The PLANI Plant Animation Framework

A Thesis

Submitted to the Faculty of Graduate Studies and Research

In Partial Fulfillment of the Requirements

For the Degree of

Master of Science

In

Computer Science

University of Regina

By

Tina Louise Mary Sturgeon

Regina, Saskatchewan

January 2020

Copyright © 2020: Tina L.M. Sturgeon (Derzaph)
Tina Louise Mary Sturgeon, candidate for the degree of Master of Science in Computer Science, has presented a thesis titled, The PLANI Plant Animation Framework, in an oral examination held on December 20, 2019. The following committee members have found the thesis acceptable in form and content, and that the candidate demonstrated satisfactory knowledge of the subject material.

External Examiner: Prof. Gerald Saul, Department of Film
Supervisor: Dr. Howard Hamilton, Department of Computer Science
Committee Member: Dr. Mary Vetter, Department of Biology
Committee Member: Dr. Boting Yang, Department of Computer Science
Committee Member: *Dr. Xue Dong Yang, Department of Computer Science
Chair of Defense: Dr. Irfan Al-Anbagi, Faculty of Engineering & Applied Science

*Not present at defense
Abstract

This thesis presents the unique PLANI plant animation framework for animation using dynamic plant models along with three novel demonstrations of the use of the framework. A dynamic plant model is a computer-based representation of a plant that can be used to simulate or animate natural behaviors, such as moving to wind, responding to light, and responding to nutrient availability. The dynamic plant models used in the thesis are based on interdisciplinary research in plant biology, animation in computer graphics, and ontology. The PLANI framework guides the specification of both design time and runtime processes for plant animations. PLANI also provides a systematic approach to incorporating algorithms to support the simulation based on dynamic plant models. The framework utilizes four domains representing plant form, plant function, plant environmental effects, and the virtual world in which the generated plant resides. PLANI also provides plant specifications for animators who are designing plants for film and real-time animations, such as gaming. PLANI enables clear communication between experts in biology and computer science using well-defined terminology from plant ontologies. The use of plant ontologies also ensures that computer algorithms align with biological concepts.

By presenting three demonstrations of the use of PLANI, this thesis provides evidence that the framework is flexible, comprehensive, and simple to use. The first demonstration of PLANI is the Plant Creator software system, which creates plant models and animations while interacting with a human user.
via a graphical user interface. This system incorporates the other two demonstrations as components. It also provides the ability to save and reload information about plant structure and taxonomy.

The second demonstration of using PLANI is a method for incorporating the effects of wind on plant growth. The method was created by selecting objects from the four domains of PLANI framework and implementing the corresponding algorithms. There are currently no other published studies that consider the effects of wind on dynamic plant model growth in animation.

The last demonstration of PLANI effectiveness is a method for animating plant colour change and growth with consideration of nutritional availability. Any deviation from the required nutrient concentration range influences a plant’s size, shape, and colour, by restricting the growth and changing the colour of various tissues. The example used here is a virtual wheat (Triticum) plant that is affected by the availability of nutrients, which provides added realism when used for animation. There are currently no other published methods for plant animation that consider nutritional availability effects on plant colour change and growth.
Acknowledgements

This thesis was supported through a Faculty of Graduate Studies Research scholarship from the University of Regina. I would like to thank Dr. Howard Hamilton for his countless hours of discussions and feedback. He is an excellent teacher and researcher who went above and beyond the call of duty to help me complete my Master of Science degree. I would also like to thank Dr. Xue Dong Yang for the time he spent helping me understand graphics concepts, Dr. Boting Yang for helping me understand mathematical concepts, and Dr. Mary Vetter for providing insight into biological concepts. Also, I would like to thank Dr. Shaun Fallat for assisting me with understanding mathematical equations required for my thesis. Moreover, I would like to thank Richard Hamilton for assistance with coding and understanding my concepts even when I could not explain them well. I would like to thank Shelby Piechotta for getting my program to work on other computers. Lastly, I would like to thank my editors, Jacqui Smith and Catherine Milos, for the hours spent to help me improve the wording of this thesis.
Dedication

I dedicate this thesis to my family who endured countless nights and days of inattention. With their encouragement and support, I was able to complete this thesis and the rigorous research that was required to support it.
# Table of Contents

Abstract ................................................................................................................................. ii

Acknowledgements ............................................................................................................. iv

Dedication ............................................................................................................................ v

Table of Contents .............................................................................................................. vi

List of Tables .................................................................................................................... ix

List of Figures ................................................................................................................... xi

Chapter 1  Introduction ................................................................................................. 1
   1.1 Plant Animation ....................................................................................................... 1
   1.2 Motivation and Objectives .................................................................................... 2
   1.3 Thesis Outline ....................................................................................................... 5

Chapter 2  Background ................................................................................................. 7
   2.1 Dynamic Plant Models .......................................................................................... 8
      2.1.1 Simulation Dynamic Plant Model Approach ............................................... 11
      2.1.2 Artistic Dynamic Plant Model Approach ...................................................... 45
      2.1.3 Digitally Scanned Dynamic Plant Model Approach ...................................... 47
   2.2 Ontologies ............................................................................................................. 51
      2.2.1 The Plant Ontology ...................................................................................... 53
      2.2.2 The Plant Trait Ontology .............................................................................. 57
      2.2.3 Gene Ontology .............................................................................................. 58
      2.2.4 The Plant Environmental Ontology .............................................................. 59
4.2.3 Stage 4: Coding the Animation .......................................................... 126
4.2.4 Results of the Animation ................................................................. 130
4.2.5 Discussion ....................................................................................... 132
4.3 Animating Nutrient Effects on Plant Growth and Colour ............... 134
  4.3.1 Stage 2: Object Selection ............................................................... 134
  4.3.2 Stage 3: Designing the Algorithms ............................................... 135
  4.3.3 Stage 4: Coding the Animation .................................................... 154
  4.3.4 Results of the Animation ............................................................... 155
  4.3.5 Discussion ....................................................................................... 158
Chapter 5 Conclusion ............................................................................. 159
  5.1 The PLANI Framework ..................................................................... 159
  5.2 Animating Wind Effects on Plant Growth ......................................... 161
  5.3 Animating Nutrient Effects on Plant Growth and Colour ............... 161
  5.4 The Plant Creator System ............................................................... 162
  5.5 Suggestions for Future Research ..................................................... 162
List of References .................................................................................... 165
Appendices .............................................................................................. 195
 Appendix A: Biology Detail ................................................................... 195
 Appendix B: Figures .............................................................................. 214
 Appendix C: Tables ................................................................................ 271
 Appendix D: Stage 2 Selection Detail ..................................................... 292
List of Tables

Table 1: The values of constants used to generate branching structures (Prusinkiewicz, Hammel, Hanan, & Měch, 1996a; Prusinkiewicz, Karwowski, Perttunen, & Sievänen, 1996b) ........................................................................................................... 271

Table 2: Examples of objects selected for three scenarios ........................................... 271

Table 3: Effect of ENVIRONMENT on FUNCTION and FORM ..................................... 273

Table 4 Radius variation leaf shape generation ................................................................ 273

Table 5: Growth rates for plant species ........................................................................... 282


Table 7: Determining colour changes ............................................................................. 286

Table 8: Colour change from limiting nutrient ................................................................. 288

Table 9: Colour changes in wheat leaves (Snowball & Robson, 1991) ....................... 288

Table 10: Number of days for wheat growth (adapted from (Stapper & Fischer, 1990 \(^1\); Calendar, 2013) ......................................................................................................................... 289

Table 11: Wheat nutritional requirements (adapted from McKenzie, 1998) ...... 289
Table 12: Nutrient influences the physiology of the plant, as explained in (Uchida, 2000; Tucker, 1999; Murdock & Call, 1999; McCauley, Jones, & Jacobsen, 2009). ................................................................. 290

Table 13: Nutrient ratios for major types of plants¹ (Knecht & Goransson, 2004), ......................................................................................................................................................... 291

Table 14: Nutrient ratios for wheat................................................................................................................................. 291

Table 15: General effects on colour due to plant nutrient deficiency (Römheld, 2012¹; Uchida, 2000²). ................................................................................................................................. 291
List of Figures

Figure 1: Turtle example for F-F+F................................................................. 214
Figure 2: Vine-like branches created in L-Studio 4.0 ....................................... 214
Figure 3: Nine branching structures produced by varying the constants of a parametric D0L-system (Prusinkiewicz, Hammel, Hanan, & Mĕch, 1996)....... 215
Figure 4: Mesotonic tree example (Prusinkiewicz, Hammel, Hanan, & Mĕch, 1996) ........................................................................................................................................... 215
Figure 5: Tree Crown parameters (Cescatti, 1997) ............................................. 216
Figure 6: Schematic representation of two phytomers (n – 1 and n) of............. 216
Figure 7: Petiole movement model .................................................................. 217
Figure 8: Leaf rotation model ........................................................................... 217
Figure 9: Framework for interaction between plant and environment from Mĕch and Prusinkiewicz 1996........................................................................ 217
Figure 10: Cutting of a tree derivation steps 2 through 9 (Prusinkiewicz, Hammel, Hanan, & Mĕch, 1996).................................................................................................................................................. 218
Figure 11: A pruning of a branching structure from (Prusinkiewicz, Remphrey, Davidson, & Hammel, 1994a)............................................................. 218
Figure 12: Side view of tiger lily flower created using Maya ......................... 219
Figure 13: Top view of tiger lily flower created using Maya ............................ 219
Figure 14: Plant ontology .................................................................................. 220
Figure 15: Plant anatomical entity ................................................................... 221
Figure 16: Plant trait ontology ......................................................................... 222
Figure 17: Gene ontology ................................................................................ 223
Figure 18: Plant framework domains with process flows ........................................... 224
Figure 19: PLANI Plant Animation Framework Ontology ........................................ 225
Figure 20: FORM selection for the leaf example ...................................................... 226
Figure 21: FORM selection for the stem example .................................................... 227
Figure 22: FORM selection for the apical meristem ............................................... 228
Figure 23: FUNCTION domain selection example ................................................. 229
Figure 24: ENVIRONMENT domain selection example ........................................ 230
Figure 25: PLANI Plant Animation Framework Composition .................................. 231
Figure 26: Various plant forms utilizing the framework with formulae (original in colour). ........................................................................................................................................ 232
Figure 27: The Plant Creator system user interface ............................................... 233
Figure 28: System database schema ........................................................................ 234
Figure 29: Species specific example ........................................................................ 235
Figure 30: Tree with oval leaves ................................................................................ 236
Figure 31: Tree with compound leaf ......................................................................... 237
Figure 32: Vine example ........................................................................................... 238
Figure 33: Palmate with leaf variation ....................................................................... 239
Figure 34: Variation of leaf radius along the xy plane ............................................. 240
Figure 35: Branch movement ................................................................................... 241
Figure 36: Branch segments and nodes .................................................................... 241
Figure 37: Blade bending in response to wind ......................................................... 242
Figure 38: Petiole bending in response to wind ....................................................... 242
Figure 39: Leaf twisting in response to wind ............................................................ 243
Figure 40: Leaf vibration in response to wind ......................................................... 243
Figure 41: Response of petiole and leaves to wind: consecutive frames of animation .............................................................................................................................................................................. 244
Figure 42: Year seven motion in response to wind, images taken four seconds apart (16 hours of slow simulated time) ........................................................................................................................................................................................................ 244
Figure 43: Growth for two seconds (8 hours of growth in slow simulated time) with and without wind ............................................................................................................................................................................................................. 245
Figure 44: Examples of fruit and seed shapes .............................................................. 246
Figure 45: Effect of a limiting nutrient on growth ....................................................... 246
Figure 46: Effect of a limiting nutrient on growth in experiments ............................... 247
Figure 47: Leaf examples with triangle strip tessellation .............................................. 248
Figure 48: Adjacent vertices example ........................................................................... 248
Figure 49: Zadoks wheat development scale (Royo & Villegas, 2011; Zadoks, Chang, & Konzak, 1974) ............................................................................................................................................................................................................. 248
Figure 50: Average processing speed for colour change (original in colour) .... 249
Figure 51: Animation of colour changes over a series of days when nitrogen (N) is the limiting nutrient during spring (original in colour) ........................................................................................................................................................................ 249
Figure 52: Tip to base colour change an actual wheat leaf control of left and nitrogen limiting nutrient on right (adapted from (Snowball & Robson, 1991)) (original in colour) ............................................................................................................................................................................................................. 249
Figure 53: Base to tip colour change in an animated wheat leaf when boron is the limiting nutrient (original in colour) ............................................................................................................................................................................................................. 250
Figure 54: Animation of middle to edge colour changes over a series of days when calcium (Ca) is the limiting nutrient during spring (original in colour) ..... 250

Figure 55: Close up of an animated wheat leaf when limiting nutrient is calcium (original in colour) ................................................................. 251

Figure 56: Animation of colour changes over a series of days when manganese (Mn) is the limiting nutrient during spring; also represents molybdenum (Mo) colour change (original in colour) ............................................................ 251

Figure 57: Colour changes in an animated wheat leaf when limiting nutrient is manganese (original in colour) ................................................................. 252

Figure 58: Colour changes in an actual wheat leaf when limiting nutrient is manganese (adapted from (Snowball & Robson, 1991)) (original in colour) .... 252

Figure 59: Animation of colour changes over a series of days when potassium (K) is the limiting nutrient during spring; also represents iron (Fe) colour change (original in colour) ................................................................. 253

Figure 60: Colour changes in an animated wheat leaf when limiting nutrient is potassium (K) (original in colour) ................................................................. 253

Figure 61: Colour changes in an actual wheat leaf control on left and limiting nutrient potassium 3 samples (adapted from Snowball & Robson, 1991) (original in colour) ........................................................................................................... 254

Figure 62: Mottling in animated wheat leaf when limiting nutrient is zinc (Zn) (original in colour) ........................................................................................................... 254

Figure 63: Mottling in an actual wheat leaf when limiting nutrient is zinc (Zn) (adapted from (Snowball & Robson, 1991)) (original in colour) ....................... 255
Figure 64: Height comparison of animated wheat; control on left vs limiting nutrient nitrogen (N) on right (original in colour) ............................................................ 255

Figure 65: Height comparison of actual wheat plant control on left vs limiting nitrogen (N) on right (adapted from (Snowball & Robson, 1991)) (original in colour) ........................................................................................................ 255

Figure 66: Height comparison of animated wheat with control on left vs limiting nutrient potassium (K) on right (original in colour) ............................................................ 256

Figure 67: Height comparison of actual wheat; control on left vs limiting potassium (K) on right (adapted from (Cooper & Regents, 2009)) (original in colour) ........................................................................................................ 256

Figure 68: Height comparison of animated wheat; control on left vs limiting nutrient phosphorous (P) on right (original in colour) ............................................................ 257

Figure 69: Height comparison of actual wheat; control on left vs limiting phosphorous (P) on right (adapted from (Snowball & Robson, 1991)) (original in colour) ........................................................................................................ 257

Figure 70: Height comparison of animated wheat; control on left vs limiting nutrient magnesium (Mg) on right (original in colour) ............................................................ 257

Figure 71: Height comparison of actual wheat; control on left vs limiting magnesium (Mg) on right (adapted from (Yara, 2017)) (original in colour) ........ 257

Figure 72: Height comparison of animated wheat; control on left vs limiting nutrient sulfur (S) on right (original in colour) ............................................................ 258

Figure 73: Height comparison of actual wheat; control on left vs limiting sulfur (S) on right (adapted from (Snowball & Robson, 1991)) (original in colour) ........ 258
Figure 74: Height comparison of animated wheat; control on left vs limiting nutrient iron (Fe) on right (original in colour) ......................................................... 258
Figure 75: Height comparison of actual wheat; control on left vs limiting iron (Fe) on right (adapted from (Brennan and Scanlan, 2015)) (original in colour) ........ 259
Figure 76: Height comparison of animated wheat; control on left vs limiting nutrient copper (Cu) on right (original in colour) ......................................................... 259
Figure 77: Height comparison of actual wheat; various deficiencies in copper (Cu); from control on left to highest Cu deficiency on right (adapted from (Evans, 2001)) (original in colour) .............................................................................. 259
Figure 78: Height comparison of animated wheat with control on left vs limiting nutrient boron (B) on right (original in colour) ......................................................... 260
Figure A-1: Plant organization Part 1................................................................. 260
Figure A-2: Plant organization Part 2................................................................. 261
Figure A-3: Flower structure............................................................................. 261
Figure A-4: Generic leaf taken with Finepix digital camera............................ 262
Figure A-5: Leaf shape morphology (adapted from Swink & Wilhelm, 1994) using InkArt......................................................................................................... 262
Figure A-6: Simple and compound leaves (adapted from Swink & Wilhelm, 1994 ) using InkArt......................................................................................................... 264
Figure A-7: Leaf types taken with FinePix camera........................................... 265
Figure A-8: Leaf phyllotaxis (adapted from Swink & Wilhelm, 1994 ) using InkArt......................................................................................................... 266
Figure A-9: Orientation of leaves using L-Systems........................................267
Figure A-10: Stem shapes created using Blender........................................267
Figure A-11: Conifer trees taken with Finepix digital camera..........................268
Figure A-12: Conifer trees created with Blender........................................269
Figure A-13: Stem branching pattern created with Blender............................269
Figure A-14: Herbaceous plant from HP 2175 all-in-one printer (scanned)........270
Figure A-15: Stem apical meristem (adapted from Swink & Wilhelm, 1994) using InkArt.................................................................270
Chapter 1  Introduction

This chapter begins by introducing plant animation in Section 1.1. Then in Section 1.2 it explains the motivations and objectives for the research reported in this thesis. Finally, it presents a brief outline of the remainder of the thesis in Section 1.3.

1.1 Plant Animation

Plant animation has been studied for over forty years by graphics researchers within computer science (Honda, 1971). Plant animation refers to the creation of a sequence of images of a plant using a computer. Applications of plant animation can be broadly categorized into three types: video gaming, biological simulation, and digital effects in films. In video games, plant animation is an important part of scene generation. In biological simulations, plant animation is a means of visualizing numerical simulation results. In films, plant animation provides synthetic, often unique, plants from digital models that may be composed with other animations or photographs of natural scenes. There are three main approaches for plant animation: functional-structural, artistic, and digitized. Functional-structural approaches use mathematical representations of a plant derived from biological observations and theory. Artistic approaches use software that allows the artist to create representations of plants. Digitized approaches use digital cameras or sensing devices placed around plants to take a sufficient number of plant images from different viewpoints to allow reconstruction of the plant in an animation. In any case, each plant animation
requires a script based on ideas about spatial arrangement and the behaviour of the plant. This thesis proposes a conceptual framework for designing plant animations independent of the application type and the approach.

1.2 Motivation and Objectives

Since much time and energy is spent on designing and implementing dynamic plant models, this thesis provides a framework intended to simplify the design process. Informally, a dynamic plant model is a computer-based representation of a plant that can be used to simulate or animate natural behaviors, such as moving to wind, responding to light, and responding to nutrient availability. In addition, the research reported in this thesis is intended to increase the alignment between plant animations and natural plants.

The primary motivation for the research is to enable a link between the process of designing dynamic plant models and the information present in biological ontologies. Management of the data for even one dynamic plant model is challenging because a typical plant contains a multitude of organs and parts. The goal of designing complete and consistent dynamic plant models motivates us to ensure the design of a dynamic plant model is aligned with biological ontology. This alignment in turn supports communication about dynamic plant models between biologists and computer scientists. This thesis provides a conceptual framework based on a common semantic language.

A second motivation is to provide for enhancement of algorithms in plant animation in video games. Several massive, multiplayer online games (MMOs)
are played over long periods of time. This timeframe provides the best environment to simulate changes to plants such as rapid jungle regrowth, changes in plant appearance due to the seasons, and overall changes in plant composition in an ecosystem. Adding such environmental changes can enhance the gaming experience. Currently, limits on computational resources assigned to simulating the natural environment in video games (with less than 10% capacity available for this purpose) restricts the sophistication of changes to plants. It is hypothesized that with future increases in CPU and GPU capacity, increased resources will be available for simulating the natural environment, which will allow for simulated growth of plants.

The final motivation to this thesis is to provide authentic plant behaviour with respect to wind and nutrient responses, which will narrow the gap between the types of plant animations in biological simulations and those in video games. The idea is to allow for biological processes to be utilized regardless of the animation type: gaming, biological simulations, or films. The two demonstrations, effects of wind on growth and effects of nutrients on growth and colour, support this motivation. This research provides a way to implement real-time animation supported by biological concepts without exceeding available computational resources.

The objectives of this thesis are three-fold. The first objective is to provide a systematic approach to designing and developing dynamic plant models. This objective is achieved by providing the PLANI plant animation framework and the
Plant Creator software, which was implemented following the PLANI framework. PLANI is a new conceptual framework (Derzaph & Hamilton, 2017) for using plant ontologies to guide the design and coding of computer algorithms and the development of plant models. The Plant Creator software is available for download on google drive\(^1\) and on github\(^2\). A demonstration of its use is available on YouTube\(^3\). The second objective is to enhance plant animation through the incorporation of the effects of wind on plant growth and motion. This objective is achieved by using a software module that was incorporated into the Plant Creator software. A demonstration of the wind effects is available on YouTube\(^4\). The third objective is to enhance plant animation through the incorporation of the effects of nutrient influences on plants. This objective is also achieved using a module added to the Plant Creator software. Example animations with deficiencies of various nutrients are available on YouTube\(^5\) nitrogen\(^6\), potassium\(^7\), phosphate\(^8\), magnesium\(^9\), sulphur\(^10\), boron\(^11\), iron\(^12\), manganese\(^13\), zinc\(^14\), copper\(^15\), molybdenum, and no deficiency (control case)\(^16\).

---

\(^1\) https://drive.google.com/drive/folders/1W2oxIoFEluFlO6kGljd0Bp0DTe54Zb8I?usp=sharing
\(^2\) https://github.com/tinasturgeon/PlantCreator
\(^3\) https://youtu.be/30uy7NfeF_A
\(^4\) https://youtu.be/q1_RWuShhVY
\(^5\) https://youtu.be/LaWz7BGXe1s
\(^6\) https://youtu.be/a9L3Ou8wzFs
\(^7\) https://youtu.be/tfQ-m2GGp_4
\(^8\) https://youtu.be/b7R5bSSgq-8
\(^9\) https://youtu.be/shmOryRLu3E
\(^10\) https://youtu.be/WI3ivhsN4lo
\(^11\) https://youtu.be/HBWTV4tRf_E
\(^12\) https://youtu.be/bTIxZQ1-c
\(^13\) https://youtu.be/4u3HTkYm_Q4
\(^14\) https://youtu.be/n-NYi3QTI1A
\(^15\) https://youtu.be/1g4GhwypoAM
\(^16\) https://youtu.be/1gKycl7s8
This research offers an innovative framework, which uses plant ontologies, to design and run dynamic plant models in a virtual world. It also provides three demonstrations of the framework: the Plant Creator software tool, which enables a user to create, animate, and save dynamic plant models, optionally identified with existing species; a biological simulation of wind effects on motion and growth; and lastly a biological simulation of nutrient effects on growth and colour. The PLANI framework guides the specification of both design time and runtime processes for plant animations. PLANI also provides a systematic approach to incorporating algorithms to support simulation and animation based on dynamic plant models. Design time is shortened through the use of the plant ontologies by ensuring that a wide variety of information about relevant factors is available in a well-organized manner, so that less time is spent in identifying this information. This framework can be utilized regardless of the type of animation (gaming, biological simulation, or film) or the animation approach (functional-structural, artistic, or digitized).

1.3 Thesis Outline

The thesis is composed of five chapters. Chapter 2 provides an overview of recent relevant research in three areas: animation, biology, and ontology. Chapter 3 provides a description of the PLANI plant animation framework, including design time and runtime considerations. Chapter 4 is divided into three sections, which reflect the three demonstrations of the PLANI framework. The three sections describe a software tool supporting the creation and animation of
a variety of plants (Derzaph & Hamilton, 2017); animating the effects of wind on plant motion and growth (Derzaph & Hamilton, 2013b); and animating the effects of nutrient availability on plant growth and colour (Derzaph & Hamilton, in prep). Lastly, Chapter 5 offers conclusions and indicates future topics for research.
Chapter 2  Background

Computer animation is the process of rendering an object within a digital space and producing life-like motion for the purposes of movies or video games (Parent, 2012). Rendering refers to the production of a 2D image from a computer’s internal representation of a 2D or 3D object. A computer simulation is a mathematical model implemented using a computer to mimic a real-world system. Both computer animation and simulation are target applications for plant models, which provide a variety of possibilities from simple animations to complex simulations.

A plant model is a digital representation of a plant. A static plant model is a static digital representation of a plant that portrays its structure. In contrast, a dynamic plant model is a digital representation of a plant that portrays its structure and potential behaviours across time; it can be used for animation or simulation.

Excluding some artistic applications, dynamic plant models are based on biological structure and physiological behaviours of plants in nature. There are thousands of plant species that have the potential to be simulated and animated. Since an individual within a species has a certain structure and behaves in a certain way in response to external environmental factors, individual plants can also be simulated uniquely. Consideration of biological principles (plant morphology, physiology, and ecology) is required for a computer simulation of plant to be valid. The same is true, to a lesser extent, of animations.
summary of the relevant concepts concerning plant morphology, plant physiology, environmental response, and ecology is provided in Appendix A.

This chapter includes two sections to link animation with biology. Section 2.1 describes current approaches to creating plant animations. Section 2.2 offers an overview of ontologies with respect to plants and animation.

2.1 Dynamic Plant Models

The goal of creating a dynamic plant model is to portray realistic renderings of plants in an artificial environment across time. Three approaches in the field of computer animation to create a dynamic plant model: simulation (functional-structural plant models), artistic (image-centered), and digitally scanned.

A simulation dynamic plant model (SDPM) approach extensively considers the morphological, developmental, structural, physiological, and genetic aspects of plants. Based on these considerations, a dynamic plant model is designed that specifies the appearance and behaviour of an actual or imaginary plant in real time using mathematical models (adapted from (Prusinkiewicz, 2000)). SDPM utilizes the mathematical patterns determined within theoretical biology. Biological observations are formalized using linear algebra, differential calculus, physics concepts, chemical concepts, fractals, Fibonacci series, patterns and grammar-based approaches (Prusinkiewicz & Lindenmayer, 1990). A plant simulation is the implementation through computer
software of the behaviour of a dynamic plant model across time; for animation, its appearance is also determined.

An **artistic dynamic plant model** (ADPM) approach creates dynamic plant models using extensive user involvement and control involving art, mathematics, and computer science techniques. The ADPM approach examines the whole plant and aims for realistic portrayal of the surfaces, shape, shading, textures and colours using artistic techniques. It also employs mathematics such as geometry, linear algebra, trigonometry, and calculus implemented into computer algorithms to provide structure and motion.

A **digitally scanned dynamic plant model** (DDPM) approach incorporates digital images of a plant into a dynamic plant model through use of software (Sirault, et al., 2013). The DDPM approach can be used to simulate natural plant development (Deussen, Colditz, Stamminger, & Drettakis, 2002) or to generate visually correct plant animations (Deussen, Hanrahan, Lintermann, Mech, Pharr, & Prusinkiewicz, 1998; Prusinkiewicz, Students, & Collaborators, 1997).

Each of the above three approaches relies on knowledge of art, biology, mathematics, and computer science to varying degrees. Regardless of the approach taken, the images that compose plant animations based on dynamic plant models are created by a six-step process called the graphics display pipeline: (Parent, 2012)
Step 1: Object Modelling where the form and shape of all 3-D objects is specified relative to the origin of their individual object spaces,

Step 2: World space transformation, where the objects and virtual eye of the observer (called the camera) are placed in a virtual environment,

Step 3 View transformation, where the origin is moved to position of the camera and the camera’s line of site is focused on the center of interest among the objects in the world space.

Step 4: Perspective transformation, where the 3-D objects are transformed into a 2-D image space.

Step 5: Image to screen space transformation, where the objects are transformed into a 2-D screen space

Step 6: Repeat steps 1 and 5 while changing the characteristics of the observer or objects in the virtual environment.

This six-step process is implemented by the three approaches for creating a plant animations by generating the current state of the dynamic plant model in step 1, putting the resulting plant and the camera in the virtual environment in step 2, rendering the plant to the screen to view the results in step 3 - 5, and adjusting the camera or the dynamic plant model and then re-rendering in step 6.

The remainder of this section will describe algorithms used in each of the approaches in three sub-sections: simulation dynamic plant model (SDPM)
approach, artistic dynamic plant model (ADPM) approach, and digital dynamic plant model (DDPM) approach.

2.1.1 Simulation Dynamic Plant Model Approach

The benefits of utilizing SDPMs can be realized in biology, ecology, horticulture, commercial applications, and enhancements in educational experiences (Prusinkiewicz, 2004b; Room, Hanan, & Prusinkiewicz, 1996). Biological applications of plant simulation modelling, though not without controversy, are diverse. The study of structure, development, physiology, and genetic influence can all benefit from SDPM applications (Keller, 2002; Minorsky, 2003; Niklas & Enquist, 2002). Ecologists can use simulations to examine effects of environmental changes on plants and provide some evidence for probable future situations motivated by scenarios involving potential changes to climate, atmosphere, or light exposure. In Horticulture, the creation of plant varieties through plant simulations can enhance plant breeding results by showing the potential usefulness of certain traits. Agronomy can explore the competition for space and resources (Room, Hanan, & Prusinkiewicz, 1996). Plant simulation models may present opportunities in relevant fields for formulating and testing new theories.

**Functional-structural plant modeling** (FSPM) is a two-fold SDPM approach that includes a plant structure (skeleton) component and a plant physiology (process) component (Vos, Evers, Buck-Sorlin, Andrieu, Chelle, & de Visser, 2009). To provide a complete morphology for a plant, a structural method, such as Lindenmayer systems (L-systems) (Lindenmayer, 1968a;
Lindenmayer, 1968b), is used to generate the stem structure (Prusinkiewicz, Mündermann, Karwowski, & Lane, 2001). The function component affects the generated structure through parameter changes during animation resulting in plant motion, growth, or other changes to plant morphology (Sievänen, Nikinmaa, Godin, Lintunen, & Nygren, 2013).

Many models and methods, such as L-systems, fractal IFS, particle systems, reference axis, automation model, and dual-scale automation, have been developed to represent a plant’s topological and geometrical structure (Zhao, 2001). Models and methods offering foundation for the PLANI approach include Honda’s recursive branching (1971), Ulam’s competition for space (1962), Reeves and Blau’s particle system (1985), Shinozaki et al.’s pipe model theory (Shinozaki, Yoda, Hozumi, & Kira, 1964a; Shinozaki, Toda, Hozumi, & Kira, 1964b; Oohata & Shinozaki, 1979), and Prusinkiewicz’s use of L-systems (Prusinkiewicz & Lindenmayer, 1990).

Honda’s recursive branching approach used geometric attributes, such as branching angles and length ratios of internodes, to vary the plant’s branch sizes. Improvements included randomization (Aono & Kunii, 1984; Bloomenthal, 1985; Bloomenthal, 1995; Bloomenthal & Wyvill, 1990; Borchert & Honda, 1984; Honda, Tomlinson, & Fisher, 1981) and a variation algorithm involving parametric L-systems (Prusinkiewicz & Lindenmayer, 1990; Prusinkiewicz, Mündermann, Karwowski, & Lane, 2001). Honda’s approach of segmentation was used to make the branches and also the petioles within the demonstrations.
described in Chapter 4. The nested L-systems allowed for variation in plant branch and vein sizes.

The space colonization algorithm treats competition for space as its primary function, using the attraction points as a guide to generate branch growth towards these points (Ulam, 1962). Stevens (1974) augmented Ulam's competition for space approach with geometric attributes such as branch widths and angles. The augmented space colonization approach was then used by Runions et al. (2005) for tree branching. Space colonization is well-suited to simulating irregular forms of temperate-climate deciduous trees, although it can take several seconds to minutes to generate a tree (Runions, Lane, & Prusinkiewicz, 2007). As the goal in the thesis was to have real-time rendering, this method was not suitable. These concepts are used for background purposes and are not used in the demonstrations described in Chapter 4.

A particle system is a process where a series of small moving images is used to portray a natural occurrence such as fire (Beaudoin, Paquet, & Poulin, 2001). Reeves and Blau developed a particle system for rendering forests and meadows (Reeves & Blau, 1985). It was used by Chiba et al. (1997) for landscapes. This approach was also used for particle flow systems to portray branch and leaf venation structures by (Eristad, 2002; Prusinkiewicz & Hanan, 1989; Prusinkiewicz & Hanan, 1990; Rodkaew, Siripant, Lursinsap, & Chongstitvana, 2002a; Rodkaew, Chongstitvatana, Siripant, & Lursinsap, 2003; Rodkaew, et al., 2002b; Rodkaew, Chuai-Aree, Siripant, & Lursinsap, 2004a) and extended to model tree crowns (Neubert, Franken, & Deussen, 2007). The
advantage of this method is that it can generate trees from photographs realistically. The disadvantage is three-fold: it takes several seconds to generate the tree, smaller branches are not easily generated, and the team was not able to create all tree species (Neubert, Franken, & Deussen, 2007). As such, this method was not suitable for the real-time implementation goal. These concepts are used for background purposes and not used within the demonstrations described in Chapter 4.

Shinozaki et al.’s pipe model theory (1964a; Oohata & Shinozaki, 1979) was used to generate morphological plant models (Chiba, 1990; Chiba, 1991) for crop research and forestry. The pipe model theory is a combination of hierarchical pipes forming nodes and internodes resulting in a binary tree that represents natural plant structure. The pipe represents the plant stem. A node is at the end of a pipe which represents where plant leaves are attached. An internode is a pipe segment between two nodes. Nodes are classified as either internal or external. An internal node has exactly two children, whereas an external node has zero children. Kruszewski and Whitesides (1998) modelled the branching structures of apple, white oak, and white fir trees based on pipe model theory (Oohata & Shinozaki, 1979; Shinozaki, Toda, Hozumi, & Kira, 1964b). The advantage of using this model is that it simulates a plant directly as biologically plants have nodes and internodes. The limitation of this model is that the number of nodes is restricted to exactly 2, but this limit does not exist in nature. This concept was used for the connection of branching in the
demonstrations described in Chapter 4 with the addition of multiple node connections.

The most commonly implemented plant structure model is the L-system, described in the next section. L-systems, although high in computational demands, provide the flexibility in producing an infinite number of plant models with the use of an axiom and production rules. This method was chosen for its flexibility in plant generation and its simplicity in algorithmic design. This plant model approach was used in the three demonstrations in Chapter 4.

2.1.1.1 L-Systems

L-systems are one of the most extensively utilized fractal branch structure approaches for simulations today (Stocker, Uhrmann, Scholz, & Sievanen, 2013). An L-system plant structure is based on fractal mathematics and their pattern generation (Hutchinson, 1981; Barnsley, Elton, & Hardin, 1989). Fractal algorithms can be theoretically infinite in calculation; however, due to limitations of the computer resources, they are restricted to finite time by parameters such as the total number of iterations. Other parameters used to construct the fractal, such as shape and colour, influence the amount of detail seen by the viewer of the fractal image.

An Iterated Function Systems (IFS) is one type of mathematical system for generating polygons by repeated substitution. An Iterated Function System is composed of an initiator polygon and production rules. The fractal algorithm takes an existing initiator polygon edge and compares it with the production
rules. The fractal algorithm then substitutes into the existing edge the polygon from the corresponding production rule, resulting in a new polygon. This process is repeated for the total number iterations required to generate the desired fractal image. There are two basic types of substitutions: deterministic and probabilistic. A **deterministic** substitution occurs if there is only one applicable production rule for an initiator or existing polygon edge; whereas, a **probabilistic** (stochastic) substitution occurs if there exist multiple applicable production rules and probability parameters guiding their selection.

