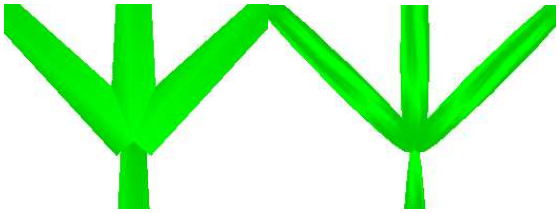

Figure 5.3: Two versions of the same joint showing the difference between the generated cube (left) and the same cube after smoothing (right)

Generalised mesh subdivision techniques are introduced in [Maierhofer 2002] and allow for the smoothing of meshes using a combination of a number of common subdivision methods (such as Catmull-Clark, Doo-Sabin and $\sqrt{3}$-Subdivision). Implemented in this project are some of these techniques in order to create more realistic, smooth trees. Figure 5.3 shows a section of an L-System tree before application of these techniques and after smoothing.

5.2 Forest Simulation

5.2.1 Plant Placement

As has been mentioned in Section 3.3.1, the plants in this scene are randomly placed. This is achieved by generating a random number in the range -100 to 200 for the x-coordinate and another in the range 100 to 900 for the z-coordinate. These ranges represent the area of the forest in coordinate terms.

[Prusinkiewicz and Lane 2002] suggests that random placement of the plants within the scene leads to a uniform distribution which is unrealistic. They apply a clustering algorithm in order to counter the uniformity of the plant distribution. No clustering algorithms are employed in this project; however, as can be seen in Figure 5.5, uniform distribution is not an issue in this implementation.

5.2.2 Competition and Domination

A simple test, suggested by [Botkin 1993], can be conducted to test the realism of the simulator's model of competition. This test involves allowing a large, sturdy tree to exist in the area and removing it after a number of cycles. If the simulation models competition realistically, the other trees in the area should grow at extremely slow rates, not grow at all or even die while the large, sturdy tree survives. Once the larger tree is removed, the smaller, dominated trees and others that join the population should grow at

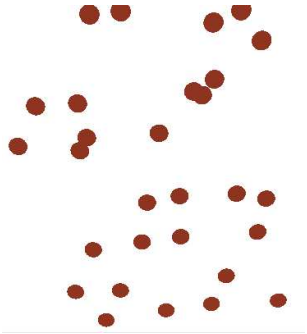

Figure 5.4: Plant distribution using random numbers

Figure 5.5: Plant distribution showing the L-Systems

increased rates.

If, as a measure of forest growth, the number of plants present per year/frame is taken, then, in general, the number of plants should be less while the large, sturdy tree exists. Once the large, sturdy tree is removed, the number of trees should increase until it is approximately equal to the number of trees present without the large, sturdy tree. The resulting chart would be expected to have a shape close to that of Figure 5.6.

Figure 5.7 shows the numbers generated whilst performing the above test on the simulator. As can be seen, the number of trees present in the forest is much less than in the ”normal” simulation while the large, sturdy tree exists. At frame 100 (highlighted by a thick black line in Figure 5.7), the large, sturdy tree is removed and forest growth increases until the number of trees present is approximately equal to that of an ordinary simulation. Growth then stabilises to keep the number of trees approximately level.

The chart shows that the shape of the graph follows the expected shape. This means that the model qualitatively realistically models forest growth which is in keeping with the testing criteria set down in Section 3.4.