The basic concept of L-systems grammar developed by Lindenmayer in 1968 is similar to an IFS. First a string is initialized to the alphanumeric terms contained in the initiator (axiom). Then the string is modified using substitutions specified by production rules, which also containing alphanumeric terms. This substitution occurs through a parallel rewriting system. The number of iterations of substitutions is controlled by a limit on the number of iterations. Because an L-system uses alphanumeric terms, it is known as a **grammar-based system** rather than a graphically based IFS. L-systems substitution criteria can be either deterministic or probabilistic (stochastic), similar to IFS. In addition, an L-system can have either a context-free or context-sensitive rewriting systems (Prusinkiewicz, Erasmus, Lane, & Harder, 2007a; Prusinkiewicz, Lane, & Mēch, 2007b; Prusinkiewicz, Allen, & Escobar-Gutierrez, 2007c). In a **context-free** rewriting system, the term in the string determines the production rule utilized for substitution, whereas, in **context-sensitive** rewriting system, the adjoining terms in the string determine the production rule used.
L-systems have been extended to incorporate a variety of simulations and techniques (Prusinkiewicz & Hanan, 2013). L-system plant models can accurately reproduce the structure and development of individual plants, show how architectural parameters (branching angles, elongation rates, vigour of branches) affect their appearance, simulate plant physiology, investigate the effects of manipulations (pruning), simulate external environmental conditions (local light microclimate, water availability, crowding) on plant development, and simulate plants in their ecological contexts (Prusinkiewicz, 2004a; Prusinkiewicz, 2004b; Prusinkiewicz, 2003c).

A parametric L-system, as used in biological simulation, is defined in equation 2.1 as an ordered quadruple (Prusinkiewicz, Lane, & Mĕch, 2007b; Prusinkiewicz & Rolland-Lagan, 2006; Parent, 2012):

$$ G = \{V, S, \omega, P\} $$

(2.1)

where $G$ is the grammar. $V$ is a set of alphabet letters. $S$ is the set of formal parameters. The **axiom**, $\omega$, is a nonempty set that is composed of elements from $S$ and $V$. $P$ is a finite set of **productions** containing elements from the sets $V$ and $S$ with the addition of arithmetic expressions. The assumptions are: that the set of alphabet letters $\{a_1, a_2, \ldots, a_n\}$ is an element of $V$, and the set of parameters $S$ is a subset of the set of real numbers $\mathbb{R}$. The syntax for the standard L-system production rule, $P$, is: Predecessor: condition $\rightarrow$ successor. The predecessor contains elements from $V$ and gives the terms that will be used to compare to the grammar term. If the predecessor is equal to the grammar
term, the condition, containing elements from S with additional arithmetic expressions, tells which successor to use for the substitution. The successor, containing elements from V and S, is then used to replace the grammar term, creating a new grammar. Suppose given the following: axiom is FA, a production, $P_1: F \rightarrow FA$ and the total number of iterations is 3. The resulting string for the first substitution iteration is $FAA$, second substitution iteration is $FAAA$, and the third substitution iteration is $FAAAA$. For a second example, consider a string (words) built of two letters a and b. Each letter is associated with a rewriting rule. The rule $a \rightarrow ab$ means that the letter a is to be replaced by the string $ab$. The rewriting process starts from a string called the axiom. Assume that it consists of a single letter $a$. In the first derivation step (the first step of rewriting) the axiom $a$ is replaced with $ab$.

The graphical interpretation of the L-system can be visualized using a turtle. For example, F symbolizes that the turtle is to go forward by one unit while drawing the graphical representation. The – indicates that the turtle is to turnaround the vertical axis left by several degrees. The + indicates a turtle is to turn right by several degrees. For example, a string F-F+F will cause the turtle to move forward, turn left, move forward, turn right, and move forward as shown in Figure 1.

A method of portraying a fractal-like plant using L-systems with consideration of phyllotaxis, inflorescence, and variation among plants of the same species has been devised (Prusinkiewicz, Lindenmayer, & Hanan, 1988). An example of herbaceous-like branching generated by the author is shown in
Figure 2. This image was generated using an L-system to produce an herbaceous-like fractal stem structure. The L-system contained one axiom and four productions rules shown on this page. The axiom initially draws two segments of unit length (FF). The next four productions are randomly chosen using a probability of 25% for each production. The first production grammar $F[+FF]F$ starts by drawing one-line segment using the $F$ symbol. Next, it places the current position and orientation in memory based on symbol [. Thirdly, after encountering the + symbol it rotates by $45^0$. Next, drawing two-line segments are drawn to the screen, based on the two $F$ symbols within the brackets. When the symbol ] is encountered, the original position is obtained from memory replacing the current position. Lastly, due to the $F$ symbol, one more line segment is drawn. Running the L-system for 5 iterations results in generating a spiralling effect, as shown in Figure 2.

axiom:  \[ FF \]

$p_1$: $F \rightarrow F[+FF]F$ at $1/4$ probability

$p_2$: $F \rightarrow F[+F]F$ at $1/4$ probability

$p_3$: $F \rightarrow F[-FF]F$ at $1/4$ probability

$p_4$: $F \rightarrow F-F$ at $1/4$ probability

A bracketed L-system incorporates the '[' symbol to represent pushing the current state of the turtle onto a stack ([), and to represent setting the current state of the turtle to the state obtained by popping the contents from the stack.
when the ""]" symbol is encountered. This results in the position of the turtle being back at the start of the bracket.

A D0L-system is a context-free deterministic L-system. An L-system is context free if every production rule refers to an individual symbol independently of any other symbol in the grammar. Deterministic is where a symbol has exactly one production rule. An L-system is deterministic if there exists only one production rule for one symbol (Prusinkiewicz & Lindenmayer, 1990).

A more complex example of parametric D0L-systems (deterministic - 0L-systems) representing nine branching structures selected from a continuum of constants can be generated by the following grammar components: (Prusinkiewicz, Hammel, Hanan, & Mĕch, 1996a):

axiom, \( w \): \( A(100, w_0) \)

production, \( P_1 \): \( A(s, w): s >\text{min} \rightarrow ! (w) F(s) \)

\[
[+ (\alpha_1)/(\phi_1)A(s * r_1, w * q ^ e)]
\]

\[
[+ (\alpha_2)/(\phi_2)A(s * r_2, w * (1-q) ^ e)]
\]

where \( A \) is an apex, \( P_1 \) is a non-identity production, \( F \) is the internode, and \( \alpha_1 \), \( \alpha_2 \), \( \phi_1 \), and \( \phi_2 \) are angle values determining orientation of the apices. The \( s \) and \( w \) parameters specify internode length and width, respectively. \( !(w) \) sets line width to \( w \) or decreases line width by a default value and \( F(s) \) sets line width to \( s \). The constants \( w_0 \), \( q \), and \( e \) control branch width, where \( w_0 \) is the initial stem diameter, \( q \) is the difference in width between descendant branches, and \( e \) is the
reduction in branch width for descendant branches. The constants $r_1$ and $r_2$ determine the decrease in internode length that occurs through descendant branches of the tree from base to apex. The $\text{min}$ represents a threshold value that will prevent formation of any branch that is less than the threshold. The $P_1$ production rule replaces the apex, $A$, by internode $F$ with a length of 100, and two more $A$ apices only when the internode length, $s$, is greater than $\text{min}$. By varying the parameters (Table 1) one can produce the nine branching structures shown in Figure 3 (Prusinkiewicz, Karwowski, Perttunen, & Sievänen, 1996b).

Shortcomings of using L-systems for simulation plant models include, complexity of modeling plant growth due to geometrical and topological information requirements, high computer resource demands, and complex rule design. Improvements for each shortcoming are being worked on by researchers through investigation of genetic algorithms to: (1) decrease the complexity of L-system rule generation to simulate leaf images (Rodkaew, et al., 2002b), (2) improve parametric L-system performance (Traxler, C.; Gervautz, M., 1996), (3) optimize the animation system (Noser, Stucki, & Wlaser, 2001)), and (4) modeling plant growth (Rodkaew, Chuai-Aree, Siripant, & Lursinsap, 2004a), (Derzaph & Hamilton, 2013b)). Creation of algorithms focused on saving memory space through improving the rendering speed and the quality for the simulation of 3-D plant images such as sub-structure algorithms (Ding, Zhang, & Zhou, 2006), an algorithm for the generation of complex structures (Kang & Dengler, 2004), and an algorithm used to calculate the number of a
plant’s organs (Yan, Francois, de Reffye, Hu, Jaeger, & Leroux, 2002) have been explored.

2.1.1.2 Leaf Structure

Leaves have several characteristics that must be taken into consideration when creating a dynamic plant model. Leaf characteristics include unit area, venation, shape, margin shape, presence and types of trichomes, and surface texture. Early models for leaves have been built as models include leaf surface models represented by spline patches (Bloomenthal, 1985; Prusinkiewicz & Lindenmayer, 1990), reaction-diffusion models of leaf venation (Meinhardt, 1976), and a mathematical model for development (Scholten & Lindenmayer, 1981). Relevant algorithms also include the Particle Transportation system used to model leaves on the basis that leaf venation generates the leaf colour (Rodkaew, Siripant, Lursinsap, & Chongstitvatana, 2004b; Rodkaew, et al., 2002b), an algorithm generating lobed leaves (Mündermann, 2003), an algorithm using generalized cylinders to model leaves (Lintermann & Deussenn, 1996b; Prusinkiewicz, Mündermann, Karwowski, & Lane, 2001), and real time rendering of plant leaves (Wang, Wan, & Baranoski, 2004). A generalized cylinder is formed by sweeping a 3D generating curve, which determines the organ’s cross-section, along a trajectory called a carrier curve that defines the organ’s axis (Snyder, 1992).

Additional models include L-systems (Prusinkiewicz & Lindenmayer, 1990; Prusinkiewicz, Hammel, & Mjolsness, 1993) and a pinnate leaf model using L-systems (Prusinkiewicz, Hammel, & Mjolsness, 1993; Prusinkiewicz,
Another model calculates an implicit contour around a branching structure (Hammel, Prusinkiewicz, & Wyvill, 1992), and a physically based model of folded surfaces (Dimian, 1997). In addition, leaf edges have been portrayed using a continuous approach in simulation modeling (Marder, Sharon, Smith, & Roman, 2003). Moreover, a single formula for diverse natural and abstract shapes was proposed by Gielis (2003) and it can be utilized for leaf models. It is shown in equation 2.2.

\[
 r = f(\theta) \frac{1}{\sqrt{\left(\frac{|a|}{|b|} \cos \frac{m \theta}{2}\right)^{n_2} + \left(\frac{|a|}{|b|} \sin \frac{m \theta}{2}\right)^{n_3}}} 
\]

(2.2)

where \( n_1, n_2, n_3, \) and \( m \) are any positive real numbers, \( a, \) and \( b \) are any positive real number excluding zero, and \( \theta \) represents any angle from 0 to 2\( \pi \). This equation is parameterized to provide the \( x, y, \) and \( z \) coordinates to form the target shape by substituting \( x = r\cos \theta \) and \( y = r\sin \theta \) for \( f(\theta) \). All methods described were computationally demanding except for the Gielis formula. For this reason, the Gielis formula was used. It should also be noted that an approach to creating leaf shapes was created by this author (Derzaph & Hamilton, 2013a) and is described in the demonstrations in Chapter 4.

2.1.1.3 Stem Structure

Stem structure has two components: branching types and stem shape. Stems are usually spirally arranged around the branch separated by the golden angle of 137.5\(^{\circ}\) (Smith, Guyomare'h, Mandel, Reinhardt, Kuhlemeier, & Prusinkiewz, 2006a). Generally, examples of stem branching types are basitonic, mesotonic, and acrotonic (see appendix A for definitions). These
patterns of stem placement are implemented using the L-systems described previously and in further detail in the following tree structure section. The second component, stem shape, is based on parametric equations of a cylinder including height, $h$, radius, $r$, angle, $\theta$, and the coordinate points $(x, y, z)$, as shown in equation 2.3.

\[
\begin{align*}
    x &= r \cos \theta, \text{ for } \theta \in [0, 2\pi] \\
    y &= r \sin \theta, \text{ for } \theta \in [0, 2\pi] \\
    z &= [0, h]
\end{align*}
\] (2.3)

where $z$ is the vertical dimension. Improvements to stem simulation can be made by incorporating physical structure and internal physiology influences, such as wind or nutrient limitation, which will alter both stem placement and shape. These improvements to stem simulation are further described in section 2.1.1.6.

2.1.1.4 Tree Structure

One basis for tree modeling is the approach of using recursive branching structures. These recursive branch models first implemented by Honda (1971) contain initial parameters of branching angles and length ratios of consecutive internodes. Improvements to Honda’s model include randomization of parameters (Aono & Kunii, 1984; Bloomenthal, 1995; Borchert & Honda, 1984; Honda, Tomlinson, & Fisher, 1981) and the use of randomization combined with organized variation parameters as a function of position (Lintermann & Deussen,
1999; Oppenheimer, 1986; Prusinkiewicz, Mündermann, Karwowski, & Lane, 2001; Reeves & Blau, 1985; Weber & Penn, 1995).

Because Weber and Penn’s method was intended to allow a user to generate a plant from a picture or a book not from real plants it limits the generation of simulated plants. In addition, rendering of the plants was not fast enough for real-time generation. Reeves & Blau used particle systems which did not generate plants in real-time as well. Honda’s method limited the variation of plants through the use of a small number of parameters, as in the improvements made by Bloomenthal. In comparison, L-systems fulfills the generation of real plants, therefore is an appropriate method to be utilized and does render a plant in real-time.

In order to ensure that dynamic plant models portrayed tree structure correctly, Prusinkiewicz and Remphrey (2000) implemented various tree architectures using L-systems and compared them to the structure in nature by the use of Petri nets. A petri net (Peterson, 1981) describes concurrent processes and is asynchronous, allowing for comparison between tree architecture and computer simulation plant modelling. Petri nets were first applied to plant development by Lück et al. (1983), and later used to analyzed experimental data via petri nets (Lück & Lück, 1991; Barlow, 1994; Barlow, Brain, & Adam, 1989). Petri nets are well-suited to modelling the cyclical behaviour of the apical meristem, whereas L-systems are well-suited to modelling the number of branch modules and showing how they are connected.
As we are interested in the overall tree structure, L-systems are more suited to our requirements.

Trees with varying branching structures, such as basitonic, mesotonic, and acrotonic (see Section A.1.3 in Appendix A for information on tree branching types), can be generated using bracketed L-systems (Prusinkiewicz & Kari, 1994). The following is an example of a mesotonic structure represented as a parametric D0L-system (Prusinkiewicz, Hammel, Hanan, & Méch, 1996):

\[
\omega : FA(0)
\]

\[
p_1 : A(v) \rightarrow [-FB(v)][+FB(v)]FA(v + 1)
\]

\[
p_2 : B(v) : v > 0 \rightarrow FB(v - 1)
\]

where \( \omega \) is the axiom, \( p_1 \) and \( p_2 \) are the productions, \( F \) is the internode, \( A \) is the apex, and \( B \) is the lateral apex. The result of running this system for 9 iterations is shown in Figure 4.

Examples of models that involve tree crown patterns (also known as envelopes), which describe the overall appearance of a tree, have been modeled by Horn (1971) and Koop (1989). Cescatti (1997) enhanced the models by allowing for higher levels of asymmetry in crown shape using up to 18 parameters for each individual virtual tree. The 18 parameters, as shown in Figure 5, were gathered from field measurements. They included the crown apex height, \( H_t \), and the crown base height, \( H_c \). An additional measurement at the height, \( H_{pi}^i \) ( \( i = 1, \ldots, 4 \)) of the greatest crown width in each of four directions.
was also taken. In addition, 4 crown radii, $C_i (i = 1, \ldots, 4)$, and 2 shape coefficients per each direction, $E_p^i (p = 1, 2, i = 1, \ldots, 4)$, were measured at the corresponding height $H_p^i$. To generate the virtual tree, the field measurements were input into the Cescatti envelope as the corresponding six control points, $P_1$ to $P_6$, and two shape coefficients, $C1$ and $C2$. $P_1$ is at the top of the crown, $P_2$ is at the bottom, and $P_3$ to $P_6$ describe a peripheral line at the greatest width of the crown projection on the $xz$-plane. Constraints dependent upon the plane are placed on $P_3$ to $P_6$. These variations illustrate the diversity of plants which is also demonstrated within the software Plant Creator System described in Chapter 4.

2.1.1.5 Herbaceous Plant Structures

An herbaceous plant is one that has a soft stem. Grasses, herbs, mosses, and various flowers are herbaceous plants. Moss and other plants that are formed from the aggregation of particles can be modelled based on the fβm of particles in 2D or 3D space using diffusion-limited aggregation (DLA) systems. A DLA system works by placing a static initial structure in a defined space. This structure is used to determine particle creation around it using a Gaussian distribution of particle placement. A Gaussian distribution of particle placement predicts the location of various particles in the 2D or 3D space based on the normal distribution (bell) curve with the initial structure as the bell curve’s apex. All the generated particles move constantly within the defined space towards the initial structure or leave the space. The model continues until all particles are removed from the defined space or a certain
parameter restricts its continuance, usually the time parameter (Nikiel, 2007). Another approach using voxel space automata (Greene, 1991; Greene, 1989), the 3-D extension to cellular automata, has been used to model climbing plants (Arvo & Kirk, 1988; Greene, 1988). A **voxel space automation** is defined as a growth function that senses and reacts to a voxel space using geometric constraints such as intersection avoidance and proximity. A **voxel space** is a 3-D space, usually rectangular, divided into identical units of space known as voxels. Several other models of herbaceous plants have been developed such as a maize model (Fournier & Andrieu, 1998) and synthetic topiary (Prusinkiewicz, Remphrey, Davidson, & Hammel, 1994a; Prusinkiewicz, Lindenmayer, & Hanan, 1988) based on L-systems.

Wheat has been modelled to study tiller growth using L-systems (Vos, et al., 2009). A **tiller** of a wheat plant is analogous to a branch of a tree. The effects of nitrogen, phosphorous, carbohydrate and infrared/red light were modeled focusing on tillering. A modular approach is taken to model wheat, the base unit is a phytomer shown in Figure 6. The wheat phytomer consists of an internode with a tiller bud at the bottom, a node above the internode, a sheath which is inserted on the node, and a leaf blade (Briske, 1991; Moore & Moser, 1995; Scanlon & Freeling, 1997); the collar marks the transition between sheath and blade.

**2.1.1.6 Plant Functions**

In nature, plant functions influence organs on a plant, such as leaves, branches, buds, roots, and bark, by causing changes in shape and structure. To
obtain realistic simulated plants, these plant functions must be implemented in plant simulation models. Some examples of plant functions are locomotion, growth, nutrient absorption and photosynthesis. **Locomotion** results in movement of the plant in response to wind, **growth** occurs in response to environment conditions, **nutrient absorption** occurs in response to nutrient availability in the soil, and **photosynthesis** occurs in response to light availability. Algorithms that incorporate plant functions are categorized into random and organized (Remphrey & Prusinkiewicz, 1997). A *random approach*, such as stochastic modelling (Fournier & Fussell, 1980), alters a model’s parameters randomly around a default value. An *organized approach* methodically alters parameters according to predefined interpolants and procedural functions (Prusinkiewicz, Mündermann, Karwowski, & Lane, 2001). Both types of algorithms are applied to change the plant structure at runtime to enable variation and enhance plant realism. This section focuses on two physiological functions of plants, locomotion (motion) and growth for leaves and stems. We describe a combination of random and organized approaches, to provide context for the demonstrations described in Chapter 4.

**Leaf Motion**

Studies have examined leaf motion with respect to wind flow (Wejchert & Haumann, 1991), biomechanics (Jirasek, Prusinkiewicz, & Moula, 2000), spring and rotational movement (Feng, Chen, Yan, & Wu, 2001), wilting (Lu, Guo, Zhao, & Li, 2008), response to wind by broadleaf types (Wu, WenHui, & Guanghui, 2006), response to wind by both broadleaf and needle leaves (Feng,
Chen, Yan, & Wu, 2001), as a component of overall plant motion (Beaudoin & Keyser, 2004), and as part of shell space (Yang, Sheng, Wu, & Sun, 2009). Additional studies have calculated the motion of a leaf based on inheriting the motion from the branch (Habel, Kusternig, & Wimmer, 2009; Zioma, 2007) or calculated it separately from the branch by considering physically-based wind influences (Li, et al., 2010; Qin & Chen, 2006; Wu, WenHui, & Guanghui, 2006). Wong and Datta (2004) divided leaf motion into three rotations, namely yaw, pitch and roll, representing x, y, and z rotations, respectively. Wong & Datta (2004) simplified the complex physics calculations required by Feng et al (Feng, Chen, Yan, & Wu, 2001) yet produced dynamic plant models with realistic wind responses.

The leaf motion technique by Feng et al. (2001) can be considered a foundational technique for leaf movement. This technique used two rotations, petiole and leaf, to make the final motion of the leaf. As shown in Figure 7, the final petiole direction is represented by the cross product of the two vectors, \( B \times F \) where \( B \) is a vector indicting petiole direction and \( F \) is the direction of the wind force on the leaf surface. Suppose the leaf surface area is \( A \) and the leaf surface density is \( \rho \). The leaf surface mass, \( m \), is obtained by integrating the density \( \rho \) over area \( A \), as shown in equation 2.4. As shown in equation 2.5, the angular acceleration, \( \epsilon \), of the petiole, is the ratio of the wind force on leaf surface, \( F \), over the petiole length, \( L \), times the leaf surface mass, \( m \). The petiole angular acceleration rotation angle, \( \theta \), is calculated according to equation 2.6 by twice integrating the angular acceleration, \( \epsilon \), of the petiole, across time.
To produce the motion, Feng et al. broke the petiole into a series of short segments and rotated each segment, sequentially, by angles from 0 to $\theta$.

$$m = \int_A \rho dA \quad (2.4)$$

$$\epsilon = \frac{F}{mL} \quad (2.5)$$

$$\theta = \iint \epsilon \quad (2.6)$$

The second rotation is of the leaf. It is assumed that there are two leaf rotations, one around $x$ axis and the other around the $y$ axis, as shown in Figure 8 using a local coordinate system. Assume that the $z$ axis is used to describe the movement of the petiole and is therefore not relevant to the leaf movement. In addition, because the movement along the $y$-axis is negligible, it is omitted from the final calculation. Suppose the leaf surface area is $A$ and the density function of the wind force is $f(x,y)$; then the moment load $M$ of the wind force on the leaf surface, relative to the x-axis, is given by equation 2.7.

$$M = \int_A f(x,y)y dA \quad (2.7)$$

Suppose also that the density of the leaf surface is $\rho$. The leaf surface’s moment of inertia $I$, relative to the x-axis, is given by equation 2.8.

$$I = \int_A \rho y^2 dA \quad (2.8)$$

The angular acceleration $\epsilon$ of the leaf surface is calculated using equation 2.9.

$$\epsilon = \frac{M}{I} \quad (2.9)$$
The rotational angle $\theta$ is obtained by twice integrating the angular acceleration of the leaf surface over time, as shown in equation 2.10.

$$\theta = \int \int \varepsilon$$  \hspace{1cm} (2.10)

The leaf motion used in the thesis is simplified from the detailed physical calculations just described. The motion of the leaf is calculated at the connection to the petiole instead of on all the segments of the petiole. The movement for the leaf is accomplished through three random directional movements whose speed is influenced by the wind. The motion of the petiole is calculated at the connection of the petiole to the branch. The motion is assumed to be a sway influenced by the wind. A more detailed description is provided in Chapter 4.

Leaf Growth

Simulating leaf growth has been done based on creating a compound leaf recursively (Prusinkiewicz & Lindenmayer, 1990), growing leaves continuously (Prusinkiewicz, Hammel, & Mjolsness, 1993), and considering a progression of leaf shapes for \textit{Arabidopsis thaliana} (Scarpella, Francis, & Berleth, 2004).

There are three general approaches to modelling leaf development. One approach specifies leaf shape over time: examples include the marginal growth algorithm (Scholten & Lindenmayer, 1981), diffuse growth algorithms (Prusinkiewicz, Hammel, & Mjolsness, 1993; Gomes, Darsa, Costa, & Velho, 1999), and auxin transport as proposed by Couder et al. (2002). The second
approach specifies a physically-based tissue growth model (Rolland-Lagan, Bangham, & Coen, 2003; Wang, Wan, & Baranoski, 2004), which used field data to relate growth patterns and shape in *Antirrhinum* petals.

A third approach is simple scaling of a shape. In other words, leaf growth is accomplished by increasing the length and width of the leaf. To do this, we can adapt the growth function for the stem provided by Rodkaew et al. (2004a), shown in equation 2.11:

$$ G(t) = L_{\text{min}} + \frac{L_{\text{max}} - L_{\text{min}}}{1 + e^{m(t-t)}} $$  \hspace{1cm} (2.11)

where $G(t)$ is the size at time $t$, $L_{\text{max}}$ is the maximum leaf length, $L_{\text{min}}$ is the minimum leaf length, $m$ is the growth slope of the leaf for a plant species, $T$ is the time when the length is $(L_{\text{max}} + L_{\text{min}})/2$, and $t$ is the independent time variable. The leaf width (also called the radius) is calculated analogously by substituting $R_{\text{max}}$ and $R_{\text{min}}$ for $L_{\text{max}}$ and $L_{\text{min}}$ in equation 2.11.

Previous to our research, animation of leaf growth effects did not include wind influences. Our approach as described in Chapter 4 has been published (Derzaph & Hamilton, 2013b).

**Stem Motion**

Studies of stem and branch movement in trees have approached the issue using physically based methods (Sakaguchi & Ohya, 1999; Akagi & Kitajima, 2006; Habel, Kusternig, & Wimmer, 2009; Ding, Chongcheng, Liyu, & Qinmin, 2009; Feng, Chen, Yan, & Wu, 2001) and procedurally based methods
Physically-based methods incorporate physical forces to maintain geometric relationships so as to portray accurate stem and branch motion. **Procedurally based methods** calculate motion by assigning arbitrary rotation angles and speeds to stems and branches. Additional stem motion approaches include motion capture (Long, Reimschussel, Britton, & Jones, 2009), simulation (Beaudoin & Keyser, 2004), spectral density noise-based real-time animation (Shin, Fujimoto, Tamura, Muraoka, Fujita, & Chiba, 2003; Wesslén & Seipel, 2005; Zioma, 2007), a hybrid of heuristic and physically-based approaches (Giacomo, Capo, & Faure, 2001), a vertex approach (Habel, Kusternig, & Wimmer, 2009), and shell dynamics (Yang, Sheng, Wu, & Sun, 2009). A few studies included motion of all components of the plant (Rodkaew, Chuai-Aree, Siripant, & Lursinsap, 2004a).

As an example, let us consider the physically-based approach. With this approach, the motion of a branch segment is described by equation 2.12 (Sakaguchi & Ohya, 1999):

\[ \mathbf{N} = (F_w + K + R + T_r) \times \mathbf{b} \]  

(2.12)

where \( \mathbf{N} \) is the moment of force (torque), \( F_w \) is the external force (i.e. wind), \( K \) is the resistivity, \( R \) is the axial damping force at a node, \( T_r \) is the force that propagates back from the child branches (if any), \( \times \) indicates the cross product, and \( \mathbf{b} \) is a vector from the node at the beginning of the segment (where it attaches to its parent segment) to the center of gravity of the segment.
A simplified form of equation 2.12 is used to animate wind effects on plant growth by adding the growth effects of wind directly to the normal plant growth. This simplified form is described in Chapter 4.

Stem Growth

In simulation plant modelling, the portrayal of the outcome of physiological effects such as growth is important. In real plants, stem growth occurs through the meristem. Several animation examples of such growth exist. One example is a program created by Karwowski (2002) that allows an apical meristem of any shape and size to be created from any small group of cells called the initials (Carbone, Gromov, & Prusinkiewicz, 2000). Another example is a model of a rhizome proposed by (Bell, Roberts, & Smith, 1979), which was re-simulated by using an L-system implementation (Prusinkiewicz, Hammel, Měch, & Hanan, 1995). A third example, utilizes a combination of discrete and continuous approaches, based on cell division patterns for a generic shoot apex (Nakielski, 2000). A discrete model has also been employed to evaluate surface growth in a shoot apex of *Anagallis arvensis* (Kwiatkowska & Dumais, 2003). In addition, Smith and Prusinkiewicz (2004) used their vv system (based on map L-systems) to model growth in a shoot apical meristem. Furthermore, another example used a shoot apical meristem model, proposed by Smith et al. (2006b), which reproduces the phyllotaxis of *Arabidopsis* vegetative shoots. Moreover, dynamic plant models were created with consideration for growth stages on all parts (
Simulation of stem growth can be accomplished by increasing the length and width of the branch segments represented by the ‘F’ in the L-string using equation 2.13 (Rodkaew, Chuai-Aree, Siripant, & Lursinsap, 2004a).

\[ G(t) = L + \frac{U-L}{1+e^{m(T-t)}} \]  

(2.13)

where \( U \) is the maximum branch length, \( L \) is the minimum branch length, \( m \) is the growth slope of the growth for a plant species, \( T \) is the time at \((U + L)/2\) and \( t \) is the independent time variable. Using nested parametric L-systems and an adaptation of equation 2.13, stem growth is implemented in Sections 4.1, 4.2, and 4.3 with environmental influences from wind and nutrients. A nested L-system is composed of L-systems, each with its own axiom, which refer to one another. In our case, the nested L-system is composed of three L-systems that represent branch, petiole, and vein. This technique allows for variation in the structure of each of the branching, petiole formation, and vein structure separately providing additional variation. For example, suppose the branch axiom is F, the petiole axiom is +F[-F][&F], and the vein axiom is +F[-F]. Each axiom is then replaced by the substitution string as described previously in this section.

Tree growth

Hammel et al. (1995) modelled ash tree growth, incorporating empirical data. From trees at two sites in Manitoba, data was collected every two days on
internode length, rachis segment length, leaflet length, and leaflet width. To obtain other useful measurements, such as stem radius (i.e. girth) and stem growth rate, they used photographic and field observations. They modelled the gradual changes in leaflet orientation with a cubic form of a sigmoidal function to illustrate the organ’s development. The quantitative and qualitative model was combined with differential L-systems (Prusinkiewicz, 1993b). To render the image, cylinders were used to represent the internodes and rachis segments, and cubic patches were used to represent the leaflets and bud scales.

In more detail, a methodology was developed to construct this model of an ash tree. This methodology consists of the following sequence of steps (Hammel, Prusinkiewicz, Remphrey, & Davidson, 1995):

- define a skeletal L-system (qualitative model)
- acquire field data (qualitative model)
- analyze field data
- estimate growth parameters that were not field measured
- define a quantitative model (differential L-system) by incorporating the skeletal L-system and the growth functions
- visualize the model
- evaluate the results
- repeat the above steps to improve the model

The methodology provides for transference of field data to the simulation model, which results in a realistic growth model of the species of interest. The number of iterations of the steps could be reduced if measurements were
obtained for all influenced structures. Such measurements could be obtained through digital sampling (Hanan, 1995; Mouli & Sinoquet, 1993) of the growth and the automation of the data into statistical programs to extrapolate growth for the plant models (Hammel & Prusinkiewicz, 1996). Plant structure, physiology, genetics, and environmental effects need to be considered for an accurate simulation plant model.

2.1.1.7 Environmental Effects

Plant simulation has been implemented with environmental effects such as limitations in water and nutrients, gravitropic effects, time cycles, and temperature (Dunbabin, Diggle, Rengel, & van Hugten, 2002; Diggle, 1988; Pagès, M.O., & D., 1989; Pagès, Assen, Pellerin, & Diggle, 2000; Somma, Hopmans, & Clausnitzer, 1998).

Incorporating interaction with the environment is essential to obtaining a comprehensive simulation of plants. It has been proposed that three forms of environment-plant interaction exist global, local, and a feedback loop between plant and environment (Mĕch & Prusinkiewicz, 1996). The first form depends on global properties, such as day length (Frijters & Lindenmayer, 1974), daily minimum and maximum temperatures (Hanan, 1995), wind (Wu, WenHui, & Guanghui, 2006; Ono, 1997; Shinya & Fournier, 1992), gravity (Barlow, Brain, & Adam, 1989; Zieschang, Brain, & Barlow, 1997), and rain. The second form depends on local properties (self-regulatory patterns), such as obstacles to growth (Arvo & Kirk, 1988; Greene, 1991), weight of branches (Fournier, et al., 1994; Schaffer, 1990), pruning (Boudon, Prusinkiewicz, Federl, Godin, &
Karwowski, 2003; Prusinkiewicz, Remphrey, Davidson, & Hammel, 1994a; Prusinkiewicz & McFadzean, 1992; MacKenzie, 1993), accessibility to support (Arvo & Kirk, 1988; Greene, 1989), predation responses (Hanan, 1992), trichome patterning on growing tissue (de Reuille, Runions, Smith, Coen, & Prusinkiewicz, 2007), and temperature in soil (Diggle, 1988). The third form depends on a plant feedback loop where the environment affects the plant, such as competition for space (Bell, 1986; Cohen, 1967; Ford, 1987; Ford, Avery, & Ford, 1990; Kaandorp, 1994; Kurth, 1994), competition for light (Greene, 1989; Chen, Ceulemans, & Impens, 1993; Chiba, Ohsida, Muraoka, Miura, & Saito, 1994; Chiba, Muraoka, Doi, & Hosokawa, 1997; De Reffye, et al., 1995; Kanamaru, Takahashi, & Saito, 1992; Takenaka, 1994), and access to water and nutrients (Clausnitzer & Hopmans, 1994; Liddel & Hansen, 1993; Kaandorp, 1994; Doussan, Pagès, & Pierret, 2003; Hua & Kang, 2011; Allen, Prusinkiewicz, & DeJong, 2005). The remainder of this section describes the global form of environmental influence, the feedback loop form of environmental influence, and environmental effects on plants with respect to pruning and topiary, water tropisms, wind algorithms and nutrient uptake.

Global Form of Environment Influence

The global form of the environment influencing the plant is the focus for the thesis. The plant is affected globally by wind through swaying of branches. The plant is also affected by wind through the bending, twisting, and vibration of leaves. The effect of nutrients on growth is also a global process where a limiting nutrient will cause growth length changes or colour changes to leaves.
and stem. Lastly, the, passage of time is controlled globally and effects seasonal changes on the plant which results in growth of branches and leaves during spring and summer. The leaves die during the fall and are absent during the winter for deciduous trees. The strength to utilizing global parameters is that each change is required in only one place and will affect all animation throughout. The weakness of global parameters is that there is no way to change the values locally to influence a subsection of an animation unless there is another global variable generated. This can cause a large list of global variables which become unmanageable. As the strengths of the global approach out weight the weakness, the global form of environmental effect was utilized in the Chapter 4 examples.

Feedbackloop Form of Environment Influence – An Environmental Framework

A modeling framework has been proposed by Mĕch and Prusinkiewicz (1996) to simulate and visualize environmental influences through a feedback loop between the plant and environment. This framework, shown in Figure 9, is conceptualized as two processes: from the environment to the plant and from the plant to the environment. The framework was implemented with an open L-system, which is an L-system supplemented with constructs for bi-directional communication between plant and environment. Open L-systems were introduced by Mĕch & Prusinkiewicz (1996) as an extension of environmentally-sensitive L-systems (Prusinkiewicz, Remphrey, Davidson, & Hammel, 1994a) that captured the impact of the environment on plant development. The
framework developed by Mĕch and Prusinkiewicz has the advantage of allowing algorithms to be added for any specific environmental influences, but it does not represent interactions between plants that could also influence the individual plants and the environment. It was used as a starting point for the proposed framework developed in this thesis.

Local Form of Environmental Influence - Pruning and Topiary

L-systems were extended with the cut symbol % (Hanan, 1992), which causes the removal of the remainder of the branch, as shown in Figure 10, to provide support for modification of the structure of simulated plant models. These L-systems were further enhanced to include responses to the environment. The enhancement, called an *environmentally-sensitive extension* of an L-system (Prusinkiewicz, et al., 1994), allows for *query modules* in a parametric L-system. The use of a cut symbol and a query module is illustrated in the following L-system, which describes an abstract developmental process influenced by the environment (Prusinkiewicz, et al., 1994):

\[
\omega : A
\]

\[P1 : A \rightarrow [+B][-B]F ? P(x, y)A\]

\[P2 : B \rightarrow F ? P(x, y) @ oB\]

\[P3 : ? P(x, y) : 4x^2 + (y - 10)^2 > 10^2 \rightarrow [+ (2y)F][- (2y)F]%,\]

where \(F\) represents a line of unit length, + and − without parameters represent left and right turns of 60°, the query module \(?P(x, y)\) returns the corresponding position, \(B\) represents a branch from a lateral apex, \(A\) represents the main apex, and \(@o\) represents drawing a circle with diameter equal to \(F\). If the query
module in production \( P1 \) appears during the reading of the L-string, it is replaced by the \([+ (2y)F][-(2y)F]\)% only if the condition \(4x^2 + (y-10)^2 > 10^2\) is true, according to production \( P3 \). The cut (%) prevents further branch development. This process is repeated for production \( P2 \). This L-system example results in a pruned branch structure confined to an ellipse, as shown in Figure 11.