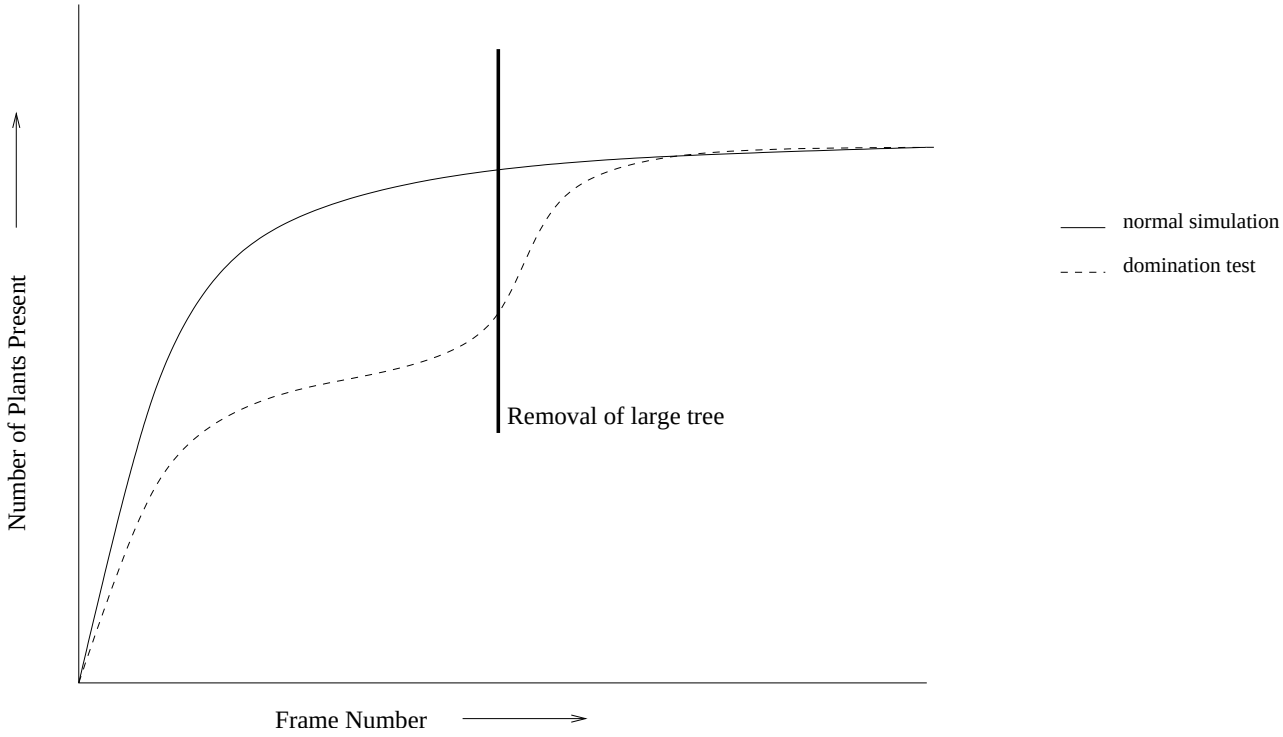

Figure 5.6: The expected shape of the domination test chart

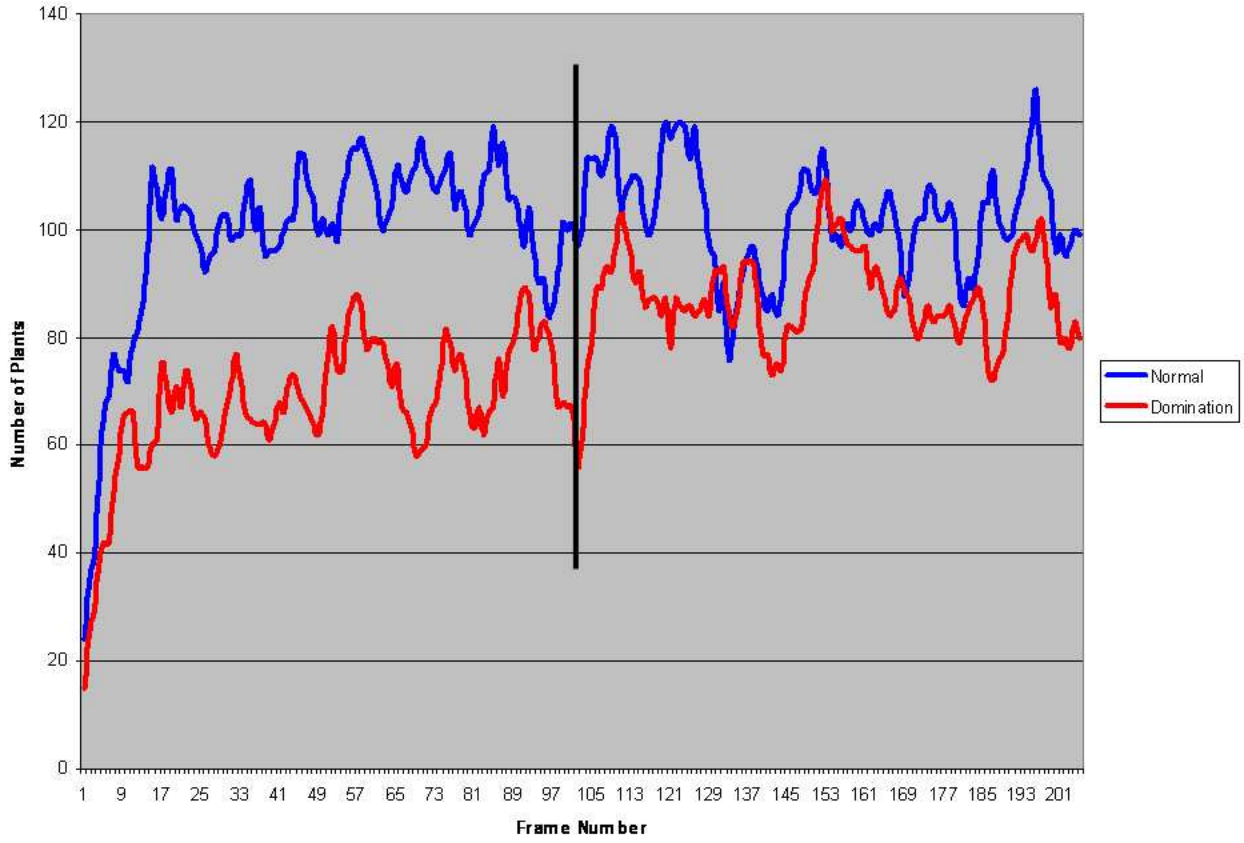

Figure 5.7: Progression of an ordinary simulation versus a domination test.

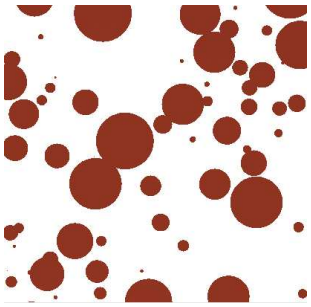

Figure 5.8: A top-down view of the forest simulation. The circles pictured represent the footprints of the plants in the forest

Figure 5.9: Another view of the forest simulation showing that the circles exist on one plane

5.2.3 Full Simulation

As mentioned in Section 3.3, the plants are represented by their footprints in the forest growth simulator. Figures 5.8 and 5.9 show moments in the simulation showing the various plant footprints. When two footprints overlap, the plants' vigour values are compared and one plant is dominated - its growth rate is decreased.

Figure 5.10 shows snapshots of the forest simulation with the L-Systems. The varying sizes of the individual trees can be seen as well as the random placement.

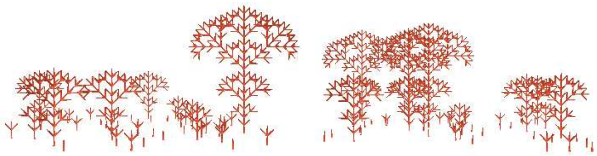

Figure 5.10: Snapshots of the forest simulation showing the L-Systems

5.2.3.1 Comparison

In order to test the quantitative realism of the produced simulator, the output of a detailed forest simulator, JABOWA III [Botkin et al. 1999] was compared to the output of the forest simulator created. If the outputs of both were seen to be similar, it could be safely concluded that this simulation adequately modelled forest development.

First, a number of species was chosen for each model (in this case 6). Then, the various aspects of each species were assigned values that were approximately equal. **??** contains the details of the species that were used in both models. Due to the more complex nature of the JABOWA simulator, there were a large number of characteristics which did not have corresponding variables in the Population class. For this reason, those extra variables were made constant for all species.

Next, site specific information was examined in both cases. This involved making the climate stay constant in the JABOWA simulation and setting the forest areas in both models to corresponding sizes. Finally, in order to gain a more accurate picture of the comparisons, both simulations were run a number of times and the outputs averaged.

Figure 5.11 shows how the shape of the graph generated by the number of plants in the created simulator was similar to that of the detailed simulator (JABOWA III). The discrepancies that can be seen (during the first 90 frames) can be attributed to the fact that the JABOWA simulation models various environmental concerns which allow for a larger number of plants during the first years of a forest's existence.

The created simulator reaches a stable population level early in its existence and maintains that level. The JABOWA simulator takes longer to achieve the stable level as the different species fight for supremacy. The fact that both the detailed simulator and the created simulator reach a stable population and maintain that level means that the created simulator does realistically model forest development.

5.3 Summary

This section reviews the results achieved by the forest simulator. Images are provided to illustrate the method employed in the simulation and the final animation. Chartss are also provided to illustrate the results of the tests that were conducted on the simulator. These tests are a simple domination test to establish whether competition is modelled realistically and a more complex test involving comparison with another forest simulator. The results of both tests were qualitative similarities between reality (in the first test), JABOWA (in the second) and the results of the simulator.