The environmental query module \( ?P(x, y) \) allows communication with the environment. This module, as part of the environmental framework previously described, receives and transmits information to the simulation plant model. The overall result is that the plant and environment interact to simulate natural interactions.

A limitation of the environmentally-sensitive extension to the L-system approach utilized in the framework is the assumption of applying self-similarity consistently at every level in simple models (Prusinkiewicz, 1999). In fact, self-similarity does not occur consistently in mature trees due to exogenous and endogenous influences that discourage or encourage branching (Sachs & Novoplansky, 1995). These influences led to several other approaches, such as the space colonization algorithm (Runions, Lane, & Prusinkiewicz, 2007), particle systems (Rodkaew, Chongstitvatana, Siripant, & Lursinsap, 2003), and the use of Petri nets (Prusinkiewicz & Remphrey, 2000), as previously described in section 2.1.1.4.
Both the local and the feedback-loop were not used in the thesis and are provided as background information for a more global picture of plant-environment interactions.

Currently there is no graphical user interface (GUI) for a system for designing dynamic plant models that provides for a wide variety of environmental effects; therefore, commands and programming are required. A foundational GUI that allows design with respect to nutrients and wind response is described in Section 4.1.

Local Form of Environmental Influence - Tropisms

Tropism, for a plant specifically, is the motion of plant organs towards or away from external objects in the environment. Algorithms have been developed to portray the effects of gravity (Jirasek, Prusinkiewicz, & Moula, 2000), obstacles (Kaandorp, 1991), light competition (Chiba, Ohsida, Muraoka, Miura, & Saito, 1994), and light quality (Gautier, Mĕch, Prusinkiewicz, & Varlet-Grancher, 2000) on plants through a variety of methods. Within this thesis, wind and nutrient concentrations are the environmental influences of interest.

Global Form of Environmental Influence - Wind

Wind results from the pressure difference between an area of high pressure and an area of low pressure, with higher velocities from greater differences. Wind is characterized by its speed and direction (Boeker & van Grondell, 2011) and thus wind velocity can be treated as a vector. Various existing methods simulate wind. The simplest is to have consistent application
of a wind with a constant velocity throughout the virtual environment. With this approach, branch motion can be demonstrated as spring-like movements, as first modeled by Ono (1997) and enhanced by Sakaguchi & Ohya (1999).

More complex wind modeling approaches include gusts (Ao, Wu, & Zhou, 2009; Feng, Chen, Yan, & Wu, 2001), Navier-Stokes equations (Oliapuram & Kumar, 2010; Perbet & Cani, March 2001), particle based fluid simulation (Yang, Huang, & Wu, 2011; Oliapuram & Kumar, 2010), noise (Habel, Kusternig, & Wimmer, 2009; Ota, et al., 2003), and incorporate real world data (Diener, Rodriguez, Baboud, & Reveret, 2008). Recently, wind effects have been expanded to include effects on development and growth (Derzaph & Hamilton, 2013b; Pirk, Nise, Hadrich, Benes, & Deussen, 2014). In this thesis, wind is represented as a simple vector influencing the target dynamic plant model.

Global Form of Environmental Influence - Water and Nutrient Uptake Algorithms

Water and nutrient uptake are required for plant viability. One algorithm specifically focused on simulating the effect of the water availability on root growth (Měch & Prusinkiewicz, 1996). A second algorithm focused on the influence of nutrient availability on the fate of buds (de Reffyre, Cognee, Jaeger, & Traore, 1988). Further research in this area included environmental effects such as drought and floods (Clausnitzer & Hopmans, 1994; Liddel & Hansen, 1993; Kaandorp, 1994; Doussan, Pagès, & Pierret, 2003; Hua & Kang, 2011; Allen, Prusinkiewicz, & DeJong, 2005).
The effects of nutrients on growth have been described by equation 2.14 (Ågren G. I., 2008; Ingestad & Ågren, 1992; Knecht-Billberger, 2006):

\[
g_n = \begin{cases} 
0 & c_n \in [0, c_{n,\text{min}}] \\
 d_n(c_n - c_{n,\text{min}}) & c_n \in [c_{n,\text{min}}, c_{n,\text{opt}}] \\
 d_n(c_{n,\text{min}} - c_{n,\text{opt}}) & c_n \in [c_{n,\text{opt}}, c_{n,\text{tox}}]
\end{cases}
\] (2.14)

where \( g_n \) is the growth rate for the plant species in the presence of nutrient \( n \) in concentration \( c_n \). \( d_n \) is the linear growth rate for the plant species, \( c_{n,\text{min}} \) is the minimum concentration of nutrient \( n \) required for growth, \( c_{n,\text{opt}} \) is the optimum concentration required for growth, and \( c_{n,\text{tox}} \) is the toxic concentration that causes plant decline and death.

The equation 2.14, is used for the nutrient influence on growth described in detail in Chapter 4.

2.1.2 Artistic Dynamic Plant Model Approach

Recall that an artistic dynamic plant model (ADPM) is a dynamic plant model generated by an artistic approach with extensive user involvement using a software package such as Maya. The benefits of utilizing ADPM models can be realized commercially within games, movies, and commercials because they provide unique objects that would not be created otherwise. ADPM focuses on the enhancement of the degree of belief by providing more extensive realism of plants interacting with characters or being present in the backgrounds of games, movies, and commercials. Another benefit of artistic plant modelling is through the production of art pieces, which may allow artistic expression. However,
ADPMs can be cumbersome and only partially realistic for dynamic plant model animation.

An artistic dynamic plant model (ADPM) model is composed of a combination of lines, polygons, and polygon meshes and textures. A **polygon** is a geometric shape that includes a multitude of sides. Polygons range from three sided triangles to a many-sided polygon that is used to represent a circle. A **polygon mesh** is a collection of vertices, edges, and faces that portray a shape. A **texture** is a digital image overlaid onto a polygon mesh. The process for a modeller to produce an artistic dynamic plant model is to examine the plant of choice, divide it into component self-similar parts, and portray those parts in polygon forms. Usually these polygon forms are high-order polygons because complex curved shapes are wanted. Once the high-order polygons designed, they are used to form the final shape. Leaves, stems, flowers, roots, plants, and complete scenes can be created in this fashion. In addition to composition, ADPM is based on a modeller’s interpretation of the plant, utilizing artistic skills such as shape, shading, placement, sizing and perspective. To portray plants realistically from scratch is a very complex and tedious undertaking.

To simplify the creation of artistic dynamic plant models, several libraries of plant models are provided by commercial modelling software. The flower pictured in Figure 12 and Figure 13 was generated in Maya. The petals are subdivision surfaces, the leaves are polygon cubes, the stem is a polygon cylinder, and the pistil is composed of two polygon spheres and one polygon cylinder. Each anther is a polygon cylinder and a polygon sphere. All the
shapes had textures added using lambert shading and an image file generated from a picture of a lily produced by a FinePix digital camera. The shapes are combined together to produce the overall image. The images in Figure 12 and Figure 13 are not realistic. Further subdivision, positioning, and changes in shape are necessary to increase realism. Creating a realistic model of a flower using the ADPM approach can take days if not weeks.

A dynamic plant model can be created with the ADPM approach without knowledge of plant functions; as a result, realism may not be achieved. To achieve more realistic dynamic plant models, the SDPM approach can be utilized in conjunction with the ADPM approach.

A rudimentary ADPM approach using textures and triangle meshes was used in the demonstrations described in Chapter 4.

2.1.3Digitally Scanned Dynamic Plant Model Approach

The digitally scanned dynamic plant model (DDPM) approach uses images of a plant to produce a plant model either in 2D or 3D. The images can be captured through the use of 2D digitization from a camera or flat-bed scanner or a 3D digitization from laser equipment (Kaminuma, et al., 2004), cameras (Shlyakhter, Rozenoer, Dorsey, & Teller, 2001; Ishizuka, Tanabata, Takano, & Shinomura, 2005), electron microscopes (Dumais & Kwiatlowska, 2002; Kwiatkowska, 2004; Kwiatkowska & Dumais, 2003), x-rays, micro-radar, microchips (Room, Hanan, & Prusinkiewicz, 1996) or other equipment. The resulting plant model may be two dimensional (2D), two and a half dimensional
(2.5D), or three dimensional (3D). A DDPM adds realism to commercial endeavors, such as computer gaming, movies, and commercials, while minimizing the time required to create the appearance of plant surfaces that duplicate nature in detail.

Billboarding (Decoret, Durand, Sillion, & Dorsey, 2003), a DDPM technique, uses a 2D plant image obtained with a camera. The image is used as an overlay to a small planar surface, called a billboard. This technique provides one static view of the plant. The benefits to this approach are the ease of duplicating the plant with one camera image, and the low demand for computer resources. However, the image is static. Since static images can interfere with a player’s immersion in a virtual world, billboarding is no longer used to represent plants in complex games. It is still used for simplistic smartphone and tablet games due to the resource limitations of these devices.

View dependent sprites, another DDPM approach, provides for variation of plant images as an avatar approaches the plant. An avatar is a character within a game that is manipulated by a person. View dependent sprites are a series of billboards generated from pictures taken of the real plant and are selected to appear based upon the avatar’s distance from the plant. The disadvantage of this approach is that the plant model is static. As well a large database of sprites is required to provide enough views of the plant model for various distances. In addition, the image lighting does not change so the image does not appear realistic from the avatar’s viewpoint.
A more recent approach, a *billboard cloud* (Lacewell, 2006; Garcia, Sebert, & Szirmay-Kalos, 2005; Decoret, Durand, Sillion, & Dorsey, 2003), divides 3D images into a series of planes with texture and transparency maps. This technique solves the lighting problem observed with view dependent sprites. However, a limitation exists dependent upon the screen size due to resource inefficiencies where rendering a pixel multiple times occurs due to interlaced billboards.

In the 2.5D approach, the textures of the plants are arranged in a cross-pattern or around a simple shape. A *cross pattern* is where two planes are arranged along axes. The second method arranges planes outside of a simple shape such as a cube or a prism. A texture is either generated using a modeling package or captured by a camera image of an actual plant. The reduced demand on computer resources makes 2.5D suitable for placing plants in large landscapes in games. Games such as World of Warcraft, Skyrim, and Rift use the 2.5D approach. The disadvantage is that a plant does not change in structure based on its motion or the viewer’s perspective. When viewed from nearby, the plant model is obviously a collection of arranged image planes rather than a 3D plant model.

The 3D DDPM approach uses a volumetric approach, a particle systems approach, or a point-based approach. A volumetric approach (Levoy, 1988; Drebin, Carpenter, & Hanrahan, 1988; Reche-Martinez, 2004) is based on a *volume*, which is a series of vertices in a 3D grid format. Such an approach adds an alpha (α) value for each pixel point (x, y, z) that is based on the opacity.
**Opacity** is the level of transparency of a pixel; it ranges from transparent (0) to opaque (1). A billboard is attached to each vertex in the volume. The billboards are rendered in a back to front order.

In contrast, the particle system approach (Reeves, 1983; Reeves & Blau, 1985; Neubert, Franken, & Deussen, 2007), generates from the images a voxel model with density values for the number of particles in a certain area. Then, vectors are calculated, determining the direction fields for the particles to flow along. Finally, particle simulation is applied along the vectors using the density calculations to generate the 3D geometry (plant branches) from the captured images. The particles are then interpolated to generate occluded branches and other organs of the plant. Motion can be captured through the movements of the particle when affected by external stimuli.

Lastly, a point-based approach (Deussen, Colditz, Stamminger, & Drettakis, 2002; Tan, Zeng, Wang, Kang, & Quan, 2007) generates a 3D point model from images, using that plant model to interpolate the occluded branches and organs. Once the 3D structure is generated, the plant image is segmented into smaller images and overlaid on the final plant model during rendering.

The DDPM approach is not suitable for animating realistic dynamic plant model growth. The approach requires images of actual plants and time to devise suitable models. Animating plant growth and colour change would require a large collection of textures for each stage of the plant growth and for
each nutrient colour change. Since colour changes can occur in a wide variety of orders, storing images for all possible permutations would be impractical.

DDPM approaches were not utilized in the demonstrations described in Chapter 4. They are described here to give a more comprehensive view of plant animation techniques.

2.2 Ontologies

An **ontology** is a way to provide semantic classification of entities in a given knowledge domain (Walls et al., 2012). An ontology describes the types or classes of objects (entities) in a domain and the relationships among them. An ontology provides standardized definitions for the terms used to describe these classes and it defines the logical relationships among these terms. An ontology also includes a hierarchical structure where each object may or may not have a dependent object. If the object does not have a dependent object, it is referred to as a terminal vertex (or last leaf node).

An ontology for a domain is generated by precisely defining terms called **concepts** to represent classes of objects and relationships amongst those concepts. For example, ontologies are used to categorize web sites, information about plants, and man-made products. An ontology is developed similarly to developing a data structure for programming. The domain corresponds to the relevant data irrespective of the operations that manipulate the data. For example, a school ontology might have the concepts for a building, floor, classroom, classroom size, teacher, student, program, degree, and class. One
relationship in this ontology is “contains” which in fact defines a hierarchy among entities from various domains. For example, building contains floor and floor contains classroom. Another relationship is the “is assigned to a” relationship, which describes a teacher being assigned to a class.

The Plant Ontology (PO) classifies plant anatomy and plant development stages (Walls R. L., et al., 2012; Avraham, 2008; Plant Ontology Home, 2016). The Gene Ontology (GO) considers molecular functioning, biological processes, and cellular components (Consortium G. O., 2009) that are important for plant physiology (Ashburner, et al., 2000; Cooper L. W., 2012). The Plant Trait Ontology (TO) considers plant traits and genotypes (Pankaj, 2013; Jaiswal, et al., 2002), and thus extends both the PO and GO ontologies (Plant Trait Ontology, 2016). Lastly, the Plant Environment Ontology (EO) considers environmental influences (Pankaj, 2013) on the plant that produce physiological responses. At the time of the writing of this thesis, no animation ontology was available. A proposed Animation Ontology based on the work of Chu & Li (2012) is incorporated into the PLANI framework, although is not the focus of the thesis.

The four plant ontologies selected are only a subset of the available ontologies provided by the open biological and biomedical ontologies (OBO) foundry (2016). The PO, TO, and EO ontologies were chosen because they are specific to plants. The GO was chosen because it applies to all organisms. GO describes concepts related to the genes of the organism, but it does not describe structure, protein-protein interaction or environment effects (Consortium G. O.,
PO describes the function and structure of plants. TO adds detailed traits information about PO. EO represents the environmental conditions in which the plant resides.

The selected ontologies conform to the ten principles and numerous standards provided by OBO. The first principle is “The ontology must be open and available to be used”, the second principle is “The ontology is expressed in a common formal language”, and the other principles can be found on the OBO website. The OBO project is a subset of the Forms of Life project now called the Tree of Life web project (Tree of Life Project, 2014). The details of the ontologies (excluding animation) are provided by the Gramene web site (Pankaj, 2013) for convenience. In addition, there are instructions and background information provided in the FAQs (Gramene, FAQ, 2016b).

The five subsections of this section present descriptions of the ontologies specific to plants that are incorporated in the PLANI framework. Plant Ontology (PO) is described in section 2.2.1, Plant Trait Ontology (TO) is described in Section 2.2.2, Gene Ontology (GO) is described in Section 2.2.3, the Plant Environment Ontology (EO) in Section 2.2.4, and, lastly, Animation Ontology (AO) in Section 2.2.5.

2.2.1 The Plant Ontology

The Plant Ontology (PO) is available on the internet at www.plantontology.org. As stated on their web site, PO includes plant anatomical and plant structure development stage entities (Consortium P. O., 1999 - 2017).
2016a), which can be interpolated when constructing an animation, as outlined in Chapter 3. PO is a bioinformatics effort to standardize the nomenclature, definitions, synonyms and relations of various terms (Consortium P. O., 2016b).

There is a process to submit information for inclusion through the web site such that consideration for alterations, additions, and deletions are controlled. As illustrated in Figure 14, a multitude of relationships is described. Ontology formalizes relationship descriptions, and PO has the following standard relationships for any new term/entity addition (Plant Ontology, 2016):

- **A is_a B** indicates every instance of A is an instance of B. For example, a *stem* is a *shoot axis*.
- **A part_of B** indicates every instance of A is part of some instance of B. For example, embryo development is part of seed development.
- **A develops_from B** indicates A and B are either cells or structures made of cells and the majority of the cell lineage of A can be traced to B.
- **A has_part B** indicates every instance of A has some instance of B as a part. This leads to partial relationships where B is not necessarily part of A.
- **A adjacent_to B** indicates every instance of A is adjacent to some instance of B.
- **A participates_in B** indicates every instance of plant anatomy A participates in an instance of plant structure development B.
- **A derives_by_manipulation_from B** indicates A is an in vitro plant structure created by humans, A exists at a point in time later than B, and A inherited a biologically significant portion of its matter from B.
- **A has_participant B** indicates every domain instance of A has some range instance of B as a participant. For example, every seed development has as a seed participant.
- **A located_in B** indicates if A resides in B then A is located in B. Very few such relationships exist in the Plant Ontology because many terms utilize the part_of relationship.
- **A preceded_by B** indicates a one-way time relationship between A and B where A precedes B.

Terms/entities have an established metadata format, which is given below and shown in Figure 15 (Planteome: Plant Anatomical Entity, 2016):

- **Accession**: indicates the UNIQUE key of format PO:#######.
- **Name**: indicates the name of the entity.
- **Ontology**: indicates whether the term is plant anatomy or plant structure development.
- **Synonyms**: are classified as exact and can include different languages.
- **Alternate IDs**: indicates other ontology identifiers.
- **Definition** describes the term.
• **Comment** provides additional information to clarify, support or expand on the term definition.

• **Subset** provides information if the term is a subset of another ontology term.

• **Related** provides any other linkages and references supporting the term.

• **Feedback** provides contact information for cases when changes to the term may be required.

• **Inferred Tree View tab** provides the hierarchical and functional relationships in a tree view.

• **Neighborhood** provides existing relationships to other Plant Ontology terms.

• **Graph Views** provides a picture related to the term and tells where it resides in the ontology hierarchy.

• **Mappings** provide links to meeting minutes on the suggested changes to the term.

The collection of PO terms is contained in a database with a defined schema and structure that can be examined at this site. This list of terms and relationships is used as a foundation for the FORM and FUNCTION domains in the PLANI framework as will be described in Chapter 3.

The PO has a hierarchy where each object is related to another object using the part_of relationship is its subordinate, as shown in Figure 14. Hierarchy can guide the designer when specifying the sequence in which
biological organs emerge. It also provides the designer with a convenient reminder of the connections amongst objects, i.e. a leaf is part_of a stem which is part_of a branch.

2.2.2 The Plant Trait Ontology

The specific difference between the PO and the TO can be understood through their definition: “Plant trait describes phenotypic traits in plants where each trait is a distinguishable feature, characteristic, quality or phenotype feature of a developing or mature plant” (OBO Foundry, 2016). The relationship definitions for PO and TO are the same.

The TO website has the same format as the PO although they are currently at differing versions of the database (Gramene, Plant Trait, 2016). Figure 16 illustrates the structure of the database and provides the definition of the TO. The metadata format is given below:

- **Term Name**: provides the name of the term
- **Term Accession**: indicates the UNIQUE key of format TO:#######.
- **Aspect**: indicates whether the term is plant anatomy or plant structure development
- **Synonyms**: are classified as exact and can include different languages.
- **Definition**: describes the term.
- **Comment** provides additional information to clarify, support or expand on the term definition.
• **Derivation**: provides the hierarchical and functional relationships in a tree view.

• **Database Cross-References** provides any other linkages and references supporting the term.

• **Children**: provides the lower hierarchical level of terms that are associated to, or children of the term.

The terms for anatomy and plant physiology traits in TO are used as a foundation for the FORM and FUNCTION domains in the PLANI framework, as will be described in Chapter 3.

### 2.2.3 Gene Ontology

The gene ontology (GO) is available at geneontology.org and supported by the Gene Ontology consortium (Consortium G. O., 2016a), (Consortium G. O., 2016b). The consortium is composed of professionals in all fields related to earth’s organisms and gene studies. The project has three structured, controlled vocabularies associated with molecular functioning, biological processes and cellular components. The other ontologies were excluded at this time as the focus was on the macro level rather than the micro level of plants. That does not exclude the inclusion of the other ontologies as required. As shown in Figure 17 (Gramene, 2016a), (Gramene, 2016b), the format for terms is the same as that of PO and TO.
2.2.4 The Plant Environmental Ontology

The Plant Environment Ontology (EO) describes the environment that a plant resides in and includes the abiotic and biotic environments (Gramene, Plant Ontology Searches, 2016). It is used in PLANI for the ENVIRONMENT domain.

Some of the hierarchy structure is as follows:

Abiotic environment (EO:0007181)

  Gaseous environment (EO:0007023)

  Seasonal environment (EO:0007027) ....

Study type (EO:0007231)

  Green house study (EO:0007248)

  Laboratory study (EO:0007255)

Biotic environment (EO:0007357)

  Pest/pathogen/animal/plant environment (EO:0007124)

  Unknown environment (EO:0007403)

The EO is defined similarly to PO and TO. The metadata format is given below:

- **Term Name**: provides the name of the term
- **Term Accession**: indicates the UNIQUE key of format TO:#######.
• **Aspect**: indicates whether the term is plant anatomy or plant structure development

• **Synonyms**: are classified as exact and can include different languages.

• **Definition** describes the term.

• **Comment** provides additional information to clarify, support or expand on the term definition.

• **Derivation**: provides the hierarchical and functional relationships in a tree view.

• **Database Cross-References** provides any other linkages and references supporting the term.

• **Children**: provides the lower hierarchical level of terms that are associated to, or children of, the term.

2.2.5 Animation Ontology

An animation ontology is provided by (Chu & Li, 2012). It includes the following classes and definitions:

• **IMWorld** is the root of the virtual world containing the world information and has the following objects
  
  o **WorldInf** is a file of the virtual world.
  
  o **WorldObject** is all the virtual objects within the world containing attributes such as name, tag, baseLevel and height
  
  o **Transform**
▪ Translation
▪ Rotation
▪ Scale
  o Ground defines the boundary of the world.
  o GeometryInfo defines the geometric shapes
▪ Polygon defines the shape of the Approximation2D
  • hotPosition is the foci of interest on World object
  • Approximation2D is a polygon that can be used to define 2D approximation of obstacles in the environment for path planner
  • Avatar is a character that can be controlled by a real user or a computer program.
    o Behavior defines the procedure in order to generate the desired animation.
    o Position defines the current position of the avatar.

This ontology is not comprehensive and does not have the term specificity of the plant ontologies. Therefore, definitions, extensions and naming changes were required to ensure that the PLANI framework could specify relevant information about several terms/entities important for animation, such as Viewpoint, Light, Time and Material. These changes will be described in Chapter 3.
Chapter 3  The PLANI Plant Animation Framework

This thesis proposes an approach to creating dynamic plant models using a new conceptual framework called the PLANI Plant Animation Framework (Derzaph & Hamilton, 2017). PLANI provides a way to use ontologies to guide the design of dynamic plant models and related algorithms and programs. Each stage of the design is accomplished by a specific role. The *designer role* is responsible for making decisions about the inclusion of the objects in each domain. The *coder role* is responsible for writing algorithms and programs that implement the chosen objects. The *tester role* is responsible for ensuring the completed code runs correctly. The *biologist role* is responsible for ensuring the consistency of the objects to biological principles.

The PLANI framework includes four domains: FORM (which represents the physical structure of a plant), FUNCTION (which represents the physiological function of a plant), ENVIRONMENT (which represents all environmental influences on a plant), and VIRTUAL WORLD (which represents all relevant computer animation factors for the dynamic plant model). A *domain* in PLANI is a collection of objects relevant to portraying a component of a dynamic plant model. Where possible, the framework adopts the ontologies described in Chapter 2 to organize information about plants and to describe the FORM, FUNCTION, and ENVIRONMENT domains and about animation to describe the VIRTUAL WORLD. By guiding the designer through a staged process using ontologies to design a dynamic plant model, PLANI ensures fidelity to biological
constraints for simulations while allowing violations of biological constraints for creative plant animation at the discretion of the designer.

Throughout this thesis, the following formats are utilized to provide information about whether the concept is a domain, object, or property:

- Domains are indicated by all capitals, e.g. FORM domain.
- Objects are indicated by mixed case and italics, e.g. Growth object.
- Properties are indicated by lower case and italics, e.g. limiting nutrient property.

The remainder of this chapter is organized into three sections: Section 3.1 describes the PLANI framework, Section 3.2 provides a design example, and Section 3.3 provides implementation factors that influence the design and coding stages.

3.1 The PLANI Framework

The PLANI framework encompasses four domains: FORM, FUNCTION, ENVIRONMENT, and VIRTUAL WORLD, as shown in Figure 18 and Figure 19. These four domains enable alignment with the current ontological classifications for plants, including the PO and TO for FORM, GO, PO, TO for FUNCTION, and EO for ENVIRONMENT, as previously described in Section 2.2.4. Each domain is further subdivided into objects. An object is a component directly linked to a term in an ontology; e.g., the Growth (GO:0040007) term in the GO is represented as the Growth object in PLANI. Lastly, an object is composed of properties. A property is the smallest described characteristic of a plant that
can be used in an animation. A property is directly linked to parameters within a computer program and also to a terminal vertex in the ontology. For example, the Stem object has a stem elongation rate property.

The PLANI framework also includes four stages to enable designers and coders of dynamic plant models to follow a predefined sequence. By appropriate following of the stages, the designer can guide the coder to instantiate dynamic plant models by using existing algorithms or developing new algorithms. In addition, the designer can consult with biology experts using the ontological semantics to ensure alignment to biological concepts and simplify communication. These stages are described as:

- Stage 1: Selection of domains for the biological simulation or animation.
- Stage 2: Selection of objects from the domains selected in Stage 1.
- Stage 3: Selection or design of algorithms for the objects selected in Stage 2.
- Stage 4: Coding of the algorithms for inclusion in the animation software.

3.1.1 Stage 1: Domain Selection

During Stage 1, shown in Figure 18(a), the designer chooses a subset of the four domains. In all cases the minimum set of selected domains is the FORM and VIRTUAL WORLD. This minimal selection enables the generation of animated plant in a virtual world.
3.1.2 Stage 2: Object Selection

As part of Stage 2 shown in Figure 18(a), to design a dynamic plant model using the framework, one proceeds as follows. First, select objects in the FORM domain to match the desired structure for the plant. Secondly, select the objects in the FUNCTION domain to provide the desired behaviour for the plant. Next, select objects in the ENVIRONMENT domain that capture the desired environmental influences on the plant. Lastly, select objects in the VIRTUAL WORLD domain that will enable the appropriate animation of the dynamic plant model using all the selected objects. This order is required to ensure that the correspondence among objects between domains is properly considered. If the designer finds that there are missing objects in the intended dynamic plant model, backtracking to previous domains is permitted until a dynamic plant model with the desired features is obtained.

Let us consider the stage 2 design-time process for designing a simple, example plant that grows and responds to limitations in nutrients. The selections from the FORM domain could include the Stem object, the Leaf object, and the Leaf Colour object along with their properties height, shape, and colour, respectively. From the FUNCTION domain, the selections could include: the Growth object, which contains the growth rate property (units in mm), and the Response to Nutrient object, which contains the limiting nutrient property. Next the selections from the ENVIRONMENT domain could include the Seasonal object, which contains the season property, and the Nutrient Regimen object, which contains the nutrient concentration property for each nutrient. Linkages
between the domains could then be provided by the biologist through discussions with the designer. There are some indications in the ontologies of possible linkages, but biological knowledge is needed to specify the linkages among the objects.

The next consideration during Stage 2 is the runtime process started by the exchange of properties between the four domains, progressing in sequence from ENVIRONMENT to FUNCTION to FORM all supported by the VIRTUAL WORLD. As part of this Stage 2, this runtime process that will occur when the dynamic plant model is used should be considered at a high level. It must be understood that the processing proceeds in the opposite order from that used in design, as illustrated in Figure 18(b). The underlying assumption about runtime processing is that objects in the ENVIRONMENT domain influence objects in the FUNCTION domain, which then influence objects in the FORM domain. These influences are implemented through objects in the VIRTUAL WORLD domain. At runtime, the virtual world is generated by the objects in the VIRTUAL WORLD domain, including the Light object, the Viewpoint object, and the Topography object. The runtime process contributes to the animation of the dynamic plant model by passing property values amongst the corresponding domain objects. It starts by providing values to the ENVIRONMENT domain objects, resulting in setting environment properties, which are then sent to the linked FUNCTION domain objects. The FUNCTION domain objects receive the environment properties and set function properties, which are then sent to the linked FORM domain objects. The FORM domain objects receive function properties and set
form properties based on the function properties, which then produce the animation.

The designer should ensure that Stage 3 and 4, which respectively involve the design and coding of algorithms, will process the objects in the right sequence through the domains. As an example of a runtime process that includes consideration of Stage 3 design, suppose the date property of the Time object of the VIRTUAL WORLD is set to August 1, 2014. The value date property is sent to the Seasonal object in the ENVIRONMENT domain. Given this value for the date property, the summer season property of the Seasonal object is set to “Summer”. Since the Seasonal object in the ENVIRONMENT domain is linked to the Growth object in the FUNCTION domain by the summer season property, the value of this property is sent to the Growth object. The Growth object determines the growth rate to be 0.01, i.e. 1 percent per day, given the season value of “Summer”. Since the Growth object in the FUNCTION domain is linked to the Leaf object in the FORM domain via the growth rate property, the growth rate value of 0.01 is sent to the Leaf object. Then the Leaf object makes any appropriate changes to its property values, such as leaf length, leaf width, leaf thickness, and leaf elongation rate.

As another example of a runtime process that includes consideration of Stage 3 design, suppose the concentration of boron was changed in the soil near the plant, i.e., the Micronutrient Regimen object in the ENVIRONMENT domain has the boron regimen property changed. The value of this property is sent to the Response to Nutrient object in the FUNCTION domain. Then this
object sends the *boron regimen* value to the appropriate plant object (such as a *Leaf* or *Stem* object) in the FORM domain. This object sets the value of a property that controls colour (such as the *leaf colour* or *stem colour* property), which changes the colour of the plant organ based on the *boron regimen* value.

Stage 2 object selection is described in four subsections, divided by the relevant domains: FORM, FUNCTION, ENVIRONMENT and VIRTUAL WORLD. Some considerations for each of the FORM, FUNCTION and ENVIRONMENT domains, with respect to the VIRTUAL WORLD domain, is required to ensure the plant is properly viewed by an end-user are also described. The FORM domain object selection depends on the target plant. For example, if the target plant is a rose, the FORM domain objects for the rose are a cylindrical stem, an oval leaflet shape with serrated edges as part of a compound leaf, a thorn, and a rose flower governed by a certain size with respect to the VIRTUAL WORLD domain viewpoint the rose would be positioned within the viewpoint range.

Stage 3 and 4 of PLANI, designing and coding of plant objects, is described in Chapter 4 in detail with an overview in this chapter in section 3.1.3 and section 3.1.4 respectively.

Stage 2 sets up the selection for the Stage 3 design through examination of the objects required in each of the domains as described in the following sections.
3.1.2.1 Stage 2 Selections in the FORM Domain

The FORM domain contains information related to plant morphology: the shapes, textures, and colours of the components of the plant. When selecting objects, the designer should consider both PO and TO objects, depending on the amount of detail required. PO is the parent of TO with respect to object inheritance. For example, the leaf sheath (PO:0020104) object, which describes leaf sheaths, has a leaf sheath trait sub-object TO:0000754, which adds traits to the leaf sheath, such as diameter (TO:0000642), length (TO:0002689), width (TO:0002721), and colour (TO:0002724). So, if PO:002014 is selected there will be less detail than if both PO:002014 and TO:0000754 are selected.

To produce an animation of a dynamic plant model containing leaves, the designer selects FORM as the domain. In Stage 2 the designer consults the PO and TO ontologies to select the relevant objects. Figure 20 represents the selection of the Leaf object, leaf width, leaf length, leaf thickness, leaf colour, and leaf shape properties of the FORM domain, as included in the PO and TO ontologies.

For a second example for FORM selection, Figure 21 represents the selection of the Stem object and the properties of the object, stem length, stem colour, stem diameter, and stem strength included in the PO and the TO for Stage 2.

For a third example, Figure 22 shows the hierarchy and selection of the Apical meristem object that is required to enable the FUNCTION domain object
Growth. This FORM object is not explicitly selected because it represents the area where the growth of a plant occurs at the end of each stem. As a result, the apical meristem is part of the Stem object in the FORM domain.

3.1.2.2 Stage 2 Selections in the FUNCTION Domain

The FUNCTION domain consists of objects representing plant motion and plant biological processes, such as plant development (growth, seasonal organ appearance), metabolic processes, and nutrient transport. The ontologies relevant to the FUNCTION domain are GO, TO and PO. As with the FORM domain, PO is the parent of TO. It is also the parent of GO, with respect to growth and development. For example, the seed development stage (PO: 0001170) in PO is inherited by TO where specification with respect to the seed development trait (TO: 0000653) provides seed maturation, days to maturity, and germination trait properties that influence seed germination. The seed development stage is also linked to GO (GO: 0048316) through TO inheritance, where the ontology represents the process as the outcome of the seed changes over time through the Growth object.

Figure 23 represents the selection of the Growth object from GO. One property within the Growth object is the growth rate for a particular plant species. This property is sent to the Stem object in the FORM domain, where the property of stem elongation rate receives the new growth rate and changes the other stem properties accordingly. The selected Stem and Growth objects would then require coding where the code parameters represent the properties for each object.
The FUNCTION domain is related to the ENVIRONMENT domain in the same fashion as the FORM domain is related to the FUNCTION domain. Properties from the objects in the ENVIRONMENT domain are received by objects in the FUNCTION domain. For example, growth is influenced by the season property from the ENVIRONMENT domain, as previously described.

3.1.2.3 Stage 2 Selections in the ENVIRONMENT Domain

The ENVIRONMENT domain contains objects and properties corresponding to all terms defined by EO (Gramene, Plant Ontology Searches, 2016) as illustrated in Figure 24. EO includes seasonal changes, radiation regime (sunlight), temperature, water, gaseous regimen, physical environment (wind, gravity), and chemical regimen (soil nutrients). A property in the ENVIRONMENT domain can affect objects and properties in the FUNCTION domain. For example, selecting the season environment property which will be used to affect the FUNCTION domain by activating or deactivating the Growth object, which in turn affects the appearance, size or nonappearance of the objects in the FORM domain.

3.1.2.4 Stage 2: Selections in the VIRTUAL WORLD Domain

The VIRTUAL WORLD domain consists of the objects used to generate an animation, including the Light, Viewpoint, and Topology objects at the minimum. This domain enables plants to be generated for a variety of virtual worlds using the same information from the FORM and FUNCTION domains. Since no formal ontology is currently available for virtual world animation, we adapted a structure from Chu and Li (2012) that was discussed in Section 2.2.5.
Their structure contains WorldObject, Transform, Translation, Rotation, Scale, Approximation2D, Polygon, Ground, GeometryInfo, HotPosition, IMWorld, and WorldInf (world information) objects. The IMWorld object is at the root of their structure; we replaced this idea by creating the VIRTUAL WORLD domain in our ontology and placing the other objects from their structure in this domain. We renamed Ground to Topology because the latter term is more description of the many possible topological environments that may be specified in a virtual world, e.g., lakes, hilly terrain, plains, or flowerpots in building interiors. Besides the objects considered by Chu and Li, we added objects for Viewpoint, Light, Time, and Material because we regard them as relevant to producing animations in a virtual world. Viewpoint specifies the position from which the animation is viewed. Light specifies the position and direction of a light. Time is specified through three different properties. One for real-time frame rate t measured in milliseconds, one for slow simulated time T measured in days and lastly one for fast simulated time ts measured in seconds. Material specifies properties of a surface, including colours (e.g., ambient, diffuse, and specular) and normal vectors. The combination of the viewpoint and the position of the dynamic plant model in the topology determine the location of the plant in the scene. Lastly, we added the background colour property in the WorldInf for the scene.

As an example, suppose that at design time the selection of the Position, Viewpoint, and Light objects in the VIRTUAL WORLD domain provides the minimum information for a plant to appear to the end-user. The coding assigns values to properties of the selected objects either directly or by passing values
from other objects. Suppose the $x$, $y$, and $z$ properties of the $Position$ object are assigned 0, 0, and 0, respectively. These values are passed to the $Stem$ object in the FORM domain to ensure proper positioning of the plant in the virtual world. As well, these values are passed to the $look$ $at$ property of the $Viewpoint$ object to ensure that the plant will be in focus. Lastly, these values are also passed to the $lighting$ $aim$ $direction$ property in the $Light$ object to provide the appropriate light on the plant.

3.1.3 Stage 3: Designing the Algorithms

The process within the PLANI framework for designing an algorithm proceeds as follows:

- Step 1: Using the objects selected in stage 2
  - Step 1a: Identify any constraints,
  - Step 1b: Determine the relationships between objects that are in separate domains. For example, $Leaf$ and $Stem$ objects in the FORM domain, $Locomotion$ object in the FUNCTION domain, and the $Wind$ object in the ENVIROMENT domain.
  - Step 1c: Add additional objects to ensure proper relationships.
  - Step 1d: Repeat Steps 1b and 1c until all relationships are identified.
  - Step 1e: Search in the available literature for an existing algorithm for each object and its relationship(s).
- Step 2: If no existing algorithm exists:
- Step 2a: Search the biological literature for a summary of the object or objects and their relationships
- Step 2b: Design the algorithm

During the first step, there are a great number of algorithms to select, as described in section 2.1. It is during this stage that designing the proper linkages to the other domains needs to be solidified rather than considered as was done in Stage 2. **Solidified** means to explicitly design and link any dependent properties between the objects within the domains for the purposes of supporting coding during Stage 4. For an interaction example, if there is a *growth rate* property in the FUNCTION domain and a *Wind* object selected in the ENVIRONMENT domain, there must be a corresponding *wind* property to provide parameters to the *growth rate* property. If there is not a selected *wind* property, then the property needs to be found. This means the designer must return to Stage 2 to select a corresponding property. When the referenced ontology has a gap in the content, it is up to the designer to add a corresponding property and send an update to the ontology through the request for inclusion process described previously in section 3.1.2.

Next consider the example started in Section 3.1.2 Stage 2 which was to define a simple plant. During Stage 3, which is to find some algorithms that would be relevant to implement the selected objects. For example, if a simple plant containing one stem and one leaf then an algorithm to generate the stem and the leaf is required. In addition, a way to generate a variety of plants would be advantageous. This would lead to the selection of the L-system algorithm.
3.1.4 Stage 4: Coding the Algorithms

It is at the coding step that several considerations need to be incorporated, such as levels of detail and linkage to the other domains. It is recommended that complete object selection across all domains are made prior to coding any of these objects. This should be supported by Stages 2 and 3 of the framework. This approach allows opportunity to consider any cross-domain aspects with respect to the properties required to complete the model. For instance, connections between the domains will occur through the coding parameters that represent the object properties. For example, the Leaf and Stem objects in the FORM domain require the properties of length, width, elongation rate, and colour. These objects are represented in the code as the parameters: length, width, elongation rate, and colour. By performing selection from all domains before coding, the connection from these objects in the FORM domain to the Growth and Response to Nutrient objects in the FUNCTION domain is noticed before coding. Advanced notice of this relationship ensures that proper coding is performed to smoothly send properties from the FUNCTION domain to the FORM domain at runtime.

Consider Stage 4 the code required to generate the selections in the FORM and VIRTUAL WORLD domains for a simple plant containing one stem and one leaf in a simple virtual world with a designated position for the plant and one light. For simplicity, there is no motion of the plant from the FUNCTION domain and no influence of external factors from the ENVIRONMENT domain in this example. In practice existing code is reused wherever possible: Here, for
simplicity, we assume that new code is being written. Code is required for: (a) the virtual world, (b) the L-System grammar function, (c) the stem shape, and (d) the leaf shape.

When designing the code for the VIRTUAL WORLD, it is important to understand that the objects selected from each domain influence how the plant appears. The code could be written with a starting position in the world space, the plant at the starting position, a viewpoint, and a light at some position aimed at the starting position. At design time, ensuring that the position property of the WorldObject object coincides with look at property of the Viewpoint object is essential to viewing the plant. If these two properties are not the same, the plant may not be visible to the end-user. In addition, if the lighting aim direction property of the Light object is not equal to the position property, then lighting may not be applied to surfaces of the plant, which may cause incorrect colouring.

Next, code is also required for the L-System grammar function. This function is intended to generate three FORM domain objects: Stem, Stem Angle, and Leaf. The code for these objects contains three properties: axiom, production rules, and iteration, which together represent the L-system grammar. An L-string is a string that is generated from the axiom by repeated substitution according to the corresponding production rules. Substitution is continued until the specified number of iterations is reached. Each character in the L-string represents one of the three objects in the FORM domain, i.e. the Stem, Stem Angle, or Leaf. For example, if the “F” character occurs in the L-
string, it represents the Stem object; likewise, the “A” character represents a Leaf object. The Stem Angle is represented by a positive “(+)” or a negative “(-)” angle.

Thirdly, code is required to generate a polygon representing the shape of the stem. More precisely, it generates a Polygon object in the VIRTUAL WORLD domain to represent the Stem object from the FORM domain. This Polygon object has properties for length and diameter, which correspond to the stem length and stem diameter properties of the Stem object in the FORM domain, respectively. The code uses these two properties and the appropriate mathematical formulas to generate a cylinder. In particular, for the “F” character example mentioned previously, the stem shape code generates a cylinder of a certain size, representing a Polygon object, at run time.

Finally, code is required to generate a polygon representing the shape of the leaf. A Polygon object is generated in the VIRTUAL WORLD domain that contains properties for length, width, thickness, and shape, which here correspond to the leaf length, leaf width, leaf thickness, and leaf shape properties of the Leaf object of the FORM domain. At run time, if an “A” character is present in the L-string, this code generates a Polygon object representing a Leaf object.

As demonstrated by the examples just discussed, each object and property selected at design time influences the runtime creation and animation of the dynamic plant model. Omission of any required object or property from any
domain would prevent the construction of the plant at runtime. The appearance of the plant at runtime provides evidence of the designer’s proper selection of objects and the coder’s proper implementation.

Clearly, there is a connection between the complexity (the number of objects selected in the various domains) and the amount of coding required for implementation. For example, if the complete Plant Ontology in the FORM domain is selected for a single dynamic plant model, with the objective of realism, more algorithms are required than if only a few objects are selected. One can expect an increase in the number of selected objects to cause a corresponding increase in design time, coding complexity, and required computer resources. Overall, the available selections provide a range of possible levels of complexity relevant to the target virtual world and the object’s usage across implementations.

3.2 Design Sample

As mentioned above, the generation of a dynamic plant model starts with Stage 1, the selection of domains. Once that stage is complete, Stage 2, the selection of objects and properties in the FORM, FUNCTION, ENVIRONMENT, and VIRTUAL WORLD domains is required to provide the most suitable features for the desired dynamic plant model. This selection is made according to the ontology for each domain and is the base activity applied to all domains for design flexibility to enable simulations, video gaming or film objects. The PLANI framework also provides a method of generating a dynamic plant model composed of objects to result in a model (FORM domain) that provides changes
to that model through external (ENVIRONMENT domain) or internal parameters (FUNCTION domain) in a VIRTUAL WORLD domain. The framework also provides the overall pattern for runtime: the domains to include, the subsequent properties, and the computer algorithm criteria required to fulfill the selections. The extent of design time in Stage 3 depends on the number of selections made in Stages 1 and 2. Overall, the domain selection, object selection: design, and coding stages of the PLANI framework provide the diversity of plants that may be observed by the viewer.

Consider an expanded example that uses the framework for an animation of a lily. Figure 19 shows a relevant subset of selected domain objects with ontology numbering derived from the current ontological hierarchy for a dynamic plant model of a lily (Walls R. L., et al., 2012; Buskiewich, et al., 2002). The sequence to design the model is as follows:

Stage 1: All the domains for this design are selected.

Stage 2: The VIRTUAL WORLD domain selection includes Light, Viewpoint, 2D Conversion of Polygons, Topography, Position of the plant in the world (World Object), Time, and any Transformations required for the motion of the plant. Secondly, the FORM domain objects include Branch (PO: 0025073), Stem (PO: 0009047), Petiole (PO: 0020038), and so forth. Note, there exists an inheritance between the PO and TO ontologies, where TO is more detailed than PO. Third, the FUNCTION domain objects include: Growth (GO: 0040007), Locomotion (GO: 00040011), Response To Stimulus (GO: 00050896), Shoot
System Development (GO: 0048367), Shoot System Development Stage (PO: 0025527), Leaf Development (GO: 0048366), Leaf Development Stage (PO: 0001050), Bud Development Stage (PO:00225528), Fruit Development (GO: 0010154), Fruit Development Stage (PO: 001002), Fruit Ripening (GO:0009835), Seed Development Stage (PO: 0001170), Root Development Stage (PO: 0007520), and Flower Development Stage (PO: 0007615). As with the previous example in the FORM domain there is inheritance between GO and PO where GO is the more detailed class than PO. GO considers changes that occur in the plant, while PO considers the external changes that affect the plant. Finally, the ENVIRONMENT domain objects include: Seasonal (EO:0007027), Radiation Regimen (sun) (EO:007151), Temperature (EO:0007175), Available Water (EO:0007198), Gaseous Regimen (available oxygen and carbon dioxide) (EO:0007023), Nutrient Regimen (all available nutrients) (EO:0007241), Wind (EO:0007382), Gravity (EO:0007146), and Pathogens (EO:0007124). The domain objects can be organized as illustrated in Figure 25.

Stage 3 includes finding algorithms or designing of the corresponding algorithms to implement a lily. The details will be omitted because Chapter 4 contains three examples of designing algorithms.

In Stage 4 the selected algorithms are coded to produce a dynamic plant model of a lily. Again, we omit this stage because Chapter 4 provides three detailed examples on how to utilize the PLANI framework in this context. During run-time, the relevant environment objects, namely Radiation Regimen, Temperature, Available Water, Gaseous Regimen, Nutrient Regimen, Wind,
Gravity, and Pathogens, all affect the corresponding function objects. The Growth of the lily is affected by the Radiation Regimen, Temperature, Available Water, Gaseous Regimen, Nutrient Regimen, and Wind. The Locomotion of the plant is affected by the Wind object from the ENVIRONMENT domain. The Response To Stimulus object of the FUNCTION domain is affected by all the ENVIRONEMNT domain objects. The Shoot System Development is stimulated by the Radiation Regimen, temperature and Available Water as in the, Shoot System Development Stage, Leaf Development, Leaf Development Stage, Bud Development Stage, Fruit Development, Fruit Development Stage, Fruit Ripening, Seed Development Stage, Root Development Stage, and Flower Development Stage. These effects in turn change the appearance of the objects in the FORM domain through adding Flowers, Seeds, Fruit, changing size of Leaves, Branches, Stems, and changing structures of Branches, Stems and Petioles. As described in Section 3.1.2 the flow of information is reverse to that of design time.

The additional example was selected to provide illustration of the flexibility of the framework including the additional objects. By dividing the design and runtime of a dynamic plant model into four domains, the framework enables the designer to plan and select any combination of objects from the online ontologies available. The ontologies provide for the base properties that must be coded to enable a dynamic plant model. The design time is reduced to a process of searching and selecting rather than starting from scratch. In addition, the designer may gain an understanding of the linkages between the domain
objects. Suppose the designer wants the leaf to move. After the selection of a leaf object in the FORM domain, the notion of movement should cause the designer to notice that the locomotion object for a leaf is required in the FUNCTION domain. Overall, the framework provides implementation flexibility, through the selection of objects and algorithms, and consistency, using four domains that have been formally specified by ontologies, and through provision of the four-stage methodology. The next chapter provides three demonstration examples for how the PLANI framework was utilized.

3.3 Implementation Factors

The PLANI framework provides four domains, which enable the designer to link the characteristics of dynamic plant models to current ontological classifications. As previously mentioned, underlying assumption is that objects in the ENVIRONMENT domain influence objects in the FUNCTION domain, which subsequently influence objects in the FORM domain, all of which are supported by the VIRTUAL WORLD. In this section, we discuss four implementation factors, their relation to the selection of objects in domains, and the overall linkage to plant ontologies. Throughout the discussion, we use three scenarios to provide illustrations.

The three scenarios are shown in Table 2. This table shows selected objects for the sequence of domains from ENVIRONMENT to FORM. For each of the scenarios, PLANI has been or could be used to design a dynamic plant model. The table provides evidence that PLANI is well suited to a wide variety of applications, regardless of animation type. The three types of animation are
gaming, biological simulation, and film. **Gaming** is a real-time animation containing an avatar that the user manipulates within a gaming environment to accomplish a series of tasks. **Biological simulation** is based on procedural animation where physical aspects of a plant for example are animated to mimic nature. **Animated film** is produced by artists creating dynamic plant models and animating them to fulfill the vision of the director (Parent, 2012). The scenarios will be discussed in more detail later in this section.

The PLANI framework provides flexibility in implementation because a variety of existing algorithms can be chosen for objects in each domain or new algorithms can be added through coding. Recall that the design process is four-fold: select one or more domains, one or more objects in the selected domain, one or more properties of the selected objects, and one or more algorithms for the objects in the domains. Once a property has been selected, it is also given a specific value, as needed. The determining factors for guiding or restricting these selections are the implementation style (coding or modelling), the type of animation (gaming, biological simulation, or film), the processing requirements (real-time or batch), and the computer resource capacity (CPU, GPU, and storage).

The first factor, implementation style, can be chosen as either coding or modelling. Two potential types of coding are procedurally such as L-systems (Prusinkiewicz, 2004b) and image capture (Wu, Chen, Yan, & Feng, 1999). The resulting code is directly linked to objects, such as Stem and Petiole, in the FORM domain. Alternatively, the implementation style can be chosen as 3D
modelling, with software such as Maya (Maya, 2014). All three alternatives represent valid methods of implementing plant objects in the FORM domain. Likewise, the FUNCTION domain has implementation style choices of either using code to move stems and leaves (Yildiz, 2010) or by presenting a series of images. Similarly, the ENVIRONMENT domain has implementation style choices between coding and presenting a series of background images. Thus, the choice of implementation style guides the selection of objects for the domains.

The second and third factors are best considered together. Selecting the type of animation will influence the processing requirements and therefore the object selections. Typically, a game requires real-time processing. As a result, a game designer should select relatively few objects in the FORM domain and very few objects from the FUNCTION and ENVIRONMENT domains. In contrast, a film requires batch processing to ensure sufficient processing resources to create a high level of detail. Thus, more objects can be selected from the FORM and FUNCTION domains. However, relatively few objects should typically be selected in the ENVIRONMENT domain due to the restricted timeframe of the film. A biological simulation can be implemented with either batch or real-time processing, depending on the desired demonstration. The selection of objects from each domain is limited accordingly.

Lastly, the fourth factor, available computing capacity, needs to be considered during design. For instance, many gaming animations use techniques to reduce GPU and CPU usage. One technique is to restrict the
representation of the *Leaf* and *Branch* objects in the FORM domain to 2.5D images. A 2.5D image is a collection of planes with textures that are placed on polygons, such as cubes or prisms. The 2.5D images require fewer polygons to be generated than 3D models, but they provide the illusion of 3D plants, as observed in games such as *World of Warcraft*, *Rift*, and *Skyrim*. This technique is appropriate for such games because of the unknown computing capacity and network bandwidth available on the user’s machine is unknown. In contrast, realistic film animation requires implementation of complex movements and considerable detail. Generally, detailed objects in the FORM and FUNCTION domains are generated in a frame by frame fashion, by using a data center, such as was done for *Avatar* in 2009 (Rath, 2010). Moreover, a biological simulation typically requires a more detailed animation than a game and a less detailed one than a film. Thus, limitations in available capacity may result in restricting the selection of objects in some domains.

Let us consider the scenarios listed in Table 2 in more detail to illustrate the impact of the four factors on design. For the first scenario, suppose animated plants are required as background elements in a 3D action-adventure game. Since gaming is the selected animation type, the developer is constrained with respect to CPU and GPU usage and must use real-time processing. Since the plants are being treated as elements in the background that enhance the gaming experience, available processor power is limited. To obtain suitable dynamic plant models for the game, the developer can restrict the FORM, FUNCTION, and ENVIRONMENT domains in a variety of ways. One
example of FORM restriction is to use 2.5D images, as previously discussed. An example of FUNCTION domain restriction to use a constant swaying is included in the animation to simulate wind with little regard to the environment, as has been implemented in several current games, such as World of Warcraft or RIFT. Moreover, the ENVIRONMENT domain is also often restricted in gaming. Some games, including Skyrim and World of Warcraft, have implemented day/night, cloud, and precipitation variation to enhance the gaming experience, although with little consideration to the effects of these variations on plants. Skyrim also provides seasonal plant changes; for example, flowers are present on trees in spring, but not in other seasons. Lastly, since the available computer capacity is unknown, the number of objects selected in the FORM, FUNCTION, and ENVIRONMENT domains need to be limited to a small number.

For the second scenario, suppose we want to animate the process of growth in a detailed biological simulation. To do so, the ENVIRONMENT, FUNCTION, and FORM domains can be implemented in greater detail than in gaming. However, biological simulations often restrict the number of FORM and FUNCTION objects to enable study of a particular effect. For example, to create an animation showing the effect of nutrients on plant growth, one could restrict the FUNCTION domain objects to Growth and Response to Stimulus (Derzaph & Hamilton, 2019). Similarly, to create an animation showing the effect of wind on plant growth, one could restrict the FUNCTION domain objects to Growth and Locomotion (Derzaph & Hamilton, 2013b). If sufficient computer capacity exists.
to run all three objects of the FUNCTION domain (*Growth*, *Locomotion*, and *Response to Stimulus*), using real-time processing, and then restrictions need not occur. As a last example, to study apical meristem growth, Prusinkiewicz et al. restrict each of the FORM and FUNCTION domains to a few objects (Allen, Prusinkiewicz, & DeJong, 2005). They consider the growth object in the FUNCTION domain and its influence on the apical meristem object in the FORM domain.

Finally, for the third scenario, to perform animation for a film, many objects may be selected in the FORM and FUNCTION domain to ensure sufficient detail. As well, the objects selected in the ENVIRONMENT domain can be restricted to the time-period of the film. For example, *Leaf Sheaths* may be selected in addition to *Leaf* objects and properties. These sheaths can be shown in a close-up as the film changes focus from the plant to the main characters. Consider the scene in the *Avatar* film where the main character touches one plant and all plants rapidly close. To design this scene with PLANI, the objects could be restricted to *Leaf* and *Stem* in the FORM domain and *Locomotion* in the FUNCTION domain. As film is the selected animation type, the processing could be limited to batch to enable the frame by frame design and creation of the detail required. The GPU and CPU requirements may be high if the images of plants are required to be realistic and aesthetically pleasing.

The above three scenarios illustrate how PLANI can be applied to a wide variety of dynamic plant model requirements. In addition, PLANI aids in organizing and validating the link between the design of a dynamic plant model
and any available plant ontologies regardless of implementation style and animation type. This thesis provides three descriptions of demonstrations using PLANI in the next chapter (Chapter 4).
Chapter 4  Demonstrations

This chapter contains three demonstrations, with one in each section: Section 4.1 describes the Plant Creator System, Section 4.2 explains animating wind effects on plant motion and growth and Section 4.3 explains animating nutrient effects on plant growth and colour. Each section is divided into subsections according to the stages of PLANI, i.e. stages 1-4. In addition, each section also includes a results subsection showing the animations and a discussion subsection.

4.1 The Plant Creator System

The Plant Creator system is a general-purpose software system for creating dynamic plant models and plant animations. Plant Creator supports a user who is following the PLANI framework to create dynamic plant models. As it runs, the system enables the user to select a subset of objects from the four domains of the PLANI framework, then generates and displays one plant animation. It provides flexibility by allowing the user to change some FUNCTION or ENVIRONMENT objects through data entry or button clicking. The system can generate a wide variety of plant animations, some of which are illustrated in Figure 26 where (a) represents a tree with oval leaves, (b) represents a tree with compound leaves, and (c) represents a gingko biloba tree with fan leaves. In the following subsections, each of stages 1 through 4 is described.

In addition, the system implements the algorithms from the demonstrations given in Sections 4.2 and 4.3. Moreover, the system provides a user interface to interact with the various parameters required to generate any
dynamic plant model, as shown in Figure 27. The system allows for user interaction to change the dynamic plant model shape, select a corresponding species, and save the parameters for future use.

4.1.1 Stage 1: Domain Selection

In Stage 1 the Plant Creator system incorporated all domains FORM, FUNCTION, ENVIRONMENT and VIRTUAL WORLD.

4.1.2 Stage 2: Object Selection

Stage 2 consists of the here PLANI is supporting the designer who is selecting objects and properties from the chosen domains. Appendix D describes the choices made to implement the animating of wind and growth in plants.

4.1.3 Stage 3: Designing the Animation

Once the selection of the objects is complete, the implementation style for the technical instantiation of the dynamic plant model is considered. We examined two current implementation styles: coding or modelling. Coding can be done using either procedural generation (Sievänen, Nikinmaa, Godin, Lintunen, & Nygren, 2013; DeJong & Da Silva, 2011; Prusinkiewicz, Hanan, & Lane, 2008; Godin, et al., 2005; Roux & Sinoquet, 2000; Korpilalti, 1997) or image capture (Wu, Chen, Yan, & Feng, 1999), and modelling. Plant Creator incorporates the procedural approach because it provides the greatest flexibility for dynamic plant model variety with simple coding. In particular, objects in the FORM domain are generated using a nested L-System for each stem, petiole,
and vein (Derzaph & Hamilton, 2013b), the Leaf object is based on the radius variation formula (Derzaph & Hamilton, 2013a) described in Stage 3, and the Fruit object is generated through the use of an ellipsoid formula. Examples of plants produced by the procedural approach are provided in Figure 26.

The FUNCTION domain algorithms for motion considered the currently available methods for producing the Locomotion object. For example, there existed several techniques for animating wind on plants that include branches (Sakaguchi & Ohya, 1999; Hu, Tao, & Guo, 2008; Long, Reimschussel, Britton, & Jones, 2009; Yildiz, 2010) and leaves (Wu, Chen, Yan, & Feng, 1999; Long, Reimschussel, Britton, & Jones, 2009). In our implementation for the Stem object we used an existing branch motion algorithm from Sakaguchi and Ohya and extended the algorithm to produce motion in the Petiole object (Derzaph & Hamilton, 2013b) and Leaf object (Derzaph & Hamilton, 2013a). In addition to the Locomotion object, our demonstration required the Growth object. Our growth algorithm generated a linear growth with the objective of considering a real-time animation rather than a frame-by-frame animation (Derzaph & Hamilton, 2013b), which was generally based on the growth algorithm described by Rodkaew et al. (2004a).

Next, consideration of algorithms for the environment; Wind, Seasonal and Nutrient Regimen objects from the ENVIRONMENT domain are needed. Our demonstration uses a simplified wind algorithm to enable real-time animation without demand on the computer resources. In the VIRTUAL WORLD domain, the Time object is calculated for each frame. The Time object is divided
between fast simulated time and slow simulated time. Fast simulated time is based on a second where as the slow simulated time is based on a day. The algorithm for the Seasonal object uses the passage of daily time (a collection of 86,400 seconds) where the season is changed when a certain time range is reached (91.3125 days which is approximately 7.9 million seconds); our demonstration is based on a simple seasonal assumption of that one quarter of a 365.25-day per year represents the spring, summer, fall, and winter season properties. Note that Growth occurs only in spring and summer whereas fall and winter do not have any growth. In addition, an algorithm is required for the Nutrient Regimen object along with values for the nutrient ranges, which were collected from various biology research papers. The nutrient limiting algorithm determines the Nutrient Regimen object’s specific limiting nutrient, and the limiting nutrient’s effect on the subsequent Growth objects effects on the size, length, width, diameter, and colour properties of the Stem, Leaf, Petiole and Fruit objects (Derzaph & Hamilton, 2019).

To generate a virtual world in which a dynamic plant model can reside requires the Light, Viewpoint, Material, Polygon, Transformation, World Object, WorldInf and Topography objects from the VIRTUAL WORLD domain. Our demonstration uses a consistent light without variation over the entire animation for the implementation of the Light object. The Viewpoint object’s properties enable the viewer to move within and around the virtual world by manipulating the x, y, and z position coordinates. Polygon objects are selected depending on the objects selected from the FORM domain and the algorithms described
earlier. Each FORM object is located in the virtual world through its position property. In Figure 26, the three dynamic plant models all have a default viewpoint, a default light source, a default position, a default material, and white background from the VIRTUAL WORLD domain.

Lastly, consideration of the interaction between the algorithms is required. This is accomplished using instantiated object properties. For example, in the ENVIRONMENT domain, the Nutrient Regimen object contains several nutrient properties, one of which is the nitrogen concentration property that is part of the Primary Macronutrient Regimen object. These properties are passed to the nutrient property in the Response to Nutrient object of the FUNCTION domain, where a process determines the limiting nutrient property. This limiting nutrient property is passed to the Growth object, where it may cause a change to the growth rate property that leads to slower growth. In turn, the Leaf object contains the leaf elongation rate property, which is changed based on the growth rate property of the Growth object in the FUNCTION domain. Finally, the Leaf width, Leaf length, and Leaf thickness properties in the FORM domain are changed based on the leaf elongation rate property. For an overview, the FORM, FUNCTION and ENVIRONMENT relationships are shown in Table 3 based on the sample concepts provided in this paragraph.

The coding described in this section was dependent upon the selection of the objects from each of the domains, consideration of the approach for coding (using a modeling software, image capture, or procedural techniques), understanding the relationships between the domain objects, and the selection
of the animation type from among gaming, biological simulation, and film. In our PLANI demonstration, the general algorithm used to produce the dynamic plant model is illustrated as follows:

Design algorithms for the structure of the plant (based on selections from the FORM domain):

- Design stem using the existing L-System A
- Design petiole using the existing L-System B
- Design vein using the existing L-System C
- Design leaf by selecting the existing leaf shape using the appropriate leaf formula, and appropriate colour.
- Design fruit by selecting the existing fruit shape, using the appropriate fruit formula, and appropriate colour.

Design algorithms for the functioning of the plant (based on selections from the FUNCTION domain)

- Design seasonal appearance of FORM objects through running the program (e.g. leaf appearance in spring)
- Design the motion effects caused by the wind through selecting the on/off button
- Design the Growth effects caused by the wind and nutrient limitation by entering the nutrient values.

Design algorithms for the environmental effects (based on selections from the ENVIRONMENT domain)

- Design the seasonal stage divisions for spring, summer, fall, and winter through running the program.
- Design the wind through selecting the on/off button.
- Design the limiting nutrients and base nutrient values through changing the code.

The Stem object in the FORM domain was designed as previously described with a cylinder for the stem and a hemisphere for the top. The Leaf object in the FORM domain was constructed using the radius variation method.
(Derzaph & Hamilton, 2013a). The branching, petiole, and vein structures were designed as a nested L-system. The functioning of the dynamic plant model was designed by selecting the algorithms for wind and nutrient effects on growth and for wind effects on plant motion. The ENVIRONMENT domain was implemented using the seasons, as described in Section 4.2.2, wind direction and speed, as described in the same section, and the nutrient ranges for wheat, as described in Section 4.3.2.

When it is run, the Plant Creator system supports a user who is following the PLANI framework. The user is presented with a window that has a variety of selection options, as is illustrated in Figure 27. The Graphical User Interface (GUI) is divided into five sections; “Pick a Plant”, “Make a Plant”, “Environment of the Plant”, “Animate Plant”, and the resulting dynamic plant model (at the bottom right of the screen). The “Pick a Plant” and “Make a Plant” sections relate to the FORM domain. The “Animate Plant” section relates to three domains: FUNCTION, ENVIRONMENT, and VIRTUAL WORLD. The “Environment of the Plant” section relates the ENVIRONMENT domain, and the dynamic plant model section implements the VIRTUAL WORLD domain.

The “Make a Plant” section was designed to support the selection of the FORM domain objects and properties. It is divided into the stem, petiole, vein, leaf, fruit and seed subsections. The Stem object and its properties for length and radius are portrayed in the first sub-section, along with the L-string field that is used to produce the overall form of the plant as well as the stem angle property. The Petiole object and its properties are portrayed in the second
subsection. These properties include petiole length, radius, as well as the L-string to produce the petiole form. The Vein object and its properties are given in the third subsection: length, radius and L-string for the venation shape. The Leaf object and its properties are given in the fourth section: length, radius, and shape. The software for constructing the 3-D model superimposes the 3-D model constructed from the L-string for the veins onto the 3-D model for the leaf shape. It was decided to design a static leaf thickness and colour. Lastly included are representations of the Fruit, Seed and Flower objects with a plan for fruit type, seed type and flower type left for future work. The graphical user interface also included three buttons: Render Make a Plant, Default Setting, and Save a Plant. The Render Make a Plant button upon selection shows the plant in the “See the Plant” section. The Default Setting button upon selection sets all the “Make a Plant” section properties back to their default settings. The Save a New Plant button upon selection saves in a database the “Pick a Plant” section taxonomy along with the “Make a Plant” properties.

The “Pick a Plant” section provides various drop-down fields for the plant taxonomy: Division, subdivision, Class, subclass, Order, suborder, Family, subfamily, genus, and species. The list of drop-down fields is obtained from a preloaded plant species database (U.S. Department of Agriculture, 2015). This allows the user to generate a species using the “Make a Plant” section and save it by clicking the Save button. The “Pick a Plant” section was designed to allow the user to choose the plant according to its scientific name in the standard plant biology taxonomy. It provides a Plant Instance drop-down field that is a
convenient way of picking the genus and species of a plant that have been animated. Once a selection is made from this field, the Render Pick a Plant button needs to be selected to make that particular species appear in the “See the Plant” section.

The “Environment of the Plant” section was designed for the ENVIRONMENT domain objects and properties. This section includes the soil texture, soil type, water, oxygen, and nutrient values. In the top subsection placeholder drop down fields are given for soil texture and soil type, whereas fields are given for water and oxygen. In the second subsection appear the macronutrient amount and micronutrient amount, which are used to implement the nutrient properties. The ranges for the nutrients are in the interface and utilized default values based on wheat. Future work will incorporate a user interface to enter those ranges. After the nutrient information the Wind object appears with wind direction and wind strength properties for entry. This subsection also includes a wind on/off button. The timestep for simulating the wind is based on fast simulated time (seconds) as part of a day and subsequently a year. Lastly the Season object is portrayed with a drop-down menu and season on/off button. The timestep for the season is based on a year that includes 365.24 days. This allows for the plant growth that simulates reality.

The “Animate Plant” section implements access to two domains: the FUNCTION and VIRTUAL WORLD. This section includes buttons that increase/decrease the age of the plant, Forward a Year and Back a year respectively. There is also the Make it Grow button that starts the Growth object
to influence the plant size and appearance. The other buttons: rotate left, rotate right, move back, move forward, rotate up, and rotate down implement the Viewpoint object from the VIRTUAL WORLD domain with the responses according to their descriptions.

The “See the Plant” section was designed to display the current choices affecting the VIRTUAL WORLD domain. It shows the dynamic plant model. Each time any of the buttons is hit, an immediate response is portrayed through changes to the dynamic plant model in this section.

4.1.4 Stage 4: Coding the Animation

The coding of the animation was completed within Microsoft Visual Studio using the C++ language with OpenGL and GLUT graphics libraries. The project contains the standard functions required for GLUT: an initDisplay, keyboard, reshape, display and main functions. The initDisplay function contained the set-up information for the menus, lighting, and background colour.

The OpenGL was incorporated into a Windows project through additional calls from the windows form. The GUI was generated through the use of a Windows form and the subsequent field and button objects as described in Section 4.1.3.

The database to support both the plant taxonomy and the storage of the dynamic plant model parameters was generated within Microsoft Access. The schema of the database is shown in Figure 28. The Category group table was for future grouping of more information in the database and was not used for this
demonstration. The Science Name table forms the plant taxonomy hierarchy shown by the ScienceName_1 table (for conceptual purposes) through the ParentScienceNameID foreign key. Each plant taxonomy name is stored in the Category table and related to the Science Name table through the foreign key CategoryID. Sourced from the data entered by the user in the “Pick a Plant”, “Make a Plant” and “Environment of the Plant” sections plant properties are all stored in the PlantGraphicsInformation table with attributes including the nutrient concentrations, L-strings for stem, petiole, and vein, leaf shape type, angles for branches, and plant taxonomy;

4.1.5 Using Plant Creator to Support PLANI framework

Once the coding is complete, Plant Creator can be used to support a user who is working in the PLANI framework. The user is presented with a window that has a variety of selection options, as is illustrated in Figure 27. The Graphical User Interface (GUI) is divided into five sections as described earlier.

The “Make a Plant” section provides data entry fields to determine the characteristics of the default dynamic plant model shown in the bottom right of Figure 27. For example, the Stem object properties are the stem length, 1.0, the stem radius is 0.05, and the stem L-string FBB[+/RFBP][+/RFBP]\RFBP. The Petiole and Vein objects are specified similarly. The Leaf object contains a drop-down field, with a default of chordate, for the leaf shape property. If the user interacts with the software and changes any of the fields in the “Make a Plant” section, the dynamic plant model is only altered once the user presses the Render Make a Plant button.
As stated earlier, the FUNCTION, ENVIRONMENT, and VIRTUAL WORLD domain objects are represented in the “Animate Plant” section’s buttons and fields. For example, the Growth object of the FUNCTION domain is implemented through the Make it Grow button, while the Seasonal object of the ENVIRONMENT domain is implemented through the Season on/off button. In addition, another Seasonal object is implemented through the Add Year button; each click on this button ages the dynamic plant model by one year. In contrast, each click on the Minus Year button reduces the age by one year. The implementation of the Viewpoint object of the VIRTUAL WORLD domain is through the Rotate Left, Rotate Right, Move Back, Move Forward, Rotate Up and Rotate Down buttons. Each button will alter the position of the virtual plan at the bottom right.

Lastly, the ENVIRONMENT domain objects are represented in the “Environment of Plant” section. The Plant Creator software system provides for entry of the concentrations for each of the nutrients for the plant. The dynamic plant model at the bottom right of in Figure 27 is only influenced if the Make it Grow button is pushed in the Animate Plant section after the values are entered.

To find a particular species, use the Plant Instance drop down in the “Pick a Plant” section on the left-hand side of the GUI. If the species does not exist, a biologist can use the Plant Creator software to create it, visualize it, and add it to the database with the chosen taxonomic information. Generating a new species of dynamic plant model is performed by entering values into various fields in the “Make a Plant” section and visualization is performed by clicking the Render
Make a Plant button. This process can be repeated as required until the species is represented properly. Once the biologist is satisfied, the Save New Plant button (middle of the GUI) in the “Make a Plant” section can be clicked to save the current values into the software’s database as shown in Figure 29.

The Plant Creator system provides the user with a way to generate any new plant forms. The process is as follows:

1. Select the taxonomy target from the drop-down boxes the “Pick a Plant” section.
2. Change the properties in the “Make a Plant” section, and when done click the Render Make a Plant button.
3. Examine the dynamic plant model in the “See The Plant” section to ensure that it aligns with the desired species.
4. Save the results by clicking the Save New Plant button.

If the user wants to access the species just created the process is as follows:

1. Select the species from the Plant Instance drop-down box in the “Pick a Plant” section.
2. Click the Render Pick a Plant and the dynamic plant model will appear in the “See the Plant” section.

The system also allows for the user to watch an animation of the plant. If the user wants to watch the effects of wind on a plant, the following process provides that capability:
1. Enter values to specify the direction of the wind where: X specifies east (-1.0) and west (+1) directions, z specifies north (-1) and south (+1) directions, and y specifies unnatural top down (1) or bottom up directions (-1). The system will accept any values in the three boxes to allow any natural or unnatural winds to be specified.