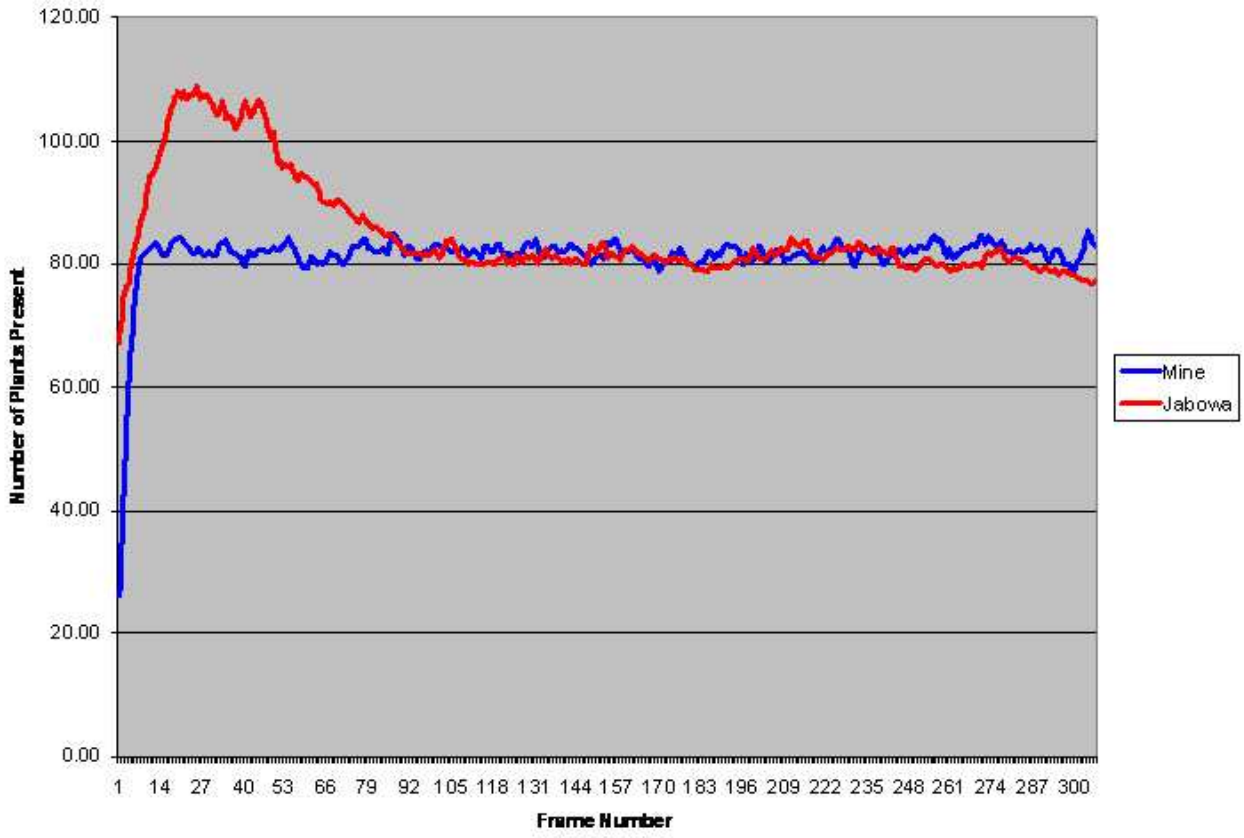

Figure 5.11: Progression of a JABOWA III simulation versus this forest simulator

Chapter 6

Conclusion

This project creates a diagrammatic representation to simulate the development and growth of a forest. The program shows the addition of new trees to the forest population and the removal of old ones. It also models the interactions which occur between the trees, resulting in the domination of one individual by another and the subsequent change in growth rate and ultimate death of the weaker specimen.

This project simplifies the complex nature of natural interactions and thus does not model forest growth in minute detail. Rather, the significant aspects of forest growth are investigated and simulated, assuming that the smaller details will affect the growth only in very small ways. This approach has meant that the simulator produced shows a realistically growing forest but one that fails more rigorous testing.

The testing conducted was twofold: first, a simple domination test was conducted which tests the realism of the competition modelling. The outcome of this test was successful, proving that the simulator models interactions between trees and the subsequent effects on growth rate realistically.

The second test was more complex and involved comparing the output of the simulator with that of another, established forest simulator. Again, the test was successful, illustrating the viability of the simulator as a depiction of reality. Certain differences were noted, however, and the negative results of the broader model could be plainly seen during a longer simulation.

The simulator produced is a viable one which depicts realistic forest growth to a degree that is acceptable for computer graphics needs. While much detail could be added to the simulator in order to correct some of the more general modelling aspects, the simulator as it stands could be employed in a computer game or movie. The simulator, however, is only diagrammatic and thus more work would have to be done on the realistic depiction of the individual trees.

References

[Baker and Herman 1970] Baker, R. and G. Herman (1970). Celia - a cellular linear iterative array simulator. In *Proceedings of the Fourth Conference on Applications of Simulation* (1970), pp. 64–73.

[Borchert and Honda 1984] Borchert, R. and H. Honda (1984). Control of development in the bifurcating branch system of tabebuia rosea: A computer simulation. *Botanical Gazette 145*, 2 (1984), pp. 184–195.

[Borchert and Slade 1981] Borchert, R. and N. Slade (1981). Bifurcation ratios and the adaptive geometry of trees. *Botanical Gazette 142*, 3 (1981), pp. 394–401.

[Botkin 1993] Botkin, D. (1993). *Forest Dynamics: An Ecological Model*. Oxford University Press, UK, 1993.

[Botkin et al. 1999] Botkin, D., D. Woodby and J. Bergengren (1999). Jabowa: A model of forest growth. Tech. rep., The Center for the Study of the Environment, New York, 1999.

[Deussen et al. 1998] Deussen, O., P. Hanrahan, B. Lintermann, R. Měch, M. M. Pharr and P. Prusinkiewicz (1998). Realistic modeling and rendering of plant ecosystems. In *SIGGRAPH 1998 Conference Proceedings* (1998), ACM Press.

[Eichhorst and Savitch 1980] Eichhorst, W. and J. Savitch (1980). Growth functions of stochastic lindenmayer systems. *Information and Control 45*, 3 (1980), pp. 217–228.

[Fitz et al. 1996] Fitz, H., E. DeBellevue, R. Costanza, R. Bournans, T. Maxwell, L. Wainger and F. Sklar (1996). Development of a general ecosystem model for a range of scales and ecosystems. *Ecological Modelling 88*, 1/3 (1996), pp. 263–295.

[Grubert 1999] Grubert, M. (1999). L-arbor. Master's thesis, Technical University of Berlin, Germany, 1999.

[Hallé et al. 1978] Hallé, F., R. Oldeman and P. Tomlinson (1978). *Tropical trees and forests*. Springer-Verlag, Heidelberg, New York, 1978.

[Hammel and Prusinkiewicz 1996] Hammel, M. and P. Prusinkiewicz (1996). Visualization of developmental processes by extrusion in space time. In *Proceeding of Graphics Interface '96* (1996), pp. 246–258.

[Hanan 1992] Hanan, J. (1992). *Parametric L-systems and their application to the modelling and visualization of plants*. PhD thesis, University of Regina, Canada, 1992.

[Herman and Rozenberg 1975] Herman, G. and G. Rozenberg (1975). *Developmental systems and languages*. North-Holland, Amsterdam, 1975.

[Honda et al. 1981] Honda, H., P. Tomlinson and J. Fisher (1981). Computer simulation of branch interaction and regulation by unequal flow rates in botanical trees. *American Journal of Botany*, 68 (1981), pp. 569–585.

[Janssen and Lindenmayer 1987] Janssen, J. and A. Lindenmayer (1987). Models for the control of branch positions and flowering sequences of capitula in mycelis muralis (l.) dumont (compositae). *New Phytologist 105*, 2 (1987), pp. 191–220.

[Kimmins 1998] Kimmins, J. (1998). Forest ecosystem simulation and forest navigator (forecast). Tech. rep., Forest Ecosystem Management Simulation Group, University of British Columbia, 1998.

[Kurth 1994] Kurth, W. (1994). *GROGRA*. PhD thesis, Institute of Forest Biometry and Informatics, Faculty of Forest Sciences and Forest Ecology at the University of Göttingen, 1994.

[Lapré 1993] Lapré, L. (1993). Lparser.

[Lindenmayer 1968] Lindenmayer, A. (1968). Mathematical models for cellular interaction in development. *Journal of Theoretical Biology* (1968).

[Lück et al. 1990] Lück, J., H. Lück and M. Bakkali (1990). A comprehensive model for acrotonic, mesotonic, and basitonic branching in plants. *Acta Biotheoretica 38* (1990), pp. 257–288.

[Maierhofer 2002] Maierhofer, S. (2002). *Rule-Based Mesh Growing and Generalised Subdivision Meshes*. PhD thesis, Vienna University of Technology, Austria, 2002.