2. Enter the Strength value: ranging from 0 to 500 km/hr.

3. Click the wind on/off button.

4. Watch the dynamic plant model respond in the “See the Plant” section.

The system also allows for the user to watch seasonal changes over time. This is accomplished by selecting the Season on/off button and watching for a 10-minute period.

The system allows for the user to turn growth on and off. This is accomplished through the following steps:

1. Click the Make It Grow button in the “Animate Plant” section.

2. Watch the dynamic plant model respond in the “See The Plant” section. The length of watching will show the subsequent years the plant will grow with 10 minutes per year.

3. Click the Make It Grow button to turn off the growth

The system allows for changes to the plant while growing through two use cases: nutrient growth effects, and wind growth effects.

The growth effects influenced by the wind process is as follows:
1. Select the desired wind direction and strength in the “Environment of the Plant” section.

2. Click the wind on/off button in the “Environment of the Plant” section.

3. Click the Make It Grow button in the “Animate Plant” section.

4. Watch the dynamic plant model in the “See The Plant” section.

5. Click the wind on/off button in the “Environment of the Plant” section to stop the wind.

6. Click the Make It Grow button in the “Animate Plant” section to stop the growth.

Growth effects from Nutrient limitations process is as follows:

1. Make changes to each of the macronutrient and micronutrient amounts required in the “Environment of the Plant” section

2. Click the Make It Grow button in the “Animate Plant” section.

3. Watch the dynamic plant model in the “See The Plant” section.

4. Click the Make It Grow button in the “Animate Plant” section to stop the growth.

The user can return to the starting plant at any time by the following process can be completed:

1. Click the Default Setting button in the “Make a Plant” section.
2. Click the Render Make a Plant button in the “Make a Plant” section.

3. Watch the dynamic plant model in the “See the Plant” section change to the default plant.

Figure 29 through 32 illustrate several plant forms that can be created and visualized with the Plant Creator software system. Figure 29 shows a specific species, *Gingko biloba* tree. Figure 30 shows a tree with oval shaped leaves, while Figure 31 shows a tree with a compound palmate leaves. Figure 32 shows a vine with twisting growth. These examples demonstrate the potential of the Plant Creator system to generate and display a great diversity of plants.

4.1.6 Discussion

The use of plant ontologies to generate design and runtime algorithms has never been done before with respect to animation. PLANI enables biologists and computer scientists to have common discussion points with respect to the ontologies used in Stage 2 of the PLANI framework.

4.2 Animating Wind Effects on Plant Growth

The second demonstration of the usage of the PLANI framework shows how a dynamic plant model can incorporate the effect of wind on a plant. This demonstration followed the four stages outlined in Chapter 3. Stage 1 requires the selection of the domains, all four domains (FORM, FUNCTION, ENVIRONMENT and VIRTUAL WORLD) are selected. Stage 2 requires a selection of objects from each domain using a list based on the ontologies, as illustrated in Figure 19. This selection is described in the Section 4.2.1. Stage 3
requires selection and designing of the algorithms for implementing each selected object. The algorithms are described in Section 4.2.2. Stage 4 requires coding the algorithms. This description is provided in Section 4.2.3 by indicating the programming languages, computer capacity, and IDE utilized. Lastly, Section 4.2.4 provides the implementation results from the design and coding stages.

4.2.1 Stage 2: Object Selection

Stage 2 consists of the designer selecting objects and properties from the chosen domains. The following sections describe the choice made to implement the Plant Creator system. All the domains, FORM, FUNCTION, ENVIRONMENT and VIRTUAL WORLD are described in Appendix D. There are exclusions from Appendix D for the FUNCTION and ENVIRONMENT domains including Response to Stimulus and Response to Chemical objects and properties, and Chemical Regimen objects and properties respectively.

4.2.2 Stage 3: Designing the Algorithms

This demonstration made use of existing algorithms for the FORM and ENVIRONMENT domains. New algorithms were designed for the FUNCTION domain.

Stage 3: VIRTUAL WORLD Algorithm Design and Selection

The VIRTUAL WORLD algorithm utilized real-time animation including parts for initialization, setup of viewpoints, and a main loop. It also contained an overall control via time based on the computer’s capacity, and display. Because
the use of these concepts in Stages 3 and 4 overlaps with the other domains, linkages to the animation portions will be described throughout.

Stage 3: FORM Algorithm Design and Selection

For this demonstration, algorithms were designed for stems and leaves. This included two programs functions, one for a cylindrical body and one for a hemispherical top.

The function for the stem body produces the \((x, y, z)\) coordinates for each vertex (point) of the cylinder surface. The design used the standard cylinder parametric equations, shown in equation 4.1 where the \(s_i\) letter represents the stem length property; the \(r\) represents half of the stem diameter property and \(\theta\) represents the angle of rotation.

\[
x = s_i \\
y = r \cdot \sin \theta \\
z = r \cdot \cos \theta
\] (4.1)

The function sequentially generated the \((x, y, z)\) coordinates required to specify a triangle strip. The top of the stem was generated using a hemisphere shape. This was designed using the standard sphere parametric equations, shown in equation 4.2 for the \((x, y, z)\) coordinates of the surface where the \(r\) represents half of the stem diameter property.

\[x = r \cdot \cos \theta \cos \varphi\]
\[ y = r \sin \theta \sin \varphi \]
\[ z = r \cos \theta \]

(4.2)

Where \( \theta \) and \( \varphi \) represent angles of rotation along x and y-axis, respectively.

The top of the hemisphere is specified as a triangle fan, with one central point and other points along the circumference. A collection of triangle strips were used for the subsequent layers of the hemisphere.

The stem elongation property was represented by a parameter that increased the current stem length property described in detail in the branch growth section. The stem colour property used the red, green, blue (R, G, B) triplet to specify a standard green colour using the colour functions within OpenGL with the values (0.3, 0.2, 0.1).

The Stem Angle object was represented by a branch angle parameter and was used in an L-string to generate the branching structure. The algorithm for the petiole object and its properties (not listed) was designed by using equation 4.1 and by calling the stem algorithm. The algorithm for the petiole branching used a separate L-system from the stem branching L-system. This nested L-system method allowed for flexibility in changing the petiole design.

The Leaf object was designed using the Radius Variation method (Derzaph & Hamilton, 2013a). The leaf shape object was represented by selection of case criteria for each leaf shape described in the Plant Creator system demonstration in Chapter 4. The various shapes available in the system are shown in Table 4. The leaf colour property used the R, G, B triplet from the
VIRTUAL WORLD domain to specify a standard green colour \( (R = 0, G = 1, B = 0) \).

The Radius Variation method is based on the parameterized formulae for a circle, as given in equation 4.3:

\[
\begin{align*}
x &= r \cos(a) \\
y &= r \sin(a)
\end{align*}
\]

To specify the surface of a partially folded leaf in 3D, this method of leaf generation utilizes the parametric equations given in equation 4.4:

\[
\begin{align*}
x &= r_a a \cos a \cos \phi \\
y &= r_a a \sin a \\
z &= r_a a \sin \phi
\end{align*}
\]

where \( a \) is the angle of rotation with \( 0 \leq a \leq 2\pi \), \( r_a \) is the leaf radius for angle \( a \), and \( \phi \) is the fold of the leaf along the \( z \) axis, as shown in Figure 34, and \( x \), \( y \), and \( z \) are the coordinates for the vector composing the leaf edge.

The 2D shape of the leaf, in the \( x \) and \( y \) dimensions, is based on variation of the leaf radius, \( r_a \). The change to the leaf radius, \( \Delta r_a \), is given in various ways. In the majority of instances, it can be calculated using equation 4.5.

\[
\Delta r_a = \frac{L - r_0}{\Delta t}
\]

where \( L \) is leaf length, \( \Delta r_a \) is change in the leaf radius, and \( \Delta t \) is the rotation interval. The leaf radius \( r \) at any given angle of rotation \( t \) is calculated using equation 4.6.
where $r_{a_i}$ is the leaf radius at angle $a_i$. Given a real constant $r_0$, representing the initial radius, real constants $a_0 = 0$, $a_1$, $a_2$, ..., $a_w = 2\pi$, such that $a_0 < a_1 < ... < a_w$, representing angles of rotation, real constants $\Delta r_0$, $\Delta r_1$, ..., $\Delta r_{w-1}$ representing the magnitude of the change in radius during the intervals $(a_0, a_1, ..)$, $(a_1, a_2, ..)$, $(a_{w-1}, a_w, ..)$ where $a_i \leq a \leq a_{i+1}$, and integer constants $s_0$, $s_1$, ..., $s_{w-1} \in \{-1,0,1\}$ representing the sign of the change during the same intervals, the radius at angle $a$.

In many cases, the $a_i$ constants form an arithmetic series of the form $2\pi i/w$, for $0 \leq i \leq w$. In other cases, the $a_i$ constants form a more complex series of the form $(jv+k)2\pi/w$, for $0 \leq j \leq p$ and $0 \leq k \leq v$ such that $pv = w$, where $p$ represents the number of repetitions of the pattern in the leaf, and $v$ represents the number of parts in a pattern. For example, in a five-leaflet palmate, $p$ is 5 because there are five leaflets, and $v$ is 2 because in each leaflet the pattern has two parts, a steady increment in the radius, followed by a steady decrement in the radius, as illustrated in palmate 4 in Table 4. Finally, for some plants, such as palmate 6 and palmate 7 in Table 4, an initial set of $a_i$ constants that describe a pattern is specified, and then the pattern is repeated. In this case, $a_i = j^*a_v + a_k$, where $j$, $k$, $p$, and $v$ are as above, and $a_0$, ..., $a_w$ are real constants that describe the first occurrence of the repeated pattern. For example, the palmate with uneven leaves shown in Figure 33 can be specified with a set of $a_i$ constants, $p = 5$, $v = 2$, $j \in \{0, 1, 2, 3, 4\}$, $v = 2$, $k \in \{0, 1\}$ and two constants $\Delta r_0 = (L-r_0)/\Delta t$ and
Δr_α = \left( \frac{L - (r_0/3)}{\Delta t} \right) \text{.} \) This enables the repeat of the small and large leaflets required to produce the leaf.

Figure 34 illustrates this general method, assuming that the petiole and main vein of the leaf are roughly aligned with the Y-axis and the domain of \( a \) is divided into four intervals. By specifying conditions for each of these intervals, a variety of leaf shapes can easily be generated. To specify the leaf shape shown in Figure 34, we set \( r_0 = 0.05, \) \( w=4, \) and the values of \( a_i \) to multiples of \( \pi/2, \) with \( a_0 = 0, \) \( a_1 = \pi/2, \) \( a_2 = \pi, \) and \( a_3 = 3 \pi/2. \) Other intervals could be defined for \( a_i \) based on \( \pi/4, \pi/6, \pi/8, \) etc.

By varying values for the \( a_i \) constants and the change in radius \( \Delta r_l \), diverse leaf shapes can be generated as surfaces. The \textit{leaf length} and \textit{leaf width} properties are represented in the Radius Variation method by \( r_α, r_z \) respectively.

To give a leaf some thickness a prism is generated using two parallel surfaces using the leaf thickness property. Leaves will only appear within the vein \( L \)-system when the letter A appears as described earlier.

Stage 3: FUNCTION Algorithm Design and Selection

Recall the objects within the FUNCTION domain that were selected: \textit{Growth} and \textit{Locomotion}. These two objects will now be generated using the Growth-Flow algorithm that was designed to implement the growing behaviour required in the FUNCTION domain.
Recall that the *Growth* object was selected for the FUNCTION domain. Growth is an irreversible increase in size, mass, or number of cells (Wareing & Phillips, 1981). Plant growth is seasonal, meaning that the algorithm for the *Growth* object in the FUNCTION domain should be linked to the *Seasonal* object in the ENVIRONMENT domain. When plant growth occurs, a sigmoidal growth curve is observed such that the growth rate slowly increases during the early part of the growing season, remains constant (linear growth) during the main part of the growing season, and then declines so that it reaches zero when the segment achieves its maximum size (Wareing & Phillips, 1981). Ignoring other environmental factors, growth for a living thing, such as a plant, shows no, logarithmic, linear, or exponential increase with respect to time, depending on the current time during the year. Three existing ways of measuring plant growth are: (1) absolute growth rate (AGR), which is the total gain in size (height, weight, radius) or mass (dry or fresh weight) over time, and (2) relative growth rate (RGR), which is the change in the logarithm of mass over time. AGR and RGR are used for measuring growth in plant physiology. For illustration AGR, and RGR, as described in in equation 4.7, and equation 4.8, and, respectively.

\[
AGR = \frac{\Delta n}{\Delta t} = \frac{n_2 - n_1}{t_2 - t_1} \tag{4.7}
\]

\[
RGR = \frac{\Delta n}{\Delta t} \cdot \frac{1}{n} = \frac{\ln n_2 - \ln n_1}{t_2 - t_1} \tag{4.8}
\]

where \(n_1\) and \(n_2\) are the relevant size or mass measurements and \(t_1\) and \(t_2\), where \(t_2 > t_1\), are the respective times at which these measurements were taken.
A common growth formula is based on the Richards growth model (1951, Jacobs, 1954; Telewski & Jaffe, 1986; Stokes, Fitter, & Coutts, 1995) which considered exponential growth, as described in equation 4.09:

\[ y_t = A(1 + be^{-kt})^{1/m}, \quad m > 1, b > 1, k > 0, t \geq 0 \] (4.9)

where \( t \) is time, \( y_t \) is the size of the growing part (e.g. amount of dry matter) at time \( t \), \( A \) is the asymptotic value of the size, and \( m, b, \) and \( k \) are constant coefficients, where \( m \) determines the curve type, \( b \) describes the linear component of growth, and \( k \) determines the rate at which growth changes over time (Jacobs, 1954). According to Richards, “\( k \) is the ‘rate constant’ which determines the spread of the curve along the time axis, and \( b \) is usually unimportant biologically, since it reflects only the choice of the zero of the time” (Jacobs, 1954). (This \( b \) should not to be confused with the vector \( b \) used elsewhere in this section.) The constant \( m \) influences the shape of the curve. The equation used for implementing this example resembles this growth formula as described in the following section.

Branch growth is affected by wind through a thickening of the stem (Goodman & Ennos, 1996; Jacobs, 1954; Mattheck, 1991; Stokes, Fitter, & Coutts, 1995; Telewski & Jaffe, 1986), and reduced elongation (Coutand & Moulia, 2000; Lawton R.O., 1982; Whitehead, 1962). The direction of the branch is altered to align more closely with the wind vector, if the wind is constant and strong over the lifetime of the plant. In general, the effect of wind on growth is not significant unless the wind persists over long periods and has sufficient
speed, e.g., over 4 meters per second for aspen (Flückiger, Oertli, & Flückiger-Keller, 1978). The amount of effect on growth is correlated with wind speed (Flückiger, Oertli, & Flückiger-Keller, 1978). Table 5 also includes the observed effect of wind on growth rate for the sample plants where available.

The new function is *Locomotion* which results from wind hitting the plant. Branches are modeled using a segment-based approach (Sakaguchi & Ohya, 1999). A *segment* is the linear part of a branch between nodes. A *node* is a point on a plant where new growth is possible. A node also corresponds to a joint in an animated model. A branch is represented by a computer science tree data structure where vertices represent segments and arcs represent the nodes. The motion of a segment depends on the motion of its child segments (connected segments closed to the external end of the branch) and the effect of wind on a particular segment (Derzaph & Hamilton, 2013b). The effect of the wind on a segment is calculated based on the combination of wind velocity, axial damping, and restoration force. The *axial damping* is influenced by the tensile strength of the branch, dependent upon its thickness. The *restoration force* (*resistivity*) is the force influencing a branch that is not in its rest position to return to that position. Equation 4.10 can be used to calculate the motion of a segment (Sakaguchi & Ohya, 1999):

\[ N = (F_w + K + R + T_r) \times b \]  \hspace{1cm} (4.10)

where \( N \) is the moment of force (torque), \( F_w \) is the external force (wind), \( K \) is the resistivity, \( R \) is the axial damping force at a node, \( T_r \) is the force that propagates
back from the child branches, $\times$ indicates the cross product, and $\mathbf{b}$ is a vector from the node at the beginning of the segment, where it attaches to the parent segment, to the center of gravity of the segment. This method described by Sakaguchi and Ohya (1999) is illustrated in Figure 35. On the left, three segments joined at two nodes are shown, and on the right, the same segments are shown after wind has moved the top segments to the right. The segment of interest is the middle one. The forces listed in Figure 35 act in the directions shown.

Growth-Flow Method

In the Growth-Flow method, the branches are grown and moved using algorithm 1 then the leaves are grown and moved relative to the branches using algorithm 2. The method described algorithmically in Branch Growth and Motion and Leaf Growth and Motion sections below contains two components resulting from wind influence: physical motion, described in the branch and leaf response to wind and Branch and leaf movement sections below, and growth effects on branches and leaves described in the Branch and Leaf growth sections below. The methods used to simulate the physical motion and growth for a branch are described in branch response to wind and branch growth sections, and those for a leaf are described in leaf response to wind and leaf growth sections.

Branch Response to Wind
To simulate the physical motion and growth for a branch, we consider the movement in the following branch movement and branch growth sections.

Branch Movement

Branch movement in response to wind was modeled by swaying the branch toward the wind vector and back over time. To explain in a simple fashion, consider a 2D example, as illustrated in Figure 36, which uses a right-handed coordinate system: part (a) shows the centers of rotation of a plant (which correspond to the nodes); part (b) shows that as the wind pushes left on a branch segment, the angle of rotation increases by $\theta$ around the center of rotation for that branch segment, and part (c) shows that the angle of rotation is decreased as the branch returns to its original position due to the restorative force and tensile strength of the branch. To produce the rotation, the value for $\theta$ is increased over half a time period until a certain angle is reached and then it is decreased back to its original value over the other half of the time period. To achieve a natural slowing down and speeding up at the ends of the arcs, a sinusoidal ease-in/ease-out function is used (Parent, 2012).

Although the example was shown in 2D, the wind vector is actually 3D. Given the branch direction, $b/b$, and wind direction $w/w$, defined in the same coordinate system, the axis of rotation for the wind effect is obtained by taking their cross-product $b/b \times w/w$. The amount to rotate around this axis is determined based on the angular acceleration, calculated by dividing the torque from (equation 4.10) by the mass of the branch. The maximum rotation, which
would cause the branch to align with the wind vector, is \( \cos^{-1}(\mathbf{b} \cdot \mathbf{w}/|\mathbf{w}|) \). The mass of a branch is arbitrarily set to 10 times the mass of a leaf. A leaf is thus affected 10 times more than a branch by the wind. An arbitrary mass for a branch is used to simplify calculations.

**Branch Growth**

To represent the effect of wind on growth, we assume that there is a species-specific minimum speed \( w_{\text{min}} \), and maximum speed \( w_{\text{max}} \), such that growth is not affected if the wind is less than or equal to \( w_{\text{min}} \) and that the full effect occurs if the wind is greater than or equal to \( w_{\text{max}} \). We assume that the effect is linearly interpolated between these speeds. To quantify the effect of wind on growth, we define the wind growth factor \( f(w) \), which yields a value in the range 0, 1, as follows:

\[
f(w) = \min \left( 1, \max \left( 0, \frac{w - w_{\text{min}}}{w_{\text{max}} - w_{\text{min}}} \right) \right)
\]  

(4.11)

The three effects on branch growth are simulated: radius increase, elongation reduction, and *cumulative change in growth angle* (henceforth called the *growth angle*). For a wind with uniform velocity \( w \) (and constant speed \( w \)), we use equation 4.12, equation 4.13, and equation 4.14 for these three factors:

\[
r_t = r_0 (1 + g_r \Delta r)^t \quad \text{where} \quad \Delta r = 1 + s_r f(w)
\]  

(4.12)

\[
l_t = l_0 (g_l \Delta l)^t, \quad \text{where} \quad \Delta l = 1 - s_l f(w)
\]  

(4.13)

\[
\alpha_t = \alpha_0 (1 + \Delta \alpha)^t, \quad \text{where} \quad \Delta \alpha = c_\alpha s_\alpha f(w)
\]  

(4.14)
where \( t \) is the elapsed time, \( r_t, l_t, \) and \( \alpha_t \) are the radius, length and growth angle at time \( t; \) \( r_0, l_0, \) and \( \alpha_0 \) are the original radius, length, and growth angle at time 0; \( g_r \) and \( g_l \) are the normal rates of radius growth and branch elongation; \( \Delta r, \Delta l, \) and \( \Delta \alpha \) are the proportional changes in radius growth, branch elongation and growth angle due to the wind, respectively, and \( c_\alpha \) is a conversion constant from meters per second to degrees per second, i.e. 1 degree/m. The species-specific constants \( s_r, s_l, \) and \( s_\alpha, \) are assumed to be in the range \([0, 1]\). Equations 4.12, 4.13 and 4.14 specify that the changes are affected by both the speed and the duration of the wind but the effect of the wind direction is negligible. If the length reaches a maximum determined by the species, no further increase in length occurs. If the wind speed is less than a species-specific minimum speed \( w_{min}, \) then growth is not affected by the wind (Wadsworth, 1960).

**Branch Growth and Motion Algorithm**

**Input:** time \( t \), branch vector \( \mathbf{b} \), wind velocity \( \mathbf{w} \), mass \( m \), radius \( r_0 \), length \( l_0 \), and angle \( \alpha_0 \)

**Output:** branch radius \( r_t \), branch length \( l_t \), and branch growth angle \( \alpha_t \), all at time \( t \)

1. Set the wind growth factor \( f(w) = \min\left(1, \max\left(0, \frac{w-w_{min}}{w_{max}-w_{min}}\right)\right) \) (Equation 4.11)
2. Set the branch radius: \( r_t = r_0(1 + g_r \Delta r)^t \), where \( \Delta r = 1 + s_r f(w) \) (Equation 4.12)

3. Set the branch length: \( l_t = l_0(g_l \Delta l)^t \), where \( \Delta l = 1 - s_l f(w) \) (Equation 4.13)

4. If \( l_t \geq l_{\text{max}} \) then \( l_t = l_{\text{max}} \)

5. Set the growth angle: \( \alpha_t = \alpha_0(1 + \Delta \alpha)^t \), where \( \Delta \alpha = c_{\alpha} s_{\alpha} f(w) \) (Equation 4.14)

6. Determine the axis of rotation for the branch as \( \mathbf{u} = \mathbf{b} \times \mathbf{w} \)

7. Rotate the branch around axis \( \mathbf{u} \) according to the growth angle \( \alpha_t \)

8. Additionally, \( \mathbf{F}_w = m \mathbf{w} \), rotate the branch around axis \( \mathbf{u} \) according to wind force \( \mathbf{F}_w \) (Equation 4.10)

Algorithm 1: The general algorithm for branch growth and movement.

*Leaf Response to Wind*

The leaf response to the wind is modeled by first considering movement and then considering growth.

**Leaf Movement**

The leaf blade moves in response to wind in one of three ways: bending, twisting, and vibrating. **Bending** is the motion when the leaf blade is facing the wind (perpendicular to the wind). **Twisting** is the motion where the leaf blade turns in response to the wind, which occurs when the blade is not perpendicular
to the wind. **Vibration** is tiny movements along the petiole/leaf axis when the leaf blade is parallel to the wind. The petiole response to wind is limited to bending, because vibration and twisting are negligible due to the small surface area. Another consideration is the effect of neighbouring leaves on the motion of the leaf where there is collision and bounce. For the Growth-Flow approach, we ignore these effects based on our assumption that they are relatively small.

As mentioned in the previous section, to model the effects of gusts of wind, the Growth-Flow approach uses a constant wind velocity but includes a stochastic parameter that influences leaf motion. Let $l_s \in [0, 1]$ represent the fraction of the wind’s velocity that affects a leaf at the current time. This factor is incorporated in the equations for leaf motion. The stochastic approach for leaf velocity was selected to enable real-time animation to have fewer calculations and thus save processing power.

The remainder of this section describes the Growth-Flow approach to leaf motion by considering each of bending, twisting, and vibrating. Throughout, we use $\Delta t$ to represent the frame time interval, $e_p$ to represent the tensile strength of the petiole, and $e_b$ to represent the tensile strength of the blade.

**Bending**

In the Growth-Flow approach, if a leaf blade is facing (perpendicular to) the wind, then bending occurs either at the blade center of rotation which is the bottom of the blade, as illustrated in Figure 37, or at the petiole center of rotation which is the bottom of the petiole, as illustrated in Figure 38. If the tensile
strength of the petiole is less than that of the blade, then any bending will be *petiole bending*; otherwise it will be *blade bending*. We denote the amount of petiole bending by $\theta$ and the amount of blade bending by $\beta$. For example, in the case of blade bending, bending occurs if the tensile strength of the blade is less than the force of the wind, as illustrated in Figure 37. We model the bending of a petiole as a rotation at a single point as follows:

$$\Delta \theta_t = s_t w l_s c_\theta$$  \hspace{1cm} (4.15)

$$s_t = \begin{cases} -s_{t-1}, & \text{if } \theta_{t-1} + \Delta \theta_{t-1} \Delta t > \theta_{\text{max}} \\ -s_{t-1}, & \text{if } \theta_{t-1} + \Delta \theta_{t-1} \Delta t < \theta_{\text{min}} \\ s_{t-1}, & \text{otherwise} \end{cases}$$ \hspace{1cm} (4.16)

$$\theta_t = \theta_{t-1} + \Delta \theta_t \Delta t$$ \hspace{1cm} (4.17)

The direction of bending, $s_t \in \{1, -1\}$, stays constant until the sum of the previous angle of rotation, $\theta_{t-1}$ plus an update of $w l_s c_\theta \Delta t$ in the same direction would be higher than the maximum angle $\theta_{\text{max}}$ or lower than the minimum angle $\theta_{\text{min}}$, in which case $s_t$ switches, as described in equation 4.16. The change to the angle of bending, $\Delta \theta_t$, is computed based on the direction $s_t$, the stochastic wind speed $w l_s$, and a conversion constant $c_\theta$. The complete motion, shown for a blade in Figure 37, is accomplished by incrementing the current angle with the change in angle at the center of rotation over a change in time $\Delta t$, as described by equation 4.17. Blade bending is done in the same fashion but with $\beta$ replacing $\theta$ in equations 4.15 to 4.17. If the leaf is oriented on the XY plane, for example, bending occurs in the Z dimension.
Twisting

Another motion leaves exhibit is twisting around the petiole. We modeled twisting as a rotation of the petiole at the point where it joins the branch. We assume that in the local frame of reference the leaf is initially lying on the XY plane and the branch is aligned roughly with the Y-axis. The change in the twist angle, $\Delta \psi_t$ is determined from the sign $s_t$, influenced by the wind speed, $w$, subject to a conversion constant $c_\psi$, in the range from $\psi_{min}$ to $\psi_{max}$, as described in equation 4.18 and 4.19. The complete motion, which is shown in Figure 39, is accomplished by incrementing the current angle with the change in angle over a change in frame time, $\Delta t$, as described by equation 4.20.

$$\Delta \psi_t = s_t w l_s c_\psi$$  \hspace{1cm} (4.18)

$$s_t = \begin{cases} 
-s_{t-1} & \text{if } \psi_{t-1} + \Delta \psi_{t-1} \Delta t > \psi_{max} \\
-s_{t-1} & \text{if } \psi_{t-1} + \Delta \psi_{t-1} \Delta t < \psi_{min} \\
s_{t-1} & \text{otherwise} 
\end{cases}$$  \hspace{1cm} (4.19)

$$\psi_t = \psi_{t-1} + \Delta \psi_{t-1} \Delta t$$  \hspace{1cm} (4.20)

Vibrating

The third motion a leaf exhibits is vibration, as illustrated in Figure 40. If the leaf is oriented flat on the XY plane, vibration is in the Z dimension. The change in the angle of vibration, $\phi_t$, is determined from, $s_t$, influenced by the wind speed, $w l_s$, subject to a conversion constant $c_\phi$, in the range from $\phi_{min}$ to $\phi_{max}$, as described in equation 4.21 and 4.22. The complete motion, which is shown in Figure 40, is accomplished by incrementing the current angle with the
change in angle over a change in frame time, $\Delta t$, as described by equation 4.23.

$$\Delta \phi_t = s_t \, \text{wl}_s \, c_\phi$$

(4.21)

$$s_t = \begin{cases} -s_{t-1} \text{ if } \phi_{t-1} + \Delta \phi_{t-1} \Delta t > \phi_{\text{max}} \\ -s_{t-1} \text{ if } \phi_{t-1} + \Delta \phi_{t-1} \Delta t < \phi_{\text{min}} \\ s_{t-1} \text{ otherwise} \end{cases}$$

(4.22)

$$\phi_t = \phi_{t-1} + \Delta \phi_{t-1} \Delta t$$

(4.23)

The equations for bending, twisting, and vibrating are the same except for the various parameters. They can be implemented with a single computer science function with different values for its parameters.

Leaf Growth

Changes in leaf thickness and leaf radius are modeled in equation 4.24 and equation 4.25, respectively, as follows:

$$h_t = h_0 (1 + g_h \bar{w} s_h)^{\Delta T}$$

(4.24)

$$d_t = d_0 (g_d \bar{w} s_d)^{\Delta T}$$

(4.25)

where $\Delta T$ is the elapsed time in days, $h_t$ is the resulting leaf thickness at time $t$, $h_0$ is the original leaf thickness, $g_h$ is the normal leaf thickness growth rate, $s_h$ is a species specific constant, $d_t$ is the resulting leaf radius at time $t$, $d_0$ is the original leaf radius, $g_d$ is the normal leaf growth rate, the $\bar{w}$ is average wind speed,and $s_d$ is a species specific constant.
Leaf Growth and Motion

The algorithm for leaf growth and motion is given in Algorithm 2. No information could be found about the frequency of the motions for leaf bending, twisting, and vibrating. Therefore, we arbitrarily chose to select from a uniform random distribution based on 1/3 for each. There is a potential for a leaf to bend, twist and vibrate in 3 second intervals. By using the random approach, we were able to demonstrate all three types of behaviour without doing extensive calculations.

Global Initialize:  
\[ t = 0, \ s_0 = 1, \ \theta_0 = 0, \ h_0 = 1, \ d_0 = 1 \]

Choose a random type of leaf motion \( r \in \{1, 2, 3\} \)

Input: Time \( t \), old leaf parameters \( (\theta_{t-1}, \beta_{t-1}, \psi_{t-1}, \varphi_{t-1}) \), wind speed \( w \)

Output: New leaf parameters \( (\theta_t, \beta_t, \psi_t, \varphi_t) \), thickness \( (h_t) \), and radius \( (d_t) \)

If \( w > c_p e_p \) or \( w > c_b e_b \), i.e., if the wind exceeds the maximum tensile strengths of the petiole \( (e_p) \) or blade \( (e_b) \), then exit because the leaf is no longer attached.

\[ \theta_t = \theta_{t-1}, \ \beta_t = \beta_{t-1}, \ \psi_t = \psi_{t-1}, \ \text{and} \ \varphi_t = \varphi_{t-1} \]

if \( r = 1 \)

if \( e_p \leq e_b \),

Bend the petiole at the petiole center of rotation
\[ \Delta \theta_t = s_t w_l c_\theta \] (Equation 4.16)

\[ \theta_t = \theta_{t-1} + \Delta \theta_{t-1} \Delta t \] (Equation 4.18) where

\[
s_t = \begin{cases} 
-s_{t-1}, & \text{if } \theta_{t-1} + \Delta \theta_{t-1} \Delta t > \theta_{\text{max}} \\
-s_{t-1}, & \text{if } \theta_{t-1} + \Delta \theta_{t-1} \Delta t < \theta_{\text{min}} \\
s_{t-1}, & \text{otherwise}
\end{cases} \] (Equation 4.17)

else

Bend the leaf blade at the leaf center of rotation

\[ \beta_t = \beta_{t-1} + s_t l_s c_\beta \Delta t \] (Equation 4.16 with \( \beta \) instead of \( \theta \), \( s_t \) varies like Equation 4.17)

else if \( r = 2 \)

\[ \Delta \psi_t = s_t w_l c_\psi \] (Equation 4.18)

Twist the leaf \( \psi_t = \psi_{t-1} + \Delta \psi_{t-1} \Delta t \) (Equation 4.20) where

\[
s_t = \begin{cases} 
-s_{t-1}, & \text{if } \psi_{t-1} + \Delta \psi_{t-1} \Delta t > \psi_{\text{max}} \\
-s_{t-1}, & \text{if } \psi_{t-1} + \Delta \psi_{t-1} \Delta t < \psi_{\text{min}} \\
s_{t-1}, & \text{otherwise}
\end{cases} \] (Equation 4.21)

else if \( r = 3 \)

\[ \Delta \varphi_t = s_t w_l c_\varphi \] (Equation 4.21)

Vibrate the leaf at the leaf center of rotation \( \varphi_t = \varphi_{t-1} + \Delta \varphi_{t-1} \Delta t \) (Equation 4.23) where
\[ s_t = \begin{cases} -s_{t-1} & \text{if } \varphi_{t-1} + \Delta\varphi_{t-1}\Delta t > \varphi_{\text{max}} \\ -s_{t-1} & \text{if } \varphi_{t-1} + \Delta\varphi_{t-1}\Delta t < \varphi_{\text{min}} \\ s_{t-1} & \text{otherwise} \end{cases} \]  

(Equation 4.22)

Calculate the leaf thickness \( h_t = h_0(1 + g_h \bar{w} s_h)^e \) (Equation 4.24)

Calculate the leaf radius \( d_t = d_0(g_d \bar{w} s_d)^e \) (Equation 4.25)

Algorithm 2: The general algorithm for growth and movement of one leaf.

Stage 3: ENVIRONMENT Algorithm Design and Selection

One environment selection from the EO is the Wind object. For simplicity, we treat the wind as having a constant velocity everywhere in the biological simulation. It is set according to a wind model. Gusts could be modeled in the wind model by rapidly increasing and decreasing the wind speed in particular areas from time to time. Instead, for simplicity, we give leaf movements a random component that can be interpreted by the viewer as wind gusts. In addition, for the branch motion a simplified wind model is used. The wind velocity is represented using a vector with speed and direction as shown in equation 4.26.

\[ \mathbf{w} = s \mathbf{d} \]  

(Equation 4.26)

where \( \mathbf{w} \) is a 3D vector representing the wind velocity, \( s \) is a scalar representing the wind speed and \( \mathbf{d} \) is a 3D unit vector in the direction of the wind.

Next environment selection from the EO, Seasonal object, is implemented through a linkage to the Time object in the VIRTUAL WORLD. We chose to let
10 minutes of elapsed slow simulated time in the *Time* object correspond to one seasonal year for the *Seasonal* object as illustrated in equation 4.27.

\[
Year = 10 \text{ minutes}
\]  

(4.27)

The seasons spring, summer, fall, and winter seasons are then calculated as \(1/4\) of a year, as shown in equation 4.28

\[
Season = \frac{Year}{4}
\]  

(4.28)

4.2.3 Stage 4: Coding the Animation

The coding of the animation was completed within Visual Studio with OpenGL and GLUT. The project contains the standard functions required for GLUT: `initDisplay`, `keyboard`, `reshape`, `display`, and `main` functions. The `initDisplay` function contains the set-up information for the menus, lighting, and background colour. The `keyboard` function contains code to change plant years (‘>’ key to increase years”, ‘<’ key to decrease years), to turn growth on/off (‘g’ key/’j’ key), turn wind on/off (‘p’ key/”?’ key), to increase/decrease wind speed by 1 km (‘v’ key/’c’ key), and to change wind directions. The center of rotation for the wind was based on the appearance of an R symbol in the constructed L-string. Because the axiom does not contain an R symbol, the first segment does not sway with the wind. For the last task, the following keys are used:

- ‘w’ for west wind (along x axis)
- ‘e’ for east wind (along negative x axis)
- ‘s’ for south wind (along z axis)
• ‘n’ for north wind (along negative z axis)
• ‘a’ for south west wind (along x axis and z axis)
• ‘b’ for north west wind (along x axis and negative z axis)
• ‘d’ for south east wind (along negative x axis and z axis)
• ‘m’ for north east wind (along negative x axis and negative z axis)
• ‘t’ for top down (along negative y axis)
• ‘r’ from top westerly direction (along negative y axis and x axis)
• ‘y’ from top easterly direction (along negative y axis and negative x axis)
• ‘h’ from top southerly direction (along negative y axis and z axis)
• ‘q’ from top northerly direction (along negative y axis and negative z axis)
• ‘i’ from bottom (along y axis)
• ‘u’ from bottom in a westerly direction (along y axis and x axis)
• ‘o’ from bottom in an easterly direction (along y axis and negative x axis)
• ‘l’ from bottom in a southerly direction (along y axis and z axis)
• ‘k’ from bottom in a northerly direction (along y axis and negative z axis)

The reshape function provides the perspective and display parameters to allow for the dynamic plant model to be rendered. The display function contains the camera parameters, along with the wind, year, and plant size parameters. It also contains calls to the three draw functions. The main function contains the
setup code for the window, calls to initialize the keyboard, reshape, and display functions along with calls to the two L-system functions.

The two L-system functions, namely calculateProductionRules and growPlant, perform the generation of the L-string through repeated substitution. The display function calls three other functions to perform graphical substitutions to produce the 3D model of the plant. The drawPlant, drawPetiole, and drawVein functions replace the L-string items with corresponding 3D geometric shapes. Each substitution is completed as follows:

- Whenever an ‘A’ letter is encountered in the L-string, the chosen leaf type is referenced, and the corresponding leaf shape is substituted.
- Whenever a ‘F’ letter is encountered in the L-string, a stem shape is substituted.
- Whenever a ‘L’ letter is encountered in the L-string, a flower shape was planned to be substituted (not implemented)
- Whenever a ‘P’ letter is encountered in the L-string, the L-string for a petiole is substituted, a call to the drawPetiole function is made, and further substitutions are conducted for in the L-string for the petiole, as described by these bullets.
- Whenever an ‘R’ letter is encountered in the L-string, a 3D rotation is applied, based on the current wind velocity.
- Whenever a ‘S’ letter is encountered in the L-string, a seed is substituted, although not in this particular demonstration.
Whenever a ‘U’ letter is encountered in the L-string, a fruit shape is substituted, although not in this particular demonstration.

Whenever a ‘V’ letter is in the petiole L-string, the L-string for the vein is substituted, a call to the drawVein function is made, and further substitutions are conducted for the L-string for the petiole, as described by these bullets. A ‘V’ letter will only appear in the L-string for a petiole.

Whenever a ‘-‘, ‘+‘, ‘&‘, ‘^‘, ‘/‘, or ‘\‘ symbol is encountered in the L-string, the stem is rotated by a default change in angle around the positive/negative x, positive/negative y, or positive/negative z axis, respectively.

Whenever a ‘(‘ symbol is encountered in the L-string, all succeeding symbols until a ‘)‘ symbol are evaluated to produce a real number. This number is applied as a change in length if the ‘(‘ symbol is preceded by an ‘F‘ symbol, or as a change in the angle of rotation if the ‘(‘ symbol is preceded by a ‘-‘, ‘+‘, ‘&‘, ‘^‘, ‘/‘, or ‘\‘ symbol.

Whenever a ‘[‘ symbol is encountered in the L-string, the current position is stacked so that when a ‘]‘ symbol is next encountered processing can continue at the original position.

Whenever a ‘{‘ symbol was encountered in the L-string, a new year of plant growth is started. The use of the ‘{‘ symbol simplified the task of moving between years in response to keyboard commands.

The parameters that are passed amongst the calculateProductionRules, growPlant, drawPlant, drawPetiole, and drawVein functions include the L-string,
current year, growth rate, wind speed and direction, and various plant size parameters. The software also initializes the stem and leaf graphical shapes once to streamline processing at runtime.

4.2.4 Results of the Animation

The tree branch, petiole, and vein structure were generated using a nested L-system and the leaves were generated using the Radius Variation method (Derzaph & Hamilton, 2013a). Figure 41 demonstrates the effect of a 10 km/h, left-to-right wind on a tiny branch with four leaves attached by petioles over one full cycle of swaying motion. To simplify viewing, in this figure, the branch is shown at one-third scale and the leaves and petioles have their normal size. During the first 16 frames, the branch is moving from right to left. For example, the leftmost petiole visible in the first frame can be seen (where visible) to gradually move to the left in the first 16 frames. Then it moves back to the center over the next 16 frames, it continues to the right over 16 more frames, and finally back to the left over the remaining 16 frames to complete the cycle. The leaves also exhibit a variety of bending, twisting, and vibrating behaviors. For three seconds, one of the three behaviors are randomly selected for updating (1/3 probability each). We set the maximum and minimum angles for these behaviors as $\theta_{\text{min}} = -5$, $\theta_{\text{max}} = 5$, $\beta_{\text{min}} = -10$, $\beta_{\text{max}} = 10$, $\psi_{\text{min}} = -15$, $\psi_{\text{max}} = 15$, $\phi_{\text{min}} = -5$, and $\phi_{\text{max}} = 5$. Experiments with other ranges for the three behaviors also resulted in apparently natural plant behavior, although the ranges chosen seemed the most realistic visually. Overall, as Figure 41 illustrates for an
arbitrary plant, the animated branches sway and the animated leaves twist, bend, and vibrate.

Further illustration of the response to wind is provided in Figure 42 for a seven-year-old tree. As in Figure 41, the tree trunk and branches are shown at a smaller scale.

The Growth-Flow method, in addition to supporting full motion response of branches and leaves to the wind, also limits growth as illustrated in Figure 43. Parts (a) and (b) of this figure demonstrate growth over a two-day period of simulated time without wind and with wind, respectively. Note that the growth speed was increased greatly allowing noticeable growth in a two-day period. The plant’s growth in terms of branch length and leaf size are both visibly reduced when wind is present, which matches observations about natural trees.

As shown by the above results, the Growth-Flow method is effective at calculating leaf behavior in response to wind, both with respect to motion (Figure 41 and Figure 42) and growth (Figure 43). The wind effect on plant motion results in the bending, swaying, and twisting of the plant. The wind effect on plant growth results in thickening of leaf blade, reduction of leaf radius, reduction in stem elongation, and increase in stem radius and influences the direction of growth to align more closely with the wind vector when its direction is consistent for a length of time.
4.2.5 Discussion

Animating responding to wind has been studied since 1999 (Sakaguchi & Ohya, 1999). Previous research did not study the animation of wind effects on growth. The essential effect of wind on growth is to reduce the height of a plant, change the angle of stem growth, and thicken the stems and leaves. Animating these effects is a novel contribution of our research.

As mentioned in the Leaf Motion subsection of Section 2.1.1.6, several studies have examined leaf motion (Wejchert & Haumann, 1991, Jirasek, Prusinkiewicz, & Moula, 2000, Feng, Chen, Yan, & Wu, 2001, Lu, Guo, Zhao, & Li, 2008, Wu, WenHui, & Guanghui, 2006, Feng, Chen, Yan, & Wu, 2001; Beaudoin & Keyser, 2004; Yang, Sheng, Wu, & Sun, 2009; Habel, Kusternig, & Wimmer, 2009; Zioma, 2007; Li, et al., 2010, Qin & Chen, 2006; Wu, WenHui, & Guanghui, 2006). Wong and Datta (2004). The main one utilized was Feng et al. 2001 for our leaf motion and was adapted with three motions bending, twisting, and vibrating. Although the leaf response to wind was randomized, the leaf motion simulated natural motion this author observed in trees.

With respect to the Stem Motion subsection in Section 2.1.1.6, several studies used physically based methods (Sakaguchi & Ohya, 1999; Akagi & Kitajima, 2006; Habel, Kusternig, & Wimmer, 2009; Ding, Chongcheng, Liyu, & Qinmin, 2009; Feng, Chen, Yan, & Wu, 2001) and procedurally based methods (Ono, 1997; Shinya & Fournier, 1992; Ota, Tamura, Fujimoto, & Maraoka, 2004; Zioma, 2007). Our approach to stem motion in response to wind was simplified by having the stem sway back and forth in response to the wind’s speed. As
physically based wind response was not the primary focus of our research, the swaying of the stems was sufficient to demonstrate that the effect of wind on growth changes with respect to wind.
4.3 Animating Nutrient Effects on Plant Growth and Colour

The sample demonstration for animating nutrient effects on plant growth and colour sample demonstration followed the four stages outlined in Chapter 3. In this demonstration, Stage 1 requires the selection of the domains, in this demonstration all four domains: FORM, FUNCTION, ENVIRONMENT and VIRTUAL WORLD are selected. Stage 2 requires a selection of objects from each domain, which is made using a list based on the ontologies as illustrated in Figure 19. This selection is described in Section 4.3.1 for each of the respective domains, FORM, FUNCTION, ENVIRONMENT and VIRTUAL WORLD domains. Stage 3 requires selecting or designing algorithms for each selected object. The algorithms are described in the Section 4.3.2. Stage 4 requires coding the algorithms which is described Section 4.3.3. The remaining aspects of the implementation that results from design and coding the algorithms are described in Section 4.3.4.

4.3.1 Stage 2: Object Selection

Stage 2 consists of the designer selecting objects and properties from the chosen domains. The VIRTUAL WORLD domain objects as described in Section 4.1.2 Stage 2 VIRTUAL WORLD domain section. The FORM domain objects list mentioned in Section 4.1.1 were selected for the FORM domain. This selection includes one of the hierarchies the selection came from and the target object or property. The FUNCTION objects listed in Section 4.1.1 (includes one of the hierarchies the selection came from and the target object or property) were selected for the FUNCTION domain. The Abiotic Seasonal
object from the plant environment ontology (EO:0007359) described in Section 4.1.1, was selected for the ENVIRONMENT domain.

4.3.2 Stage 3: Designing the Algorithms

A plant animation is created by rendering the current state of the dynamic plant model, applying growth and colour changes to the model, rendering again, and so on. The animation procedure is described by giving the updates that occur after the passage of $\Delta t$ units of slow simulation time. The plant growth rate is calculated by assuming a default steady rate of change to the size of each model component and decreasing this rate depending on the concentration of the limiting nutrient. Plant colour changes depend on deficient or toxic nutrient concentrations and the seasonal passage of slow simulated time. As mentioned in the previous section, three factors are considered when animating colour change: the leaf emergence age, final resulting colour, and pattern of the colour change on the plant. The *leaf emergence age* is the slow simulated time when the leaf emerges from the bud. For example, the first leaf to emerge would be the oldest leaf, the next leaf would be the second oldest, and so on.

Stage 3: FORM Domain Algorithm Design and Selection

To create the overall structure of a plant, a nested L-system (Rodkaew, Chuai-Aree, Siripant, & Lursinsap, 2004a) is used for the plant model. Recall that an L-system generates a string of alphanumeric characters that represent geometric shapes and that a nested L-system was used. To specify the L-systems, we used standard symbols (Rodkaew, Chuai-Aree, Siripant, &
Lursinsap, 2004a) and several nonstandard symbols (Derzaph & Hamilton, 2013b). Some standard symbols are F (for one unit of stem), + (for a positive rotation about the x-axis), and – (for a negative rotation about the x-axis). Some nonstandard symbols are A (for leaf), V (for vein), and P (for petiole). A geometric model of a plant is constructed by substituting 2D or 3D geometric objects for the symbols generated by the L-system. For the wheat models, we used a cylinder for the stem, as described previously using equation 4.1 and equation 4.2. For the linear leaf shape, a modified rectangle consisting of 9 triangle strips was constructed. For the wheat head a super ellipsoid was constructed using equation 4.29.

\[ x = r_x \cdot \cos^m \theta \cdot \sin^n \alpha \]

\[ y = r_y \cdot \sin^m \theta \cdot \sin^n \alpha \]

\[ z = r_z \cdot \cos^m \theta \] (4.29)

where \( r_x, r_y, \) and \( r_z \) are the semi-axes, \( m \) and \( n \) provide for variation of the ellipsoid shape into a variety of seed shapes, and \( \theta \) and \( \alpha \) are the amounts of rotation for the two axes, which are limited to \(-\pi/2 \leq \theta \leq \pi/2\) and \(-\pi \leq \alpha \leq \pi\), respectively. All variables represent the seed shape properties. If the parameters are altered, the shape of the seed is changed. For example if \( r_x, r_y, \) and \( r_z \) are equal, a sphere results, if the limit on rotation is \( \pi/2 \), a cone results, and if \( r_x, r_y, \) and \( r_z = \theta \), then an apple shape results. This equation is flexible enough to allow
the generation of many 3D shapes that resemble the seed shapes in plants, as shown in Figure 44.

The selections of the Stem object (PO: 009047), stem elongation property (TO:0006036), stem length property (TO:0000576), stem diameter property (TO:0020083), and stem colour property (TO:0000056) are implemented by the cylinder shape properties and its colour properties, as described previously. The selections of the Stem Angle (TO:0000577), Leaf Angle (TO:0000206) and Petiole (PO:0020038) Angle objects were implemented by the nested L-systems. The selection of the leaf object (PO: 0025034), leaf length property (TO:0000135), leaf width property (TO:0000370), leaf thickness property (TO:0000258), leaf colour property (TO:0000326), and leaf shape object (TO:0000492) were implemented by using the linear leaf shape and colour properties. The seed object (PO: 09010), seed size property (TO:0000391), seed colour property (TO:0000486), seed shape property (TO:0000484) were implemented by the wheat head shape and colour properties.

Stage 3: FUNCTION Domain Algorithm Design and Selection

The generic algorithm for animating plant growth with limiting nutrients and colour changes as is given as Algorithm 3:

1. Generate the initial geometric plant model using L-systems
2. Perform animation:
   a. Determine the current fast simulated time ($t_s$) and elapsed fast simulated time ($\Delta t_s$)
b. Determine the limiting nutrient \( (c_n) \) according to equations 4.30, 4.31, 4.32, and 4.33.

c. Based on the slow simulated time \( (T) \):
   - Determine current development stage, \( (s) \).
   - If growth occurs in current development stage, \( (s) \) this determines the addition of stems and leaves:
     - Determine the growth rate change \( (g_n) \) and the current limiting nutrient \( (c_n) \) based on elapsed slow simulated time \( (\Delta T) \) using equation 4.35.
     - Determine current growth rate of each component \( (l) \) using equation 4.36, where \( r_l \) represents the components: stem length \( (sl) \), leaf length \( (l) \), leaf radius \( (lr) \), leaf thickness \( (lt) \), seed radius along x-axis \( (sr_xt) \), seed radius along y-axis \( (sr_yt) \), and seed radius along z-axis \( (sr_zt) \).
     - Determine colour change required for every component \( (l) \) according to Table 7.

3. Draw each component based on the current frame time.

**Algorithm 3 Limiting nutrient and colour change.**

Algorithm 3 summarizes the approach described in this section. This algorithm can be specialized for any plant model of a target species by providing values for the parameters for the changes in developmental stages, growth,
colour, and geometric shape. For example, it can be specialized to wheat (*Triticum*). The following sections describe the animation of a wheat plant with respect to the FUNCTION domain list selections. The *Response to nutrient levels* (GO:0031667) object is described in three sections: the Limiting Nutrient section, Wheat growth section, and resulting Colour changes section. Interestingly, the *Growth* (GO:0040007) object is described in the Wheat Growth section because there exists a relationship between *Response to nutrient levels* (GO:0031667) and *Growth* (GO:0040007) objects such that a lack of nutrients slows or prohibits growth. The development stage objects, *Leaf development stage* (PO:0001050), *Leaf development* (GO:0048367), *Shoot system development stage* (PO:0025530) and *Shoot development stage* (GO:0048367) will be described in the Wheat Animation section.

**Limiting Nutrient**

When animating the effect of nutrients on the growth of a plant species, only the nutrients specifically influencing the growth of that species are considered. The concentration of every nutrient is determined and the relevant constants (*c*<sub>min</sub>, *c*<sub>opt</sub>, *c*<sub>tox</sub>) are obtained from the biological literature (McKenzie, 1998). If any nutrient is outside the range between *c*<sub>min</sub> and *c*<sub>tox</sub>, then the limiting nutrient is identified as the one with the least suitable concentration. Liebig’s law of minimums (Liebig, 1840; Liebig, 1855) states that growth is controlled by the availability of the limiting nutrient rather than that of all required nutrients (Jerz, 2017). An *unsuitability score* (*u*<sub>i</sub>) for each nutrient *i* is calculated based on Figure 45. Four cases are considered, as shown in equations 4.30 to 4.33:
\[ u_i = 0 \text{ if } c_{opt} \leq c_i < c_{tox} \quad (4.30) \]

\[ u_i = \frac{(c_{opt} - c_i)}{(c_{opt} - c_{min})} \text{ if } c_{min} \leq c_i < c_{opt} \quad (4.31) \]

\[ u_i = 1 + \frac{(c_{min} - c_i)}{(c_{tox} - c_{min})} \text{ if } c_i < c_{min} \quad (4.32) \]

\[ u_i = \min \left\{ 1 + \frac{c_{min}}{(c_{tox} - c_{min})}, 1 + \frac{(c_i - c_{tox})}{(c_{tox} - c_{min})} \right\} \text{ if } c_i \geq c_{tox} \quad (4.33) \]

Any concentration between \( c_{opt} \) and \( c_{tox} \) is perfectly suitable and so has an unsuitability score of zero (see equation 4.30). Any concentration between \( c_{min} \) and \( c_{opt} \) is somewhat suitable with an unsuitability score that is zero at \( c_{opt} \) and increases linearly to one at \( c_{min} \) (see equation 4.31). Any concentration outside of the range \( c_{min} \) to \( c_{tox} \) is unsuitable and is assigned an unsuitability score of more than one. In such cases, it is desirable that relative scores which can be compared between nutrients are obtained. Also it is desirable that the scores increase with the distance between the observed concentration and the desirable range, up to a maximum value when a concentration of zero exists. Thus, if the concentration is less than \( c_{min} \), then the score is one plus a ratio calculated between the distance from the observed concentration to the desirable range and the length of the interval from \( c_{min} \) to \( c_{tox} \) (see equation 4.32), which provides a relative score. An unsuitability score is assigned in a similar manner if the concentration is greater than or equal to \( c_{tox} \). In this case, the maximum score is limited to the same value as when the concentration is zero (see equation 4.33). The intuition is that no overabundance is worse than
the complete absence of the nutrient. Finally, the nutrient with the highest unsuitability score is chosen as the limiting nutrient; any ties are resolved by choosing a nutrient with the largest required concentration. Let \( n \) represent the limiting nutrient selected by the above procedure.

Figure 46 shows an example of identifying the limiting nutrient \( n \) where \( c_{\text{min}} = 1 \), \( c_{\text{opt}} = 2 \), and \( c_{\text{tox}} = 3 \) and the four possible limiting nutrients are numbered 1, 2, 3, and 4 with concentrations of 2.3, 1.5, 0.01, and 3.1, respectively. The concentration \( c_1 \) of the first limiting nutrient is in the most desirable range because \( c_{\text{opt}} \leq c_1 < c_{\text{tox}} \) and so the unsuitability score is 0 by equation 4.30. The second limiting nutrient has \( c_2 = 1.5 \), and so \( c_{\text{min}} < c_2 < c_{\text{opt}} \) Thus, according to equation 4.31, \( \frac{(c_{\text{opt}}-c_2)}{(c_{\text{opt}}-c_{\text{min}})} \) is used, resulting in an unsuitability score of 0.5. The third limiting nutrient has \( c_3 = 0.01 \). Since \( c_3 < c_{\text{min}} \), the unsuitability score is calculated as \( 1 + \frac{(c_{\text{min}}-c_3)}{(c_{\text{tox}}-c_{\text{min}})} \) by equation 4.32, which yields 1.495. Since the fourth limiting nutrient has \( c_4 = 3.1 \), then by equation 4.33, \( c_4 \geq c_{\text{tox}} \) and therefore the unsuitability score is the minimum of \( 1 + \frac{c_{\text{min}}}{(c_{\text{tox}}-c_{\text{min}})} \) and \( 1 + \frac{(c_4-c_{\text{tox}})}{(c_{\text{tox}}-c_{\text{min}})} \). The resulting unsuitability scores are 1.5 and 1.05, respectively. The minimum of these scores is 1.05. Overall, the limiting nutrient is nutrient 3 because \( c_3 = 0.01 \) has the greatest unsuitability score (1.495). Conceptually, as shown in Figure 46, concentration \( c_3 \) is farthest from the desirable range.
Wheat Growth

Growth is an important part of realistic animation of a wheat plant. The maximum height of a wheat plant can be up to 120 cm, depending on the species and variety. Growth in a wheat plant occurs in three places: from the tip of the wheat stem (the *apical meristem*), across the leaf, and at the top of the internode of the stem where a leaf is attached. The maximum rate of growth is predetermined assuming optimal nutrient concentrations. For example, the rate of growth for a stem is 0.78 cm / growth degree day (Villegas, Aparicio, Blanco, & Royo, 2001) where a *growth degree day* (*GDD*) is defined as the average of the high and low temperature for a day minus a base temperature specific to each plant species. For the case of wheat, the base temperature for wheat is 0°C (Miller, Lanier, & Brandt, 2001). To produce an animation of a growing wheat plant, the growth rate at every time step is calculated based on the limiting nutrient at that time step. The general limiting nutrient effect on growth is shown in Figure 47. It is calculated using equation 4.34, which was adapted from Ågren (2008), Ingestad (1992), and Knecht-Billberger (2006):

\[
g_n = \begin{cases} 
0 & c_n \in [0, c_{n,\text{min}}] \\
\frac{d_n\left(c_n - c_{n,\text{min}}\right)}{d_n\left(c_{n,\text{opt}} - c_{n,\text{min}}\right)} & c_n \in \left[c_{n,\text{min}}, c_{n,\text{opt}}\right] \\
\frac{d_n\left(c_{n,\text{opt}} - c_{n,\text{min}}\right)}{d_n\left(c_{n,\text{opt}} - c_{n,\text{tox}}\right)} & c_n \in \left[c_{n,\text{opt}}, c_{n,\text{tox}}\right] \\
0 & c_n \in \left[c_{n,\text{tox}}, \infty\right]
\end{cases} \quad (4.34)
\]

where \(d_n\) is the change per day to the rate of growth for nutrient \(n\). \(c_n\) is the current concentration of nutrient \(n\). \(c_{n,\text{min}}\) is the minimum concentration of
nutrient $n$ required for growth. $c_{n,\text{opt}}$ is the optimum concentration of nutrient for growth. $c_{n,\text{tox}}$ is the toxic concentration of nutrient $n$ that stops growth.

In practice many existing experimental results concerning the effect of limiting nutrients assume that an unspecified concentration of background nutrients is available in the soil. Also, as described for some experiments reported in Biology (El-Zanaty, El-Nour, & Shaaban, 2012), a concentration of 0 mg/kg of some nutrients is associated with a positive amount of growth. This situation is better modeled with equation 4.35.

$$g_n = \begin{cases} 
  g_{\text{min}} + 0 & c_n \in [0, c_{n,\text{min}}] \\
  g_{\text{min}} + d_n (c_n - c_{n,\text{min}}) & c_n \in [c_{n,\text{min}}, c_{n,\text{opt}}] \\
  g_{\text{min}} + d_n (c_{n,\text{opt}} - c_{n,\text{min}}) & c_n \in [c_{n,\text{opt}}, c_{n,\text{tox}}] \\
  0 & c_n \in [c_{n,\text{tox}}, \infty) 
\end{cases} \quad (4.35)$$

where $g_{\text{min}}$ is the growth rate in the presence of the background concentrations of nutrients and the other variables are as described above. This function is shown in Figure 46. In this graph, $g_{\text{min}}$ corresponds to the $y$-intercept.

To calculate the effect of limiting nutrients on the growth rate, the specific changes caused by the concentration of each limiting nutrient were collected from various papers. These changes were plotted, and a line was drawn using a least squares calculation. The slope of the line was computed, as shown in the last column of Table 6, and used as the slope ($d_n$) in equation 4.32. Recall that $d_n$ is the change to the rate of growth per day for nutrient $n$. For simplicity, we assumed that the change in growth rate could be linearly interpolated as a single rate from the known cases. For example, magnesium concentrations in the soil
of 0, 60, 120 and 180 mg/kg result in growth of 0.27, 0.36, 0.34 and 0.35 cm/day, respectively, as shown in Table 6. The least squares line through these points has a slope of $d_n = 0.00061$ cm-kg/day-mg and a y-intercept of 0.36 cm/day. If the limiting nutrient is magnesium at concentration ($c_n$) of 65 mg/kg, assuming $c_{n,min}$ is 0 mg/kg, the resulting growth rate, $g_n = 0.36 \text{ cm/gdd} + 0.00061 \text{ cm-kg/day-mg}(65 \text{ mg/kg}-0 \text{ mg/kg}) = 0.39965 \text{ cm/day}$.

The limiting nutrient growth rate, $g_n$, is then used in equation 4.36 to determine the plant growth rate.

$$r_T = r_{T-\Delta T} + g_n \Delta T \quad (4.36)$$

where $r_T$ is the current organ size, $r_{T-\Delta T}$ is the previous organ size, $g_n$ is current growth rate (adjusted by the limiting nutrient), and $\Delta T$ is the slow simulated elapsed time. The growth rate in this case is the change from the observed growth degree day mentioned earlier given that it is affected by a limiting nutrient.

The limiting nutrient ($n$) also influences colour change, including the pattern of colour change ($p_n$), the leaf emergence age ($a_n$), mottling ($m_n$), and the resulting colour change ($h_n$). Six common patterns, gathered from the biological literature, are presented in Table 7. Table 8 illustrates the colour changes that occur due to 12 specific limiting nutrients using the numeric values presented in Table 7.
Wheat Colour Change

To improve the realism of the animation of a growing plant, we also show the colour changes that occur in the wheat plant in the presence of a limiting nutrient. A summary of the colour changes is provided in Table 9, which describes the changes that occur for particular limiting nutrients. For example, suppose the limiting nutrient is nitrogen. According to Table 8, the leaf class order \((o_n)\) is 0. Based on Table 7, 0 indicates that the effect is applied first to the oldest leaves, i.e., those with a leaf emergence age of 0 (representing “old”). Leaf emergence is the time at which a leaf appears from a bud. The first emergent leaves are “old” whereas the last emergent leaves are “young”. Assuming that nitrogen is the limiting nutrient, the colour change continues through to the middle-aged leaves and finally to the young leaves. As shown in Table 8, the pattern of colour change \((p_n)\) when nitrogen is the limiting nutrient is tip to base \((p_n = 2)\). So, for the wheat model, the colour change for the leaves begins at the tip and moves along the leaf towards the stem. The effect consists of a colour change from normal (green) to pale yellow over frame time. For example, for a simple wheat model, the starting colour could be green \((0.0, 1.0, 0.0)\) and the target colour could be yellow \((0.75, 1.0, 0.0)\) and an incremental change could be made between them.

A colour change over frame time, corresponding to a specific pattern, can be produced by manipulating the RGB (Red-Green-Blue) colours of the vertices in a geometric model with a simple, sequential flow algorithm. In general, a flow algorithm is an algorithm that describes the propagation of values though a
graph (Ford & Fulkerson, 1956). For our animations, we require the propagation of colour changes in a geometric model of a leaf. In our colour change algorithm, the colour is changed by following sequential paths through the vertices over frame time. For example, the margin to middle leaf colour change involves changing the colour of the vertices in rows from the outsides (margins) of the leaf to the middle. Consider a linear leaf, as shown in Figure 47(a). For the left side of the leaf, the colour changes are applied to the vertical triangle strips numbered 1 to 4 from left to right, first to the odd, then even vertices. For the right side, an analogous process occurs from right to left for triangle strips 9 to 6. Lastly, the vertices of the triangle strip in the middle are changed.

For an oval shaped leaf, the colour is changed in pairs of vertices in the triangle strip. Consider the middle to margin colour change in the leaf model illustrated in Figure 47(b), which has six radial triangle strips. For a typical triangle strip, as shown in more detail on the right in Figure 47(b), first vertices 0 and 1 are changed, then vertices 2 and 3, and so on.

Vertex pairs are changed in tandem for all strips to show tip to base colour change in a linear leaf, which is similar to the method of middle to margin colour change in an oval shaped leaf. Given a geometric model with \( v \) vertex pairs, the number of pairs, \( v_c \), to change each day over \( s_{ΔT} \) days is given by equation 4.37.

\[
v_c = \frac{v}{s_{ΔT}}
\]  

(4.37)
For example, if a leaf with 100 vertices changes to yellow over 3 days \((s_{\Delta T})\) of slow simulated time, the rate of change is 50 vertex pairs \((v) / 3 \text{ days}\), which yields roughly 17 pairs \((v_c)\) changed to yellow each day.

To apply a typical green to yellow colour change due to a limiting nutrient during the growing period, a uniform daily increment from the initial colour to the target colour is applied to the RGB colour value of each vertex. The assumption for the colour change is that each vertex is changed in sequence until all vertices have been changed. Based on that assumption, the colour interval \((c)\) is calculated using equation 4.38:

\[
    c_i = \frac{l_c v_c}{f \Delta T c_T}
\]  

where \((l_c)\) is the total number of units of colour change desired, \((v_c)\) is the number of pairs to change per day, \((f)\) is the desired frame rate (frames /sec), \(\Delta T\) is the elapsed slow simulated time in days, and \((c_T)\) is the constant relationship between frame time (in seconds) and slow simulated time (in days). Suppose the total desired colour change, \((l_c)\), is 0.75 colour units for a pair of vertices. Nitrogen is the limiting nutrient. Also suppose that colour change should be applied to 50 vertex pairs and occurs over 6.25 days of slow simulated time. Then 8 pairs of vertices need to change to yellow per day. If 1 second represents one day of slow simulated time, then the colour change is an increment by 0.10 colour units per frame \((0.75 \text{ colour units} / \text{ pairs of vertices} \times (50 \text{ pairs of vertices} / 6.25 \text{ days}) / 60 \text{ frames per second})\). Whenever 0.75 units
of colour change have been applied to a pair of vertices, the colour change starts being applied to the next pair of vertices.

The mottling pattern is implemented by first choosing the number $k$ of mottled spots per leaf; $k$ is 100 in our demonstration. Then we randomly select $k$ initial vertices in the leaf. For each frame, we select one of the $k$ vertices, change the colour of up to 8 of its adjacent vertices, as illustrated in Figure 48, through random selection. This process is repeated as needed for the frame according to the number of vertices that should be done per frame. The result is a spotted colour change on the leaf.

To simplify the plant model, all leaves and tillers on the plants were assumed to appear at the same time and grow at the same rate because the emphasis was on the changes in colour for each leaf. In nature, the time when leaves and tillers appear is influenced by environmental factors such as nutrient availability e.g. nitrogen, carbohydrate production, and light availability (Evers & Vos, 2013; Evers, et al., 2005).

**Wheat Development Stages**

We assumed the wheat was spring wheat. A plant goes through a series of developmental stages until it reaches maturity. The majority of the stages producing visible changes to the plant should be animated to achieve realism. The sequence of stages for wheat has been described as follows: germination, seedling development, tillering, stem elongation, booting, heading, flowering, milk, dough and ripening (Saskatchewan Agriculture, 2013). Several of these
stages are illustrated in Figure 49 (University of Delaware, 2013). The *germination stage* occurs as the seed produces a root and a leaf. The *seedling development stage* occurs from the first leaf to the first tiller; usually the wheat plant has three leaves at the end of this stage. The time of leaf emergence and the appearance of the tillers is controlled by the amount of nutrients, carbohydrates, and light availability (Evers & Vos, 2013; Evers, et al., 2005). The *tillering stage* is where the stems are added but do not begin to grow; once the maximum number of stems have emerged, stem elongation begins. The *stem elongation stage* is where every wheat stem grows at its nodes, which are the points where the leaves connect to the stem; growth during this stage results in separation of the leaves and continues until the stem has reached its maximum length. The *booting stage* occurs when the wheat head starts to form and grow in the last leaf, which is called the *flag leaf*. The *heading stage* begins when the head tip of the wheat is visible above the flag leaf and finishes when the head has fully emerged. The *flowering stage (inflorescence)* is where the flower is developed; this stage ends with pollination of the flower. The *milk stage (anthesis)* occurs when seed kernels begin to form. The *dough stage* completes the kernel formation. During the *ripening stage*, which is the last stage, the kernel loses water. In total, the duration of wheat maturation is approximately 180 days for spring wheat and 190 days for winter wheat, as illustrated in Table 10 (Stapper & Fischer, 1990).
Stage 3: The ENVIRONMENT Domain Algorithm Design and Selection

Seasonal Change

For simplicity, the Seasonal object was implemented as one quarter of a year (91.25 days) for each season as previously shown in equation 4.26 and equation 4.27.

The various development stage objects for wheat from the FUNCTION domain were added to the seasonal calculations using a 180 day growing cycle. The 8 stages described in the previous subsection spans over the 180 days during spring, summer, and fall seasons. As development stages vary among wheat species and varieties, a generic variety is assumed for this demonstration. We assume the starting day for seed development begins on day 91.25 of the year of slow simulated time, with the others following: tillering on day 101.25, stem elongation on day 121.25, booting on day 141.25, heading on day 161.25, flowering stage on day 181.25, milking stage on day 201.25, and the ripening stage on day 221.25. We skipped the dough stage because it does not need to be animated because the relevant changes occur inside the head of the wheat. Each of the stages is assumed to end on the day before the next stage began, with the exception of the ripening stage which ended at 270.25 days. The total number of days is 180.

Wheat Nutrient Requirements

Wheat nutritional requirements are provided in Table 11 (adapted from (McKenzie, 1998)). The nutrients shown in the table affect growth and colour if
they are not available in the appropriate concentrations. The minimum \((c_{n,\text{min}})\), optimal \((c_{n,\text{opt}})\), and toxic \((c_{n,\text{tox}})\) values are given for these nutrients with respect to wheat. The first six entries describe the macronutrients; for these nutrients, concentrations are expressed as a percentage of the volume in the whole plant. The remaining six entries describe the micronutrients; for these nutrients, concentrations are expressed in parts per million (ppm) in the whole plant. These limits are used in the animation to directly influence the stem length, leaf length, and leaf colour, as described in the Section 4.3.3.

Summary of Wheat Animation Design

When the animation algorithm is applied to the wheat model, it produces each component of the plant based on time. For example, during the tillering stage, one tiller appears for every day that elapses until the required number of tillers has been produced. Next, during the elongation stage, the elongation of the stem (tiller) and leaves occurs. During these stages, if nutrients are not in the optimal range, plant growth and leaf colour are affected by the limiting nutrient.

In more detail, considering the general algorithm 3 given previously, we selected appropriate parameter values to produce the wheat animation algorithm given as Algorithm 4 below.

1. Generate the initial geometric plant model using L-systems. Note that all stems and leaves are generated with zero size at this time.

2. Perform animation:
• Determine the current slow simulated time \((T)\) and slow simulated elapsed time \((\Delta T)\).

• Determine limiting nutrient \((n)\):
  
  o If the concentration of any nutrient is outside of the range \(c_{\text{min}}\) to \(c_{\text{tox}}\):
    
    ▪ Find an unsuitability score \((u)\) for each such nutrient \((i)\), using Equations 4.30 and 4.31.
    ▪ Select the nutrient with the highest unsuitability score as the limiting nutrient \((n)\).
  
  o Else
    
    ▪ Find an unsuitability score \((u)\) for each such nutrient \((i)\), using Equations 4.32 and 4.33.
    ▪ Select the nutrient with the highest unsuitability score as the limiting nutrient \((n)\).

• Based the current slow simulated time \((T)\) and slow simulated elapsed time \((\Delta T)\):
  
  o Using the current time \((T)\), determine the current development stage \((s_i)\).
  
  o If growth occurs in the current development stage \((s_i)\):
    
    ▪ Determine growth rate change \((g_n)\) based on the concentration \((c_n)\) of the limiting nutrient using Equation 4.34.
- Based on the fast-simulated elapsed time ($\Delta t_s$), determine the size for each component ($l$) using Equation 4.35 (i.e. stems and leaves).
  - If the development stage ($s$) is at ripening:
    - For every component ($l$), change the colour from the current colour toward the fall colour according to the fast-simulated elapsed time ($\Delta t_s$) with respect to the total time of the ripening stage.
  - Otherwise, depending on the limiting nutrient ($n$), then for every component ($l$) that is a leaf:
    - Based on the current fast simulated time ($t_s$) and the order of effect by leaf emergence age ($o_n$), determine the emergence age ($a_{n,l}$) of leaves currently being affected.
    - If the emergence age ($a$) for leaf $l$ is equal to the currently affected emergence age ($a_{n,l}$):
      - If the colour change ($h_n$) is not yet complete, use pattern ($p_n$) and colour change ($h_n$) from Table 8 to change the colour.
      - Otherwise, use the given mottling ($m_n$).