[McCormack 1993] McCormack, J. (1993). Interactive evolution of l-system grammars for computer graphics modelling. In *Complex Systems: from Biology to Computation*, D. Green and T. Bossomaier, Eds. ISO Press, Amsterdam, 1993.

[Meyerowitz 1994] Meyerowitz, E. (1994). The genetics of flower development. *Scientific American 271*, 5 (1994), pp. 56–65.

[Mock 1998] Mock, K. (1998). Wildwood. In *International Conference on Evolutionary Computing (ICEC '98)* (1998).

[O'Donnell and Olson 1981] O'Donnell, T. and A. Olson (1981). Gramps: A graphics language interpreter for real-time, interactive, three-dimensional picture editing and animation. *Computer Graphics 15*, 3 (1981), pp. 133–142.

[Prusinkiewicz 1986] Prusinkiewicz, P. (1986). Graphical applications of l-systems. In *Proceedings of Graphics Interface '86 - Vision Interface '86* (1986), pp. 247–253.

[Prusinkiewicz 1993] Prusinkiewicz, P. (1993). Cpfg version 2.0. Tech. rep., Virtual Laboratory, University of Calgary, 1993.

[Prusinkiewicz 1998] Prusinkiewicz, P. (1998). L-studio. Tech. rep., University of Calgary Computer Science, 1998.

[Prusinkiewicz 2004] Prusinkiewicz, P. (2004). Modeling plant growth and development. *Current Opinion in Plant Biology 7*, 1 (2004), pp. 79–83.

[Prusinkiewicz et al. 2000] Prusinkiewicz, P., C. Jirasek and B. Moulia (2000). Integrating biomĕchanics into developmental plant models expressed using l-systems. In *Plant BioMĕchanics 2000*, H. Spatz and T. Speck, Eds. Georg Thieme Verlag, Stuttgart, 2000.

[Prusinkiewicz and Lane 2002] Prusinkiewicz, P. and B. Lane (2002). Generating spatial distributions for multilevel models of plant communities. In *Proceedings of Graphics Interface* (2002), pp. 69–80.

[Prusinkiewicz et al. 2001] Prusinkiewicz, P., B. Lane, L. Muendermann and R. Karwowski (2001). The use of positional information in the modeling of plants. In *SIGGRAPH 2001 Conference Proceedings* (2001), ACM Press, pp. 289–300.

[Prusinkiewicz et al. 1993] Prusinkiewicz, P., E. Mjolsness and M. Hammel (1993). Animation of plant development. In *SIGGRAPH 1993 Conference Proceedings* (1993), ACM Press.

[Prusinkiewicz and Měch 1996] Prusinkiewicz, P. and R. Měch (1996). Visual models of plants interacting with their environment. In *SIGGRAPH 1996 Conference Proceedings* (1996), ACM Press, pp. 397–410.

[Prusinkiewicz et al. 1995] Prusinkiewicz, P., R. Měch and M. Hammel (1995). The artificial life of plants. In *Artificial Life for Graphics, Animation, and Virtual Reality, v7 of SIGGRAPH 1995 Course Notes* (1995), ACM Press, pp. 1.1–1.38.

[Prusinkiewicz et al. 1994] Prusinkiewicz, P., R. Měch and M. James (1994). Synthetic topiary. In *SIGGRAPH 1994 Conference Proceedings* (1994), ACM Press.

[de Reffye et al. 1988] Reffye, P. de, C. Edelin, J. Francon, M. Jaeger and C. Peuch (1988). Plant models faithful to botanical structure and development. *Computer Graphics 22*, 4 (1988), pp. 151–158.

[Rodkaew et al. 2002] Rodkaew, Y., C. Lursinsap, T. Fujimoto, S. Suripant, P. Chongstitvatana and N. Chiba (2002). Modelling leaf shapes using l-systems and genetic algorithms. In *International Conference NICOGRAPH* (April, Japan, 2002), April, Japan.

[Rozenberg 1973] Rozenberg, G. (1973). T0l-systems and languages. *Information and Control 23*, 4 (1973), pp. 357–381.

[Sorrensen-Cothern et al. 1993] Sorrensen-Cothern, K., E. Ford and D. Sprugel (1993). A model of competition incorporating plasticity through modular foliage and crown development. *Ecological Monographs 63*, 3 (1993), pp. 277–304.

[Szilard and Quinton 1979] Szilard, A. and R. Quinton (1979). An interpretation for d0l systems by computer graphics. *The Science Terrapin 4* (1979), pp. 8–13.