3. Draw each component based on the current real-time frame rate ($t$).

**Algorithm 4: Wheat limiting nutrient and growth algorithm**

In particular, separate growth rates for stem length ($sl$), leaf length ($l$), leaf radius ($lr$), leaf thickness ($lt$), seed radius along x-axis ($sr_x$), seed radius
along y-axis \((s_{ry})\), and seed radius along z-axis \((s_{rz})\), were specified. The time-dependent parameters were: duration of development \((T_d)\) converted to frames per second using a conversion constant \((c)\), and a series of time ranges \((T_n)\) represented by start times in days of slow simulated time for each wheat development stage. As well, the size parameters of maximum stem length \((s_{max})\), maximum stem radius \((r_{max})\), maximum leaf length \((l_{max})\), maximum leaf radius \((l_{r_{max}})\), maximum leaf thickness \((l_{t_{max}})\), and three maximum seed radii \((s_{r_{x_{max}}}, s_{r_{y_{max}}}, \text{and} \ s_{r_{z_{max}}})\) were specified.

4.3.3 Stage 4: Coding the Animation

The coding of the animation was completed within Visual Studio with GLUT. The project contains the standard functions required for GLUT: an initDisplay, keyboard, reshape, display and main functions. The initDisplay function contained the set-up information for the menus, lighting, and background colour. The keyboard function contained code descriptions to respond to changes in the number of years (> key to increase by 1 year, and < key to decrease by 1 year), and requests to turn growth on and off (g key and j key).

The reshape function provided the perspective and display parameters to allow for the dynamic plant model to be rendered and thus seen. The display function contained the camera parameters, along with the wind, number of years, plant size parameters, and the calls to the three draw functions. The main function contained the setup for the window, calls to the other functions mentioned along with calls to the two L-system functions mentioned previously in
Section 4.2.3 i.e. calculateProductionRules and growPlant. Three other functions performed graphical substitutions for the plant; the drawPlant, drawPetiole and drawVein functions each took an L-string item and graphically substituted the corresponding shape for it, as mentioned previously in Section 4.2.3. The parameters that passed amongst them include the L-string, the current year, the growth rate, the limiting nutrient, and various plant size parameters.

Two additional functions, colour_changer, and switcher, were used to change the colour of the leaves. Colour changer function included the different cases to change the leaf colour according to the desired pattern: top to bottom, bottom to top, margin to middle, middle to margin, and mottled. The switcher function controlled the timing for the colour change with respect to the current time and the amount of colour change that was required at each vertex per frame, according to the colour vertex change formula described in the previous section. In addition, to create a realistic model for wheat, a new shape function for the wheat head was added, it was called through the L-string. The fruit shape was constructed using the parametric equations for a super-ellipsoid described by equation 4.28.

4.3.4 Results of the Animation

A series of experiments were run to determine the effects of limited nutrients on plant animation. Plant animations were produced using a Dell laptop PC with an Intel® Core ™ i7-2630QM CPU and a NVidia GeForce GT 550M graphics card. The operating system was Windows 7, 64-bit, the programming
language was C++, and the graphics libraries were OpenGL and GLUT. Video capture was performed using Microsoft Expression Encoder version 4. Segments of the captured video appear in through Figure 41 and Figure 43 Figure 51 through Figure 63.

We tested the colour change method with a simple wheat model with linear leaves, each composed of up to 9 triangle strips that were each composed of 100 vertices. We varied the number of leaves from 1,000 to 100,000 and the number of strips from 1 to 9, using the tip to base pattern. Each possibility was tried 20 times and the results were averaged. Our testing showed that CPU time increased from 7,500 ms to 55,000 ms as the number of leaves increased from 1,000 to 100,000, as is illustrated in Figure 50. In addition, as the number of strips increased from 1 strip to 9 strips per leaf, the processing time increased linearly. By controlling the number of leaves and number of strips on each leaf suitable performance can be obtained for a particular application.

The results of the experiments on animations over frame times for limiting nutrients are illustrated in Figure 51 through Figure 63. Figure 51 shows a leaf colour change from green to yellow with a tip to base pattern when nitrogen (N) is the limiting nutrient. Figure 52 shows an actual leaf with a tip to base pattern.

Figure 54 provides a base to tip pattern when boron is the limiting nutrient. Figure 54 shows a leaf colour change from green to yellow with a middle to edge pattern when calcium (Ca) is the limiting nutrient. Figure 55
shows a close up of a mid-stage for the same case. We could not find an image illustrating the effect of calcium deficiency on actual leaves.

Figure 56 shows a colour change with an interveinal, middle to edge pattern when manganese (Mn) is the limiting nutrient; This interveinal pattern results in striping. Molybdenum (Mo) gives the same pattern. Figure 57 and Figure 58 provides a comparison of middle to edge colour changes between an animated leaf (Figure 57) and actual wheat leaf (Figure 58) when manganese (Mn) is the limiting nutrient.

Figure 59 shows a leaf colour change from green to yellow over frame time with an interveinal, edge to middle pattern when potassium (K) is the limiting nutrient; iron (Fe) gives the same pattern. This interveinal pattern has similar striping to that observed in Figure 57. Figure 60 and Figure 61 provides an animated leaf (Figure 60) compared to an actual wheat leaf (Figure 61) for comparison when potassium is limited. Figure 62 shows the result of an animated leaf colour change from green to yellow with the mottled pattern when zinc (Zn) is the limiting nutrient; magnesium (Mg) gives the same pattern. Figure 63 shows mottling on actual wheat leaves when zinc is the limiting nutrient.

Lastly, the effects on growth of several limiting nutrients are shown in Figure 64 through Figure 78. Figure 64 and Figure 65 provide a comparison of height between an animated wheat plant (Figure 64) and a real wheat plant (Figure 65) when nitrogen is the limiting nutrient. In both Figure 64 and Figure 65, two plants are shown: a control plant on the left and the result of a limiting
nutrient on the right. Figure 66 through Figure 77 show similar comparisons where the limiting nutrient is potassium, phosphorous, magnesium, sulfur, iron and copper. Figure 78 shows an image of an animated plant with boron as the limiting nutrient; we were unable to locate a suitable image showing corresponding real plants. The observed reductions in the growth rate are in alignment with the growth rates per day shown in Table 6. Limiting calcium or zinc results in an increase in size (not shown), as expected from Table 6.

4.3.5 Discussion

The approach for animating nutrient effects on plants described in this section is the first comprehensive approach to the problem. Vos (2009) considered nutrient effects on animated plants. although attention was restricted to nitrogen. The idea of identifying a limiting nutrient as a first step toward animating nutrient effects is an original contribution of this author. This is the first time the macro and micronutrients are considered together to obtain a limiting nutrient. Once the limiting nutrient is found changes to the growth of the plant occur, in most cases restriction of growth.
Chapter 5  Conclusion

This chapter presents conclusions and suggestions for future research. Sections 5.1 to 5.4 describe conclusions with respect to the PLANI framework, animating wind effects on plant motion and growth, animating nutrition effects on plant growth and colour, and the Plant Creator software system, respectively. Section 5.5 gives potential topics for future research.

5.1 The PLANI Framework

The PLANI plant animation framework provides a way to use plant ontologies to guide the design, coding, simulation, and animation of dynamic plant models. The use of the ontologies when selecting relevant objects allows the design to proceed in a well-organized manner while considering biological factors. The key to the approach is the combination of the four interrelated domains (FORM, FUNCTION, ENVIRONMENT, and VIRTUAL WORLD) with current plant ontologies. The framework enables a designer to consider all aspects of animating a dynamic plant model by providing all relevant objects and parameters in a hierarchical format that provides linkages amongst the objects. It provides a structure for considering inclusion or exclusion based on four stages: (1) selection of required domains, (2) selection of objects and properties in the chosen domains, (3) selection or design of algorithms for the chosen objects and properties, and (4) coding of the algorithms, as necessary, to produce a dynamic plant model based on appropriate resource constraints for the application (gaming, biological simulation, or film). In addition, the framework provides a way to directly link each animation object to the corresponding biological object; this
linkage promotes consistency between the objects. For example, the Leaf object of the plant ontology is directly related to the leaf object of the plant model. In addition, radius, colour, and shape parameters of the plant trait ontology are directly related to the radius, RGB colour, and polygonal shape, respectively, of the leaf in the plant model. Regardless of the animation type (gaming, biological simulation, or film), a designer who is guided by the PLANI framework can produce a plant animation consistent with the stated design and implementation considerations.

The use of the framework was demonstrated with three examples in this thesis: in the design of the Plant Creator software system, in the animation of wind effects on plant motion and growth, and in the animation of nutrient effects on plant growth and colour. The last two demonstrations were biological simulations using specific dynamic plant models: a generic plant was used to illustrate wind effects and a wheat plant was used to illustrate nutrition effects. In contrast, the Plant Creator software system can be applied to create a variety of dynamic plant models. With this system, the designer can select properties of objects in the FORM domain (e.g., Stem, Petiole, Leaf, and Vein), FUNCTION domain (e.g., Growth and Motion), and ENVIRONMENT domain (e.g., Wind and Season) to specify a dynamic plant model and then use the interface to produce animations of the plants in scenarios specified by the properties of the VIRTUAL WORLD domain. As the designer changes the values of the properties, the changes are immediately reflected in the displayed models. Examples showed a
default plant, a species-specific example featuring the *Gingko biloba* tree, a tree with oval leaves, a tree with compound leaves, and a vine.

5.2 Animating Wind Effects on Plant Growth

Both motion and growth of branches and leaves are influenced by the wind. The Growth-Flow method proposed in this thesis and a journal article (Derzaph & Hamilton, 2013b) is the only method known for plant animation that considers wind influences on growth. This method, as illustrated in Figure 41, Figure 42, and Figure 43, includes the following changes to growth: decrease in branch elongation, increase in branch radius, change in branch growth angle, increase in leaf thickness, and decrease in leaf size. All of these effects were simulated over extended periods of fast simulated time. It was observed that prolonged exposure to high velocity wind resulted in a noticeable reduction of the height of an animated plant and an increase of in its radius. This work has influenced current research on wind effects on leaf growth (Pirk, Nise, Hadrich, Benes, & Deussen, 2014).

5.3 Animating Nutrient Effects on Plant Growth and Colour

The nutritional requirements of plants are well documented in biology, from both theoretical models and observations. Since the biological models are based on a standard growth rate for each species (assuming optimal concentrations of nutrients are available) and Liebig’s law of the minimum, we adopted a similar approach. The proposed method implements the influences of nutrient concentrations on plant growth and colour in animation. We use the
development stages \( (s_i) \), growth rate \( (g) \), limiting nutrient concentration \( (c_n) \), and colour \( (a_n, p_n, m_n, h_n) \) parameters to represent the development stages, appearance of the plant components over the stages, constrained nutrient concentration, and colour, respectively. The approach to colour change allows for a simplified function that directly manipulates the vertices representing the leaves. In addition, the approach allows more realistic colour changes to be incorporated with little computational overhead, which provides a potential advantage for games.

5.4 The Plant Creator System

The Plant Creator software system was implemented by following the PLANI framework and by reusing code created to demonstrate the effects of wind and nutrient on plants. It allows dynamic plant models to be generated for a variety of plants through user interaction. The system also provided the end-user with the capability to change the FORM domain properties, which directly changed the dynamic plant model. This allows the user to link the taxonomy to the proper form and save that form for future use. The system also allows for interaction with the ENVIRONMENT domain properties of macronutrient and micronutrient concentrations. The changes are immediately shown to the user.

5.5 Suggestions for Future Research

Further study is required to improve dynamic plant models. While growth is being simulated, the software should detect potential three-dimensional collisions between plant parts in advance and adjust growth accordingly. With
respect to wind effects, the influence of petiole and leaf motions should be added to the motions of the branches. Utilization of a global approach to incorporate leaf–petiole–branch interactions warrants further investigation.

Future work is also required to incorporate other environmental factors into the Growth-Flow method. Some factors to consider are light intensity (Reynolds, Foulkes, Furbank, Griffiths, & Kin, 2012), temperature (Wheeler, Hong, Ellis, Batts, & Morison, 1996), carbon dioxide and water availability (Levang-Brilz & Biondini, 2002; El-Zanaty, El-Nour, & Shaaban, 2012). The generic nutrient and Growth-Flow algorithms included in this thesis could be extended to incorporate variations in seasons and growth rates. In addition, the treatment of limiting nutrients could be enhanced by incorporating the multiple limitation of nutrients hypothesis rather than using a single limiting nutrient (Ágren, Martin Wetterstedt, & Billberger, 2012; Bloom, Chapin, & Mooney, 1985). In addition, future research could investigate giving masses and radii to the branches in the Growth-Flow method.

Further enhancements to the Plant Creator system could be accomplished. This software system could be easily expanded by including more objects in its four domains. For example, Fruit, Root (this object includes all underground parts for simplicity), Flower, and Seed objects could be added to the FORM domain, Response to Temperature, Response to Carbohydrate (through growth and cell building) (Vos, et al., 2009), and Response to Disease objects could be added to the FUNCTION domain, Temperature and Disease objects could be added to the ENVIRONMENT domain, and more complex
*Lighting* and *Topology* (geometric surface) objects could be used in the VIRTUAL WORLD domain.
List of References


International Conference on Artificial Reality and Telexistence - Workshops, pp. 200-204.


Appendices

Appendix A: Biology Detail

The biological concepts presented in this appendix are common knowledge available in various current textbooks on botany and physiology. This is not an exhaustive summary of plant biological concepts, but some are presented here as a convenience for readers lacking a background in biology.

A.1 Plant Structure

*Morphology* (the study of shape and structural characteristics of an organism) is relevant to animation of dynamic plant model structure (Raven, Evert, & Eichhorn, 2005). *Plant structure* is the form a plant has in the world. An *organ* is a structural and functional part. A typical plant is composed of several organs, e.g., leaves, stems (branches, stalks), roots, buds (root and lateral), fruits, thorns, bulbils, bulblets, spurs, seeds, and flowers as shown in Figure A-1 and Figure A-2. Each organ has a generic structure (see Figure A-3 for generic flower structure).

*Biological taxonomy* (classification) categorizes organisms into groupings based on anatomy, morphology and genetic characteristics. Currently accepted taxonomic groupings include domain, kingdom, phylum/divisions, class, order, family, genus and species (Mayr, 1998). The categorization of plants is dependent upon detailed examination of the patterns for specific organs and parts of organs. Plant species groupings are based on various characteristics
such as branching structures, and especially reproductive structures, and increasingly informed by genetic information. Rationale; genetic (DNA sequencing) is used a great deal today for determining/revising species (and higher) relationships. In Section 4.1, the demonstration of PlantCreator software uses the plant taxonomy to link a dynamic plant model to a plant species by allowing a user to select a plant species and create a plant form that matches that species. This section will summarize some of the plant organs that are relevant to the understanding of the PLANI framework and supporting the three demonstrations in the previous sections 4.1, 4.2, and 4.3.

A.1.1 Leaf Overview

A leaf is an organ that stops growing when it reaches a certain shape, size and structure determined genetically and influenced by the environment. A leaf is composed a lamina (blade) as shown in Figure A-4. In addition, a leaf may include a petiole. The lamina or blade provides energy and helps to transport water for the plant through photosynthesis, respiration and transpiration. The petiole functions to attach the leaf lamina to the shoot or stem of the plant.

The lamina and veins follow a consistent pattern generation which contributes to the division of plants within angiosperms, into the known groupings of today i.e. monocots and dicots. Each plant species is classified using a combination of its leaf morphology and leaf phyllotaxis (after classification with reproductive morphology), as explained in the following sections.
Leaf Morphology

Leaf morphology includes its lamina shape, thickness, base shape, apex shape, margin type, leaf form, and leaf surface. Each component of the leaf has generic patterns used to classify the plant (along with other organ structures). Knowledge of these patterns is required to specify dynamic plant model corresponding to realistic plants. Examples include acicular which is needle-like, orbicular is rounded shaped, ovate is egg-shaped, linear is several times longer than wide, lanceoloate is lance-shaped, and hastate which is spear-headed (Harlow. & Harrar, 1968) as shown in Figure A-5.

There are two forms of leaves: simple and compound (Glimn-Lacy & Kaufman, 2006). A **simple leaf blade** is undivided single units whereas a **compound leaf blade** is fully divided containing multiple units. Compound leaf arrangements follow common patterns such as pinnate and palmate, shown in Figure A-6.

Leaf classifications include differing leaf types amongst the divisions described in the next section.

Leaf Types

Each leaf division or type within the plant kingdom has certain characteristics (Glimn-Lacy & Kaufman, 2006). Some ferns have simple leaves
while others have compound leaves termed fronds. Angiosperms have leaves of a variety of shapes as shown in figure A-6. Lycophytes have micropyll leaves. Most conifers have leaf types that are evergreen and scale-like (*Juniperus*), evergreen and needle-like (*Pinus*), or needle-like and deciduous (*Larix*). Grasses, such as wheat, have monocot leaves containing a sheath. Dicot leaves vary in form and venation and are described in detail in the previous morphology section and further described in the following phyllotaxis section. Gingko leaves have a petiole and a blade with veins in a fan-shaped pattern. Grasses such as wheat have leaves with parallel venation, and are comprised of a distal narrow blade and a proximal sheath that surrounds the stem of the plant shown in Figure A-7. NOTE: Figure A-7 does not show a sheath clearly.

Leaf Phyllotaxis

As first defined by Hofmeister (1868), *phyllotaxis*, has two components: leaf arrangement at the node, and leaf arrangement along a stem. A **node** is the point of attachment of one or more leaves on a stem, the leaf axil (the angle formed at a node between the upper side of the leaf and the stem) contains one or more buds (or a branch, flower, or inflorescence produced by a bud).

The first component of phyllotaxis is the distribution of leaves at nodes. **Leaf arrangement** at a node can be alternate, opposite, whorled or rosulate as shown in Figure A-8. If one examines a plant for alternate leaf arrangement the node will contain only one leaf. An opposite leaf arrangement has two leaves per node at the same level on opposite sides of the stem. Whorled leaves have three
or more leaves per node. Rosulate plants have three or more leaves in a rosette circling the base of the stem nearest the soil. Leaves at subsequent nodes are directly above or rotated to some degree (depending on the species) relative to leaves below.

The second component of phyllotaxis is leaf orientation, the pattern the leaves on one node created with respect to the adjacent leaf containing nodes as shown in Figure A-9. Leaves are classified according to how they are oriented about the stem. Orthostichous leaves lie directly above one another when comparing leaves between nodes. If a node contains one leaf there are three different orientations: monostichous, distichous, and tristichous (Bell A. D., 1991). Monostichous orientation, one complete parallel row on one side of the stem, is rare in the plant kingdom and usually accompanied with a slight helix arrangement called spiromonostichous (Bell, 1994). A distichous orientation has two parallel rows and can be of either opposite or alternate arrangement. A tristichous orientation involves three parallel rows offset by 180 degrees. Another orientation is decussate, which has two sets of parallel rows. For this orientation, the leaves always have the opposite leaf arrangement.

Leaf phyllotaxis are influenced by genetic predisposition. Leaf morphology is influenced by the environment (light, nutrients, water, and predation), and developmental stages. These factors can alter each leaf’s appearance leading to slight differences throughout the plant that must be taken into account when modeling a plant described in the plant physiology section A.2.
A.1.2 Stems and Stalks Overview

The generic plant stem, also known as a stalk, is the main supporting structure (Glimn-Lacy & Kaufman, 2006). The stem structure is composed of nodes and internodes. The node is the point of attachment of one or more leaves, and the internode is section of stem between nodes. In trees, a stem is also known as a trunk. A stem also acts as a transport mechanism for nutrients and water, a storage organ for nutrients, provides some photosynthesis, enables responses to gravity and light, and produces new living material.

Stem Types

Stem types can vary amongst species and include above ground and below ground types. Above ground stem types for example include herbaceous stems e.g. climbers, runners and woody stems (University of Washington, 2015). Below ground stems include bulbs, tubers, and rhizomes: The shape for the majority of stems is cylindrical but some herbaceous plants have square or flat stems as shown in Figure A-10. Below ground stems are not explored in this thesis.

A.1.3 Trees and Shrubs Overview

A tree contains trunks, branches, leaves, fruit or seeds, and roots. A shrub contains branches, leaves, fruit or seeds, and roots. The difference between a tree and a shrub is that a tree has one main stem with branches, while a shrub has several stems, many of which are approximately the same size.
Branches are structural members of a scrub or tree. In trees, branches are connected to, but not part of, the trunk whereas shrubs do not have a trunk yet have a continuum along the shrub growth form morphology. Leaves were described in the leaf overview section A.1.1. Roots are excluded from this thesis as animation focuses on above ground design.

One characteristic in determining tree or shrub grouping is the crown shape. **Crown shape** is determined by the size, ratio of height to width, and overall shape of its silhouette. The shape of the crown is somewhat determined by access to light: where light is more from above, trees tend to be more broad than tall, and, where light is more from the side trees tend to be more tall than broad.

The early computer models of tree crowns and shrubs use cones and ellipsoids (Norman & Welles, 1983) or convex polyhedra (Cluzeau, Dupouey, & Courbaud, 1995). Recently, crown shape has been further modeled considering environmental conditions such as through xylem and phloem transport (Nikinmaa, Sievänen, Perttunen, & Hölttä, 2013), study of clumping in Scots pine (Stenberg, Möttus, Rautiainen, & Sievänen, 2013), and evaluation of PipeQual model for Norway spruce (Kalliokoski, Mäkinen, & Mäkelä, 2013) based on the basic shapes mentioned. These shapes need to be considered when generating or creating trees in animation shown in Figure A-11 and Figure A-12.
Tree and Shrub Branching Types

The branching type of the tree considers two components: symmetries (lateral and longitudinal) and the type of shoot growth. Symmetries are based on (1) the preferential development of lateral axes on a vertical parent axis (longitudinal symmetry: acrotonic, mesotonic, basitonic shown in Figure A-13 (Bell A. D., 1991)) and (2) the preferential development of lateral axes on a horizontal, curved or slanted parent axis (lateral symmetry: hypotonic, epitonic, amphitonic) (Schweingruber, et al., 2007). Shoot growth is further explored in the growth section.

Most branches are arranged in a spiral around the stem/stalk, or trunk and can include patterns such as acrotony, basitony, and mesotony as shown in Figure A-13. In **acrotonic** architecture, the distal branches grow more vigorously. **Basitonic** architecture is where the proximal (closest to the soil) branches grow more vigorously, and **mesotonic** architecture is where the branches near the middle grow more vigorously (Bell A. D., 1991). A common name, witches'-broom, is used to describe an abnormal bushy growth of stems that seem to originate from one point (Glimn-Lacy & Kaufman, 2006).

Branching forms were categorized into 19 overall types by Alexander von Humboldt in 1808. Hallé and Oldeman expanded to 24 architectural types in the 1970s based on tropical trees (Hallé & Oldeman, 1970a; Hallé, Oldeman, & Tomlinson, 1978). These 24 tree architecture models were later found to
encompass many more types of trees and became the base for generic patterns of tree shape that can be used within computer animations.

A.1.4 Herbaceous Plant Overview

The classification of a plant as herbaceous is mainly due to their stem; an herbaceous plant does not contain bark on the stem (Glimn-Lacy & Kaufman, 2006) for example grasses such as wheat. Herbaceous plants have live vegetative stems rather than woody stems that are mainly dead tissue, shown in Figure A-14. Herbaceous plants also contain the organs of a generic plant: leaves, stem, flowers, seeds, and roots. The majority of herbaceous plants tend to be either vines or small plants. Herbaceous leaf phyllotaxis contains an arrangement (alternate, opposite or whorled) and has one type of orientation (e.g. monostichous, distichous, or tristichous (Bell A. D., 1991)). Herbaceous plants are the annuals, perennials, vines, and herbs such as wheat, tulips, virginia creepers, and mint.

A.2 Plant Physiology

Plant physiology includes the functions required for a plant’s sustainment. Physiological processes contribute to survival and provide nutrients, water, and energy to cells and organs through photosynthesis, respiration, transport, growth, development, and environmental response. Photosynthesis produces the energy the plant needs through absorption of light. Respiration produces the plant’s energy needs without the need for light. Transport ensures nutrients and
hormones can reach each cell for survival. **Growth** results in an increase in size of the plant. **Development** involves the appearance and maturation of various plant organs throughout a plant’s lifetime. **Environmental response** occurs when a plant responds to external factors such as water, oxygen, nutrients and wind. There are other chemical processes that exist for the protection of a plant to ward off external attacks from insects and disease which are not part of this thesis. In addition, only growth, development and environmental response will be described in the subsequent sections as they form the biological basis for this thesis.

### A.2.1 Growth

**Growth** begins at the germination of a seed and continues until a full-grown plant results. **Germination** occurs within a seed or a spore when seed dormancy is terminated. **Seed dormancy** is terminated when certain conditions such as, water, temperature and oxygen levels are met. Some seeds require further conditions such as freezing of the seed coat, heat from a fire (cones), abrasion of sand or other particles, or require digestion (hydrolysis) in an animal’s gut. The seedling first develops roots then the shoot emerges from the embryo and follows the light to arrive above the soil. The embryonic and young show contains one or two (many in most gymnosperms) embryonic leaves that produce the energy required for the young plant. Shoot growth (one stem) follows either sympodial or monopodial growth patterns. **Sympodial growth** occurs when the tip of the main stem or branch dies or transforms into a specialized structure (e.g. a reproductive structure); further growth then occurs from axillary
Meristems. **Monopodial growth** occurs when the tip of the shoot does not die or transform but continues to grow.

**Meristems** are regions responsible for growth in a plant and are present within root tips, shoot tips, and leaves and comprise vascular and cork cambiums (Nabors, 2004). Growth through the meristems occurs at the tip of each shoot, at the tip of each root, and, within vascular and cork cambium for growth of veins and barks (OpenStax College, 2013). The meristems produce cells that determine the length of each stem or root and the type of organ produced: leaf tissue, further stem tissue, bark, or root tissue as part of a bud on a stem, or a root tip on a root. In some plants, only the primary or apical meristem at the end of a stem or root grows (Hangarter, 2000; Strasburger, Noll, Schenick, & Karsteu, 1908). **Tropic response** occurs when growth is changed in a certain direction, for example away from an object. **Nastic response** occurs is any direction, for example when plants do not get enough water they wilt.

The **shoot apical meristem** produces leaf, stem, and flower primordia, shown in Figure A-15. Very generally, just below the apical meristem there are three regions of cells, the outer region produces the epidermis, the middle region produces the central tissues of the stem or root, and the inner region produces the conductive tissues of xylem and phloem. The vascular and cork cambiums (lateral meristems) of plants with secondary growth (gymnosperms and many angiosperms) produce secondary tissues (secondary xylem and phloem and periderm) and thus an increase in stem diameter.
A.2.2 Development

**Development** is the process of growth and the differentiation of plant tissues and organs over a plant’s lifetime, influenced by genetic and environmental factors. The seasonal changes and environmental factors influence the appearance of various organs such as leaves, flowers, fruit during plant development. This section describes the leaf, stem, shoots, and tree development relevant for the dynamic plant models discussed in Chapter 4.

A.2.2.1 Leaf Development Stages

Leaves grow from a leaf primordium on the apical meristem of shoots. Leaves have several meristematic areas called intercalary meristems, i.e. meristems inserted between primary tissues in a position different from that of lateral meristems (cambiums) and grow along three dimensions: proximal-distal, medial-lateral, and adaxial-abaxial. As the leaf matures it unfurls to its full size ready to provide the plant with the energy it produces.

Leaf developmental stages are influenced by cell division, cell elongation, and cell differentiation. The way they develop from the beginning to a leave’s death categorizes an individual plant in being homoblastic or heteroblastic depending on the differences among leaves on a plant i.e.between juvenile and adult leaves. In a **homoblastic** plant all the leaves are similar generally in shape, while in a **heteroblastic** plant juvenile and adult leaves substantially differ.
morphologically between juvenile and adult leaves a plant. NOTE: Insects can attack a leaf and result in leaf damage altering its individual structure.

A.2.2.2 Stem Development

Stems grow from a shoot originating from a apical meristem. As an apex grows from the activity of an apical meristem, it produces an axis (a linear stem structure from origin to apical meristem or extremity) (Millet, Bouchard, & Edelin, 1999). The axis may include monopodial or sympodial branching points. Monopodial axis is one that can continue to grow indefinitely, although it may become inactive or dormant at times. A monopodial axis can refer to the original stem of a plant or to a branch. Thus, in L-systems terminology, a “monopodial axis is a sequence of branch segments, each of which extends it predecessor in the terminal position” (Prusinkiewicz & Remphrey, 2000). However, in many plants the axis that grows from an apical meristem dies or transforms (or has become dominated) at some point, stopping growth in length of that axis. Growth then occurs from axillary meristems, forming branches, this is referred to as sympodial branching. A Sympodial axis is one that grows until it dies or transforms, and can refer to the original stem of a plant or to a branch. In L-system terminology “Sympodial branching indicates that the child axis or axes appear in the lateral position” (Prusinkiewicz & Remphrey, 2000). Continuing in L-system terminology, “If a dominant lateral apex takes over the role of the terminal apex that has aborted, produced a terminal inflorescence, or has become dominated, the sympodial branching produces a pseudo-monopodial
A.2.2.3 Tree Development

Tree development is the collection of shoot, stem, and leaf growth and development over annual cycles (OpenStax College, 2013). The tree starts as a seed, root sprout, or other forms of vegetative (asexual) reproduction and grows annually through the addition of vegetative lateral and apical branches and stem radius. Lateral branch growth occurs horizontal to the ground whereas apical branch growth occurs at the main axis of the stem. In addition, at various locations along the branch some trees develop annually to produce inflorescence structures that mature to fruit or seeds to provide for additional trees the next season. An *Inflorescence structure* is a group of flowers arranged in a structure based on a stem. Increase in the diameter of a tree occurs through the proliferation of cells in two secondary meristems (cambiums). Stem radius increases through this proliferation of cells outward adding the radius through a ring structure.

A.2.2.4 Herbaceous Plant Development

Herbaceous plant development has three types of development: annual, biennial and perennial. Annual plant development occurs where the plant (with stem and leaves) grows within one season to produce one or more flowers and ripens into fruit containing seeds. Biennial plant development is where a plant requires two seasons to complete the vegetative stage to ripen into fruit(s) containing seed(s).
in the second year. Perennial plant development is where a plant regrows every season without the need for reseeding.

A.2.3 Environmental Response

Environmental responses need to be considered when creating viable simulations (Room, Hanan, & Prusinkiewicz, 1996). **Tropism** is a plant's response to the surrounding environment dependent upon the direction of the stimulus. **Nasties** are a plant’s response to the surrounding environment not dependent on the direction of stimulus. (Wilson & Gartner, 1996; Timell, 1986; Wilson, 1998). The environmental responses of interest are the nutrient availability effect on growth, colour changes resulting from the limiting nutrient, and the effect of wind on growth and movement.

A.2.3.1 Nutrient Availability and Physiological Impact

Each nutrient influences the physiology of the plant, as explained in
Table 12, (McCauley, Jones, & Jacobsen, 2009; Murdock & Call, 1999; Tucker, 1999; Uchida, 2000). Optimizing nutrient concentrations and the ratio of nutrients to each other ensures the support of metabolism and growth. A summary of the optimal concentrations of five key nutrients (N, P, K, Ca, Mg) for the major types of plants is given in
Table 13, where the concentration of each nutrient is given relative to the nitrogen concentration (Knecht & Goransson, 2004). A similar summary for wheat with respect to a larger group of nutrients is provided in Table 14. The nutrient ratios are based on the Redfield ratio (Redfield, 1958) and further analysis of terrestrial plants (Knecht & Goransson, 2004). For instance, the optimal concentrations of carbon, oxygen, and hydrogen are similar to each other, whereas, the ratio of nitrogen (N) to phosphorous (P) is approximately 10:1 (Knecht & Goransson, 2004; Salisbury & Ross, 1992). If nutrients are deficient, the plant responds through growth reductions and colour changes. Lack of calcium affects the meristematic regions, killing the terminal buds and root tips. Phosphorous deficiency will stunt growth of a plant and darken leaves.

A.2.3.2 Nutrient Availability and Colour Changes

The colour of a plant is caused by the generation of pigments such as chlorophylls (green), xanthophylls (yellow) and carotenoids (red and orange) in plant cells for support of photosynthesis. The green colour of the leaf is due to the higher concentration of chlorophyll in relationship with the other pigments. When nutrients are deficient, chlorosis occurs. Chlorosis refers to reduced generation of chlorophyll resulting in reduced green colour, allowing other pigments within the leaf to become visible. Chlorosis generally changes the colour of leaves from green to yellow and eventually white depending on type of plant, the duration of nutrient availability, the leaf emergence age (i.e., the relative age of the leaves compared to other leaves on the plant), and the degree of deficiency, as described in Table 15. After chlorosis occurs, if the nutrient
deficiency continues, necrosis usually follows. *Necrosis* occurs when the plant is stressed to the point where cells die, and tissues become brown.

A.2.3.3 Wind, Growth, and Movement

*Wind* is air in motion resulting from change in pressure between layers of air that are either warmer or cooler than each other. Wind produces a Coriolis effect which determines prevailing wind directions, and generally plant growth responses are to prevailing wind direction. Wind affects plant development, growth, and movement. Wind is known to influence plant development including changing branching structure (Marshall, 1998; Peltola, 1996). Several studies have been conducted that conclude plant stem growth is reduced by wind (Jaffe, 1973; Neel & Harris, 1971; Salisbury, 1963; Smith & Ennos, 2002). Wind also affects stems or branches to thicken providing more strength to the stem or branch, restricts their elongation, and changes the direction of their growth. In response to wind, leaves thicken and round their shape (Anten, Alcala-Herrera, Schieving, & Onoda, 2010; Flückiger, Oertli, & Flückiger-Keller, 1978). Tree branch motion in response to wind depends upon its tensile strength, the size of its surface area facing the wind, and its mass (James, Haritos, & Ades, 2006; Vogel, 1989). Branch movement is observed as a swaying (back and forth) motion, which is also evident in leaf movement. In addition, leaves display a
twisting motion in response to the wind and a vibration motion when the wind is tangent to the blade (Zioma, 2007).

A.2.3 Ecology

The interaction between biotic (living) and abiotic (non-living) components and processes within a plant’s ecosystem regarding consumption and provision of compounds should be considered in dynamic plant model creation to provide for a more accurate plant model. Ecological phenomena affect plant growth, nutrient availability, and environmental response. For example, plant density self-thinning (Legendre & Legendre, 1993), clustering (Dale M. R., 1999), energy flow (such as solar, geothermal) through trophic levels (feeding relationships), competition, and abiotic cycles (water, wind), impact growth, biological physiology, and biological morphology of plants.
Appendix B: Figures

Figure 1: Turtle example for F-F+F

Figure 2: Vine-like branches created in L-Studio 4.0
Figure 3: Nine branching structures produced by varying the constants of a parametric D0L-system (Prusinkiewicz, Hammel, Hanan, & Mĕch, 1996)

Figure 4: Mesotonic tree example (Prusinkiewicz, Hammel, Hanan, & Mĕch, 1996)
Figure 5: Tree Crown parameters (Cescatti, 1997)

Figure 6: Schematic representation of two phytomers (n – 1 and n) of the wheat (Triticum aestivum) plant (Evers & Vos, 2013).
Figure 7: Petiole movement model

Figure 8: Leaf rotation model

Figure 9: Framework for interaction between plant and environment from Mêch and Prusinkiewicz 1996
Figure 10: Cutting of a tree derivation steps 2 through 9 (Prusinkiewicz, Hammel, Hanan, & Mĕch, 1996)

Figure 11: A pruning of a branching structure from (Prusinkiewicz, Remphrey, Davidson, & Hammel, 1994a)
Figure 12: Side view of tiger lily flower created using Maya

Figure 13: Top view of tiger lily flower created using Maya
Figure 14: Plant ontology
Figure 15: Plant anatomical entity
Figure 16: Plant trait ontology

**Term Name:** plant trait

**Term Accession:** TO:0000387

**Aspect:** plant_trait_ontology

**Synonyms (2):**
- plant phenotype, plant quality

**Definition:** A measurable or observable characteristic of a cellular component (GO:0005575), biological process (GO:0008150) or molecular function (GO:003674) that is part of, or has participated in, a plant anatomical entity (PO:0025131) and/or a plant structure development stage (PO:0009012).

**Comment:** None

**Derivation:**
- plant trait (TO:0000387)
  - anatomy and morphology trait (TO:000017)
  - stature or vigor trait (TO:0000133)
  - stress trait (TO:0000164)
  - other miscellaneous trait (TO:0000182)
  - biochemical trait (TO:0002977)
  - growth and development trait (TO:0000352)
  - yield trait (TO:0000371)
  - sterility or fertility trait (TO:0000392)
  - quality trait (TO:0000557)

**Database Cross-References (8):**

**Parents:**

**Children (9):**

**Associations:**
Figure 17: Gene ontology

<table>
<thead>
<tr>
<th>Term Name</th>
<th>biological_process</th>
</tr>
</thead>
<tbody>
<tr>
<td>Term Accession</td>
<td>GO:0008150</td>
</tr>
<tr>
<td>Synonyms (5)</td>
<td>biological_process, biological_process, biological_process unknown, GO:0003004, GO:0017812, physiological_process</td>
</tr>
<tr>
<td>Definition</td>
<td>Any process specifically pertinent to the functioning of integrated living units: cells, tissues, organs, and organisms. A process is a collection of molecular events with a defined beginning and end.</td>
</tr>
<tr>
<td>Comment</td>
<td>None</td>
</tr>
<tr>
<td>Source Ontology Database</td>
<td>The GO browser of Gene Ontology Database</td>
</tr>
</tbody>
</table>

**Derivation**

- biological_process (GO:0008150) #805:
  - reproduction (GO:0022656) #74
  - cellular process (GO:0009987) #33228
  - cell biology (GO:0022416) #16
  - biological diseases (GO:0005620) #5
  - growth (GO:0006281) #210
  - cellular process (GO:0048727) #364
  - reproductive process (GO:0022414) #0
  - biological warfar (GO:0003661) #0
  - organ (GO:0022383) #3
  - multicellular organismal process (GO:0048727) #0
  - development process (GO:0048731) #999
  - formation (GO:0048731) #23
  - single-organism process (GO:0048731) #0
  - single-cellular process (GO:0048731) #0
  - negative regulation of biological process (GO:0048518) #0
  - negative regulation of biological process (GO:0048518) #0
  - response to stimulus (GO:0007262) #219
  - localization (GO:0005488) #21
  - establishment of localization (GO:0048518) #0
  - multi-organism process (GO:0048731) #0
  - biological regulation (GO:0048518) #0
  - cell parts component organization or biogenesis (GO:0007163) #0

**Database Cross-References (2)**

| Parents (0) |
| Children (22) |
| Associations (8,051) |
Figure 18: Plant framework domains with process flows
Figure 19: PLAN Plant Animation Framework Ontology
<table>
<thead>
<tr>
<th>FORM</th>
<th>FUNCTION</th>
<th>ENVIRONMENT</th>
<th>VIRTUAL WORLD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Based on Plant anatomy and morphology trait (TO: 0000017) and Plant anatomical entity (PO: 0025131) ontologies)</td>
<td>Based on Gene biological process (GO:0008150), growth and development trait (TO:0000357) and Plant structure development stage (PO: 0009012) ontologies</td>
<td>Based on plant environment (EO:0007359) ontology</td>
<td>Based on animation components</td>
</tr>
<tr>
<td>• Leaf Vein (PO:000048)</td>
<td>• leaf elongation rate (TO:0000360)</td>
<td>• leaf width (TO:0000370)</td>
<td>• World Object</td>
</tr>
<tr>
<td>• Leaf (PO:0025084)</td>
<td>• leaf width (TO:0000370)</td>
<td>• leaf length (TO:0000135)</td>
<td>• WorldInf</td>
</tr>
<tr>
<td>• leaf elongation rate</td>
<td>• leaf angle (TO:0000290)</td>
<td>• leaf thickness (TO:0000258)</td>
<td>• 2D conversion</td>
</tr>
<tr>
<td>(TO:0000360)</td>
<td>• leaf color (TO:0000326)</td>
<td>• leaf angle (TO:0000290)</td>
<td>• Polygon</td>
</tr>
<tr>
<td>• leaf thickness (TO:0000258)</td>
<td>• leaf shape (TO:0000492)</td>
<td>• leaf angle (TO:0000290)</td>
<td>• Transform</td>
</tr>
<tr>
<td>• Leaf sheath (PO:0020104)</td>
<td>• leaf sheath diameter (TO:0000642)</td>
<td>• translation</td>
<td>• translation</td>
</tr>
<tr>
<td>• leaf sheath diameter</td>
<td>• leaf sheath length (TO:0002689)</td>
<td>• rotation</td>
<td>• rotation</td>
</tr>
<tr>
<td>(TO:0000642)</td>
<td>• leaf sheath width (TO:0002721)</td>
<td>• scale</td>
<td>• scale</td>
</tr>
<tr>
<td>• leaf sheath length</td>
<td>• leaf sheath color (TO:0002724)</td>
<td>• light</td>
<td>• Light</td>
</tr>
<tr>
<td>(TO:0002689)</td>
<td></td>
<td>• light position</td>
<td>• colour</td>
</tr>
<tr>
<td>• leaf sheath width</td>
<td></td>
<td>• light aim direction</td>
<td>• normal</td>
</tr>
<tr>
<td>(TO:0002721)</td>
<td></td>
<td>• Viewpoint</td>
<td>• Viewpoint</td>
</tr>
<tr>
<td>• leaf sheath color</td>
<td></td>
<td>• look at</td>
<td>• Topography</td>
</tr>
<tr>
<td>(TO:0002724)</td>
<td></td>
<td>• Time</td>
<td>• Time</td>
</tr>
</tbody>
</table>
Figure 21: FORM selection for the stem example

FORM
Based on Plant anatomy and morphology trait (TO: 0000017) and Plant anatomical entity (PO: 0025131) ontologies
- Stem (PO: 0009047)
  - stem elongation (TO: 0006036)
  - stem length (TO: 0000576)
  - stem width (TO: 0001035)
  - stem color (TO: 0000056)
  - stem diameter (TO: 0020083)
  - stem strength (TO: 0000051)
  - stem angle (TO: 0000577)

FUNCTION
Based on Gene biological process (GO: 0008150), growth and development trait (TO: 0000357) and Plant structure development stage (PO: 0009012) ontologies

ENVIRONMENT
Based on plant environment (EO: 0007359) ontology

VIRTUAL WORLD
Based on animation components
- World Object
  - position
- WorldInf
  - background
  - 2D conversion
  - Polygon
- Transform
  - translation
  - rotation
  - scale
- Light
  - light position
  - light aim direction
- Material
  - color
  - normal
- Viewpoint
  - look at
- Topography
- Time
Figure 22: FORM selection for the apical meristem
Figure 23: FUNCTION domain selection example
<table>
<thead>
<tr>
<th>FORM</th>
<th>FUNCTION</th>
<th>ENVIRONMENT</th>
<th>VIRTUAL WORLD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Based on Plant anatomy and morphology trait (TO: 0000017) and Plant anatomical entity (PO: 0025131) ontologies)</td>
<td>Based on Gene biological process (GO: 0008150), growth and development trait (TO: 0000357) and Plant structure development stage (PO: 0009012) ontologies</td>
<td>Based on plant environment (EO: 0007359) ontology</td>
<td>Based on animation components</td>
</tr>
<tr>
<td>- Stem (PO: 0009047)</td>
<td>- Growth (GO: 0040007)</td>
<td>- Abiotic (EO: 0007191)</td>
<td></td>
</tr>
<tr>
<td>o stem elongation (TO: 0006036)</td>
<td>o Locomotion (GO: 0040011)</td>
<td>o Seasonal (EO: 0007027)</td>
<td></td>
</tr>
<tr>
<td>o stem length (TO: 0000357)</td>
<td>o Response to Stimulus (GO: 00050896)</td>
<td>o Radiation regime (EO: 007151)</td>
<td></td>
</tr>
<tr>
<td>o stem width (TO: 0001035)</td>
<td>o Response to Chemical (GO: 0042221)</td>
<td>o Temperature (EO: 0007175)</td>
<td></td>
</tr>
<tr>
<td>o stem color (TO: 0000506)</td>
<td>o Response to Nutrient (GO: 0007584)</td>
<td>o Water (EO: 0007198)</td>
<td></td>
</tr>
<tr>
<td>o stem diameter (TO: 0020083)</td>
<td>o Response to External Stimulus (GO: 0009991)</td>
<td>o Gaseous (EO: 0007023)</td>
<td></td>
</tr>
<tr>
<td>o stem strength (TO: 0000051)</td>
<td>o Response to Nutrient Levels (GO: 0031667)</td>
<td>o Chemical regimen (EO: 0007189)</td>
<td></td>
</tr>
<tr>
<td>o stem angle (TO: 0000577)</td>
<td>o Shoot System Development (GO: 0048367)</td>
<td>o Nutrient regimen (EO: 0007241)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>o Shoot System Development stage (PO: 0025527)</td>
<td>o Physical (EO: 0007316)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>o Wind (EO: 0007382)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>o Gravity (EO: 0007146)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>o Biotic (EO: 0007357)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>o Pest/Pathogen/Animal/Plant (EO: 0007124)</td>
<td></td>
</tr>
</tbody>
</table>
| | |   - World Object
| | |   o position
| | |   o WorldInf
| | |   o background
| | |   o 2D conversion
| | |   o Polygon
| | |   o Transform
| | |   o translation
| | |   o rotation
| | |   o scale
| | |   o Light
| | |   o light position
| | |   o light aim direction
| | |   o Material
| | |   o colour
| | |   o normal
| | |   o Viewpoint
| | |   o look at
| | |   o Topography
| | |   o Time |
Figure 25: PLANI Plant Animation Framework Composition
Figure 26: Various plant forms utilizing the framework with formulae (original in colour).
Figure 27: The Plant Creator system user interface
Figure 28: System database schema
Figure 29: Species specific example
Figure 30: Tree with oval leaves
Figure 31: Tree with compound leaf
Figure 32: Vine example
Figure 33: Palmate with leaf variation
Figure 34: Variation of leaf radius along the xy plane
Figure 35: Branch movement

Figure 36: Branch segments and nodes.
Figure 37: Blade bending in response to wind.

Figure 38: Petiole bending in response to wind
Figure 39: Leaf twisting in response to wind.

Figure 40: Leaf vibration in response to wind.
Figure 41: Response of petiole and leaves to wind: consecutive frames of animation.

Figure 42: Year seven motion in response to wind, images taken four seconds apart (16 hours of slow simulated time).

(a) With no wind
(b) With wind

Figure 43: Growth for two seconds (8 hours of growth in slow simulated time) with and without wind.

a) apple
b) cucumber
c) pumpkin
d) grape
e) melon
f) wheat
Figure 44: Examples of fruit and seed shapes.

Figure 45: Effect of a limiting nutrient on growth

Figure 46: Example of limiting nutrient calculation
Figure 46: Effect of a limiting nutrient on growth in experiments
Figure 47: Leaf examples with triangle strip tessellation

Figure 48: Adjacent vertices example

Figure 49: Zadoks wheat development scale (Royo & Villegas, 2011; Zadoks, Chang, & Konzak, 1974)
Figure 50: Average processing speed for colour change (original in colour)

Figure 51: Animation of colour changes over a series of days when nitrogen (N) is the limiting nutrient during spring (original in colour)

Figure 52: Tip to base colour change an actual wheat leaf control of left and nitrogen limiting nutrient on right (adapted from (Snowball & Robson, 1991)) (original in colour)
Figure 53: Base to tip colour change in an animated wheat leaf when boron is the limiting nutrient (original in colour)

Figure 54: Animation of middle to edge colour changes over a series of days when calcium (Ca) is the limiting nutrient during spring (original in colour)
Figure 55: Close up of an animated wheat leaf when limiting nutrient is calcium (original in colour)

Figure 56: Animation of colour changes over a series of days when manganese (Mn) is the limiting nutrient during spring; also represents molybdenum (Mo) colour change (original in colour)
Figure 57: Colour changes in an animated wheat leaf when limiting nutrient is manganese (original in colour)

Figure 58: Colour changes in an actual wheat leaf when limiting nutrient is manganese (adapted from (Snowball & Robson, 1991)) (original in colour)
Figure 59: Animation of colour changes over a series of days when potassium (K) is the limiting nutrient during spring; also represents iron (Fe) colour change (original in colour)

Figure 60: Colour changes in an animated wheat leaf when limiting nutrient is potassium (K) (original in colour)
Figure 61: Colour changes in an actual wheat leaf control on left and limiting nutrient potassium 3 samples (adapted from Snowball & Robson, 1991) (original in colour)

Figure 62: Mottling in animated wheat leaf when limiting nutrient is zinc (Zn) (original in colour)
Figure 63: Mottling in an actual wheat leaf when limiting nutrient is zinc (Zn) (adapted from (Snowball & Robson, 1991)) (original in colour)

Figure 64: Height comparison of animated wheat; control on left vs limiting nutrient nitrogen (N) on right (original in colour)

Figure 65: Height comparison of actual wheat plant control on left vs limiting nitrogen (N) on right (adapted from (Snowball & Robson, 1991)) (original in colour)
Figure 66: Height comparison of animated wheat with control on left vs limiting nutrient potassium (K) on right (original in colour)

Figure 67: Height comparison of actual wheat; control on left vs limiting potassium (K) on right (adapted from (Cooper & Regents, 2009)) (original in colour)
Figure 68: Height comparison of animated wheat; control on left vs limiting nutrient phosphorous (P) on right (original in colour)

Figure 69: Height comparison of actual wheat; control on left vs limiting phosphorous (P) on right (adapted from (Snowball & Robson, 1991)) (original in colour)

Figure 70: Height comparison of animated wheat; control on left vs limiting nutrient magnesium (Mg) on right (original in colour)

Figure 71: Height comparison of actual wheat; control on left vs limiting magnesium (Mg) on right (adapted from (Yara, 2017)) (original in colour)
Figure 72: Height comparison of animated wheat; control on left vs limiting nutrient sulfur (S) on right (original in colour)

Figure 73: Height comparison of actual wheat; control on left vs limiting sulfur (S) on right (adapted from (Snowball & Robson, 1991)) (original in colour)

Figure 74: Height comparison of animated wheat; control on left vs limiting nutrient iron (Fe) on right (original in colour)
Figure 75: Height comparison of actual wheat; control on left vs limiting iron (Fe) on right (adapted from (Brennan and Scanlan, 2015)) (original in colour)

Figure 76: Height comparison of animated wheat; control on left vs limiting nutrient copper (Cu) on right (original in colour)

Figure 77: Height comparison of actual wheat; various deficiencies in copper (Cu); from control on left to highest Cu deficiency on right (adapted from (Evans, 2001)) (original in colour)
Figure 78: Height comparison of animated wheat with control on left vs limiting nutrient boron (B) on right (original in colour)

Figure A-1: Plant organization, part 1
Figure A-2: Plant organization, part 2

Figure A-3: Flower structure
Figure A-4: Generic leaf taken with Finepix digital camera
Figure A-5: Leaf shape morphology (adapted from (Swink & Wilhelm, 1994) using InkArt)
Figure A-6: Simple and compound leaves (adapted from (Swink & Wilhelm, 1994) using InkArt).
Figure A-7 Leaf types taken with FinePix camera
Figure A-8: Leaf phyllotaxis (Swink & Wilhelm, 1994) using InkArt).
Figure A-9: Orientation of leaves using L-systems

Figure A-10: Stem shapes created using Blender
Figure A-11: Conifer trees taken with FinePix digital camera
Figure A-12: Conifer trees created with Blender

Figure A-13: Stem branching patterns created with Blender.
Figure A-14: Herbaceous plant from HP 2175 all-in-one printer (scanned)

Figure A-15: Stem apical meristem (adapted from (Swink & Wilhelm, 1994) using InkArt)
Appendix C: Tables

Table 1: The values of constants used to generate branching structures (Prusinkiewicz, Hammel, Hanan, & Mĕch, 1996a; Prusinkiewicz, Karwowski, Perttunen, & Sievänen, 1996b)

<table>
<thead>
<tr>
<th>Figure</th>
<th>$r_1$</th>
<th>$r_2$</th>
<th>$\alpha_1$</th>
<th>$\alpha_2$</th>
<th>$\phi_1$</th>
<th>$\phi_2$</th>
<th>$\psi_0$</th>
<th>$q$</th>
<th>$e$</th>
<th>$m$</th>
<th>$n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>.75</td>
<td>.77</td>
<td>35</td>
<td>-35</td>
<td>0</td>
<td>0</td>
<td>30</td>
<td>.50</td>
<td>.40</td>
<td>0.0</td>
<td>10</td>
</tr>
<tr>
<td>b</td>
<td>.65</td>
<td>.71</td>
<td>27</td>
<td>-68</td>
<td>0</td>
<td>0</td>
<td>20</td>
<td>.50</td>
<td>.50</td>
<td>1.7</td>
<td>12</td>
</tr>
<tr>
<td>c</td>
<td>.50</td>
<td>.85</td>
<td>25</td>
<td>-15</td>
<td>180</td>
<td>0</td>
<td>20</td>
<td>.45</td>
<td>.50</td>
<td>0.5</td>
<td>9</td>
</tr>
<tr>
<td>d</td>
<td>.60</td>
<td>.85</td>
<td>25</td>
<td>-15</td>
<td>180</td>
<td>180</td>
<td>20</td>
<td>.45</td>
<td>.50</td>
<td>0.0</td>
<td>10</td>
</tr>
<tr>
<td>e</td>
<td>.58</td>
<td>.83</td>
<td>30</td>
<td>15</td>
<td>0</td>
<td>180</td>
<td>20</td>
<td>.40</td>
<td>.50</td>
<td>1.0</td>
<td>11</td>
</tr>
<tr>
<td>f</td>
<td>.92</td>
<td>.37</td>
<td>0</td>
<td>60</td>
<td>180</td>
<td>0</td>
<td>2</td>
<td>.50</td>
<td>.00</td>
<td>0.5</td>
<td>15</td>
</tr>
<tr>
<td>g</td>
<td>.80</td>
<td>.80</td>
<td>30</td>
<td>-30</td>
<td>137</td>
<td>137</td>
<td>30</td>
<td>.50</td>
<td>.50</td>
<td>0.0</td>
<td>10</td>
</tr>
<tr>
<td>h</td>
<td>.95</td>
<td>.75</td>
<td>5</td>
<td>-30</td>
<td>-90</td>
<td>90</td>
<td>40</td>
<td>.60</td>
<td>.45</td>
<td>25.0</td>
<td>12</td>
</tr>
<tr>
<td>i</td>
<td>.55</td>
<td>.95</td>
<td>-5</td>
<td>30</td>
<td>137</td>
<td>137</td>
<td>5</td>
<td>.40</td>
<td>.00</td>
<td>5.0</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 2: Examples of objects selected for three scenarios

<table>
<thead>
<tr>
<th>Scenario</th>
<th>VIRTUAL WORLD objects</th>
<th>FORM objects</th>
<th>FUNCTION objects</th>
<th>ENVIRONMENT objects</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Gaming</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>New 3D game creation</td>
<td>World Object, Position, Polygon, Transformation, Light, Material, Topography</td>
<td>Stem, Leaf, Flower</td>
<td>Locomotion</td>
<td>Radiation Regimen, Wind</td>
</tr>
<tr>
<td><strong>Biological Simulation</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Film</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Animated film with plant</td>
<td>World Object, Position, Polygon, Transformation, Light, Material, Topography</td>
<td>Stem, Leaf, Flower</td>
<td>Locomotion</td>
<td>Radiation Regimen, Wind</td>
</tr>
</tbody>
</table>
Table 3: Effect of ENVIRONMENT on FUNCTION and FORM

<table>
<thead>
<tr>
<th>FORM Object or property</th>
<th>FUNCTION Object</th>
<th>ENVIRONMENT Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stem</td>
<td>Motion</td>
<td>Wind</td>
</tr>
<tr>
<td>stem elongation rate</td>
<td>Growth</td>
<td>Nutrient Regimen and Seasonal</td>
</tr>
<tr>
<td>stem colour</td>
<td>Response to nutrient</td>
<td>Nutrient Regimen and Seasonal</td>
</tr>
<tr>
<td>Leaf</td>
<td>Motion</td>
<td>Wind</td>
</tr>
<tr>
<td>leaf elongation rate</td>
<td>Growth</td>
<td>Nutrient Regimen and Seasonal</td>
</tr>
<tr>
<td>leaf colour</td>
<td>Response to nutrient</td>
<td>Nutrient Regimen and Seasonal</td>
</tr>
<tr>
<td>Fruit</td>
<td>Motion</td>
<td>Wind</td>
</tr>
<tr>
<td>fruit size</td>
<td>Growth</td>
<td>Nutrient Regimen and Seasonal</td>
</tr>
</tbody>
</table>

Table 4 Radius variation leaf shape generation

<table>
<thead>
<tr>
<th>Leaf type</th>
<th>Real example</th>
<th>Leaf method</th>
<th>Variation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acicular (Needle), Filiform, Subulate</td>
<td>(1)</td>
<td></td>
<td>(1)</td>
</tr>
<tr>
<td>Shape</td>
<td>Image</td>
<td>Diagram 1</td>
<td>Diagram 2</td>
</tr>
<tr>
<td>--------</td>
<td>-------</td>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td>Deltoid</td>
<td><img src="image1.png" alt="Deltoid Image" /></td>
<td><img src="image2.png" alt="Diagram 1" /></td>
<td><img src="image3.png" alt="Diagram 2" /></td>
</tr>
<tr>
<td>Elliptic</td>
<td><img src="image5.png" alt="Elliptic Image" /></td>
<td><img src="image6.png" alt="Diagram 1" /></td>
<td><img src="image7.png" alt="Diagram 2" /></td>
</tr>
<tr>
<td>Type</td>
<td>Description</td>
<td>Image</td>
<td></td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>Falcate</td>
<td>(petiole at the wide end)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flabelate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hastate</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lanceolate</td>
<td>(petiole at the wide end)/oblanceolate (petiole at the narrow end)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linear</td>
<td><img src="image" alt="Linear" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>----------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oblong</td>
<td><img src="image" alt="Oblong" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Obtuse, Ovate (petiole at the wide end), Obovate (petiole at the narrow end)</td>
<td><img src="image" alt="Obtuse, Ovate, Obovate" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Orbicular, Round</td>
<td><img src="image" alt="Orbicular, Round" /></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1)
Palmate and lobed
<table>
<thead>
<tr>
<th>Shape</th>
<th>Description</th>
<th>Image 1</th>
<th>Image 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sagittate</td>
<td>(spear-shape)</td>
<td><img src="image1" alt="Image" /></td>
<td><img src="image2" alt="Image" /></td>
</tr>
<tr>
<td>Scale</td>
<td></td>
<td><img src="image3" alt="Image" /></td>
<td></td>
</tr>
<tr>
<td>Spatulate</td>
<td>(spoon-shape)</td>
<td><img src="image4" alt="Image" /></td>
<td></td>
</tr>
<tr>
<td>Trifoliate/Ternate</td>
<td></td>
<td><img src="image5" alt="Image" /></td>
<td><img src="image6" alt="Image" /></td>
</tr>
</tbody>
</table>
Table 5: Growth rates for plant species

<table>
<thead>
<tr>
<th>Species</th>
<th>Branch growth rate</th>
<th>Change in growth rate due to wind</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aspen (Populus tremula)</td>
<td>25 or more inches per year (used Populus tremuloides)</td>
<td>50% reduction in leaf area</td>
<td>(Flückiger, Oertli, &amp; Flückiger-Keller, 1978)</td>
</tr>
<tr>
<td>Bamboo (Poaceae)</td>
<td>Some grow 39 inches/day</td>
<td>No study found</td>
<td>(Farrelly, 1984)</td>
</tr>
<tr>
<td>Barley</td>
<td>18% increase per day</td>
<td>No change in growth rate</td>
<td>(Wadsworth, 1959), (Wadsworth, 1960)</td>
</tr>
<tr>
<td>Birch (Betula pendula)</td>
<td>13 or more inches a year</td>
<td>26%</td>
<td>(Dirr, 1998)</td>
</tr>
<tr>
<td>Species</td>
<td>Growth Rate</td>
<td>Effect on Leaf Area</td>
<td>Author(s)</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------</td>
<td>------------------------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>Dogwood (Cornus sanguinea)</td>
<td>45 cm per year</td>
<td>20% reduction in leaf area</td>
<td>Royal Horticultural Society, 2011</td>
</tr>
<tr>
<td>Honeysuckle (Lonicera xylosteum)</td>
<td>1.5 m per year</td>
<td>33.3% reduction in leaf area</td>
<td>Flückiger, Oertli, &amp; Flückiger-Keller, 1978 (wind effects)</td>
</tr>
<tr>
<td>Lodge Pole Pine (Pinus contorta v. latifolia)</td>
<td>24 or fewer inches per year</td>
<td>12 to 40 inches per year</td>
<td>Dirr, 1998 (growth rate)</td>
</tr>
<tr>
<td>Kidney beans (Phaseolus vulgaris L. cv. Red Cherokee Wax)</td>
<td>6.4 mm/day ave. increase in elongation per day</td>
<td>1.4% decrease in elongation</td>
<td>Hunt &amp; Jaffe, 1980</td>
</tr>
<tr>
<td>Plant</td>
<td>Increase in leaf diameter</td>
<td>Decrease in leaf diameter</td>
<td>References</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------------------</td>
<td>---------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Oak (Quercus robur)</td>
<td>24 or fewer inches per year (used White Oak)</td>
<td>25% reduction in leaf area</td>
<td>(Dirr, 1998) (growth rate) (Flückiger, et al., 1978) (wind effects)</td>
</tr>
<tr>
<td>Pea</td>
<td>11% increase per day</td>
<td>No change in growth rate</td>
<td>(Wadsworth, 1959), (Wadsworth, 1960)</td>
</tr>
<tr>
<td>Rape (Brassica napus)</td>
<td>25% increase per day</td>
<td>No change in growth rate</td>
<td>(Wadsworth, 1959), (Wadsworth, 1960)</td>
</tr>
<tr>
<td>White Ash (Fraxinus Americana)</td>
<td>13 to 24 inches per year</td>
<td>40% decrease in leaf area</td>
<td>(Dirr, 1998) (growth rate) (Flückiger, et al., 1978) (wind effects)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Nutrient:</th>
<th>Nitrogen (N)</th>
<th>Rate of reduction ($d_n$) per growth day</th>
<th>Standard deviation of wheat height</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concentrations (mg/kg soil)</td>
<td>0</td>
<td>40</td>
<td>80</td>
</tr>
<tr>
<td>Height of Wheat (cm)</td>
<td>45</td>
<td>57</td>
<td>62</td>
</tr>
<tr>
<td>Nutrient:</td>
<td>Phosphate (P): Potassium (K)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Concentrations (kg/ha)</td>
<td>0:0</td>
<td>30:30</td>
<td>60:60</td>
</tr>
<tr>
<td>Height of Wheat (cm)</td>
<td>41.38</td>
<td>55.08</td>
<td>65.23</td>
</tr>
<tr>
<td>Nutrient:</td>
<td>Potassium (K): Phosphate (P)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Concentrations (kg/ha)</td>
<td>0:0</td>
<td>30:30</td>
<td>60:30</td>
</tr>
<tr>
<td>Height of Wheat (cm)</td>
<td>41.38</td>
<td>55.08</td>
<td>66.98</td>
</tr>
<tr>
<td>Nutrient:</td>
<td>Calcium (Ca)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Concentrations (kg/ha)</td>
<td>0</td>
<td>7.5</td>
<td>15</td>
</tr>
<tr>
<td>Height of Wheat (cm)</td>
<td>75.29</td>
<td>75.17</td>
<td>74.54</td>
</tr>
<tr>
<td>Nutrient:</td>
<td>Magnesium (Mg)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Concentrations (kg/ha)</td>
<td>0</td>
<td>60</td>
<td>120</td>
</tr>
<tr>
<td>Height of Wheat (cm)</td>
<td>27</td>
<td>36</td>
<td>34</td>
</tr>
<tr>
<td>Nutrient:</td>
<td>Sulfur (S)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Concentrations (kg/ha)</td>
<td>0</td>
<td>25</td>
<td>50</td>
</tr>
<tr>
<td>Height of Wheat (cm)</td>
<td>86.6</td>
<td>105.0</td>
<td>97.7</td>
</tr>
<tr>
<td>Nutrient:</td>
<td>Zinc (Zn)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Concentrations (kg/ha)</td>
<td>0</td>
<td>60</td>
<td>120</td>
</tr>
<tr>
<td>Nutrient:</td>
<td>Copper (Cu)(^7)</td>
<td>Nutrient:</td>
<td>Iron (Fe)(^8)</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------</td>
<td>-----------</td>
<td>-----------------</td>
</tr>
<tr>
<td>Concentrations (mg/kg soil)</td>
<td>0.03515</td>
<td>Concentrations (kg/ha)</td>
<td>0.00143</td>
</tr>
<tr>
<td>Height of Wheat (cm)</td>
<td>37.4</td>
<td>Height of Wheat (% change in cm)</td>
<td>17.59</td>
</tr>
<tr>
<td>(cm ha/ kg)</td>
<td>42.9</td>
<td>(cm mg/ kg)</td>
<td>1.3628</td>
</tr>
</tbody>
</table>

Table 7: Determining colour changes

<table>
<thead>
<tr>
<th>Leaf emergence age ((a_i, A))</th>
<th>Order of effect by leaf emergence age ((o_n))</th>
<th>Pattern of colour change ((p_n))</th>
<th>Resulting colour change ((h_n))</th>
<th>Mottling ((m_n))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Old (0)</td>
<td>Old to middle to young (0)</td>
<td>Margin to center (0)</td>
<td>Green to yellow to white (0)</td>
<td>None (0)</td>
</tr>
<tr>
<td>Middle (1)</td>
<td>Middle to old to young (1)</td>
<td>Center to margin (1)</td>
<td>Green to yellow (1)</td>
<td>Mottled (1)</td>
</tr>
<tr>
<td>Young (2)</td>
<td>Young to middle to old (2)</td>
<td>Tip to base (2)</td>
<td>Green to reddish purple (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Base to tip (3)</td>
<td>Green to yellow to brown (3)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Intervereval: vein to mid-vein (4)</td>
<td>Green to grey-green (4)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Intervereval: mid-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 8: Colour change from limiting nutrient

<table>
<thead>
<tr>
<th>Limiting nutrient ($n$)</th>
<th>Order of effect by leaf emergence age ($o_n$)</th>
<th>Pattern of colour change ($p_n$)</th>
<th>Resulting colour change ($h_n$)</th>
<th>Mottling ($m_n$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nitrogen</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Potassium</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Phosphorous</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>Calcium</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Magnesium</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Sulfur</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Molybdenum</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Manganese</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Zinc</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Copper</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Iron</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Boron</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 9: Colour changes in wheat leaves (Snowball & Robson, 1991)

<table>
<thead>
<tr>
<th>Nutrient</th>
<th>Effect on wheat plant colour</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nitrogen (N)</td>
<td>Oldest leaves affected first, yellowing in a tip to base pattern.</td>
</tr>
<tr>
<td>Phosphorus (P)</td>
<td>Oldest leaves affected first, yellowing in a tip to base pattern.</td>
</tr>
<tr>
<td>Potassium (K)</td>
<td>Oldest leaves affected first, yellowing in an interveinal vein edge to vein middle pattern and will have mottling.</td>
</tr>
<tr>
<td>Calcium (Ca)</td>
<td>Youngest leaves affected first, yellowing in a middle to edge pattern.</td>
</tr>
<tr>
<td>Magnesium (Mg)</td>
<td>Youngest leaves are affected first, yellowing in a mottled pattern.</td>
</tr>
<tr>
<td>Sulphur (S)</td>
<td>Youngest leaves affected first, yellowing in a tip to base pattern.</td>
</tr>
<tr>
<td>Molybdenum (Mo)</td>
<td>Middle leaves affected first, yellowing in an interveinal vein middle to vein edge pattern.</td>
</tr>
<tr>
<td>Manganese (Mn)</td>
<td>Youngest leaves affected first, yellowing in an interveinal pattern and has mottling.</td>
</tr>
<tr>
<td>Zinc (Zn)</td>
<td>Middle leaves affected first, yellowing in middle to edge pattern.</td>
</tr>
<tr>
<td>Copper (Cu)</td>
<td>Youngest leaves affected first, grey-green in a tip to base pattern with mottling.</td>
</tr>
<tr>
<td>Iron (Fe)</td>
<td>Youngest leaves affected first, yellowing in an interveinal pattern.</td>
</tr>
<tr>
<td>Boron (B)</td>
<td>Youngest leaves affected first, yellowing in a base to tip pattern.</td>
</tr>
</tbody>
</table>
Table 10: Number of days for wheat growth (adapted from (Stapper & Fischer, 1990; Calendar, 2013))

<table>
<thead>
<tr>
<th>Development Stage</th>
<th>Number of development days</th>
<th>Spring Wheat (Canada)</th>
<th>Winter Wheat (US)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emergence and tiller</td>
<td>60</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Vegetative (elongation)</td>
<td>30</td>
<td>60\textsuperscript{1}</td>
<td></td>
</tr>
<tr>
<td>Heading</td>
<td>30</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Filling</td>
<td>30</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Maturity\textsuperscript{1}</td>
<td>30</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>180</td>
<td>190</td>
<td></td>
</tr>
</tbody>
</table>

Table 11: Wheat nutritional requirements (adapted from McKenzie, 1998)

<table>
<thead>
<tr>
<th>Nutrient (n)</th>
<th>$c_{n,\text{min}}$</th>
<th>$c_{n,\text{opt}}$</th>
<th>$c_{n,\text{tox}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nitrogen (N)</td>
<td>1.25%</td>
<td>1.75%</td>
<td>4.0%</td>
</tr>
<tr>
<td>Potassium (K)</td>
<td>1.0%</td>
<td>1.5%</td>
<td>5.0%</td>
</tr>
<tr>
<td>Phosphorous (P)</td>
<td>0.15%</td>
<td>0.26%</td>
<td>0.8%</td>
</tr>
<tr>
<td>Calcium (Ca)</td>
<td>0.1%</td>
<td>0.2%</td>
<td>1.5%</td>
</tr>
<tr>
<td>Magnesium (Mg)</td>
<td>0.1%</td>
<td>0.15%</td>
<td>1.0%</td>
</tr>
<tr>
<td>Sulphur (S)</td>
<td>0.1%</td>
<td>0.020%</td>
<td>0.8%</td>
</tr>
<tr>
<td>Molybdenum (Mo)</td>
<td>0.01ppm</td>
<td>0.03 ppm</td>
<td>10 ppm</td>
</tr>
<tr>
<td>Manganese (Mn)</td>
<td>10 ppm</td>
<td>15 ppm</td>
<td>250 ppm</td>
</tr>
<tr>
<td>Zinc (Zn)</td>
<td>10 ppm</td>
<td>15 ppm</td>
<td>150 ppm</td>
</tr>
<tr>
<td>Copper (Cu)</td>
<td>3.0 ppm</td>
<td>4.5 ppm</td>
<td>50 ppm</td>
</tr>
<tr>
<td>Iron (Fe)</td>
<td>15 ppm</td>
<td>20 ppm</td>
<td>500 ppm</td>
</tr>
<tr>
<td>Boron (B)</td>
<td>3 ppm</td>
<td>5 ppm</td>
<td>75 ppm</td>
</tr>
</tbody>
</table>
Nutrient influences the physiology of the plant, as explained in (Uchida, 2000; Tucker, 1999; Murdock & Call, 1999; McCauley, Jones, & Jacobsen, 2009).

<table>
<thead>
<tr>
<th>Nutrient</th>
<th>Role in plant</th>
<th>Required for growth</th>
<th>Required for proper colour</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nitrogen (N)</td>
<td>Required for proteins, chlorophyll, and protoplasm. Influences cell size, photosynthesis, promotes growth, fruit, seed, and maturity.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Potassium (K)</td>
<td>Required to activate enzymes and co-enzymes, photosynthesis, sugar transport and protein generation.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Phosphorus (P)</td>
<td>Required for the development of energy, sugars and nucleic acids.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Calcium (Ca)</td>
<td>Required for plant cell wall.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Magnesium (Mg)</td>
<td>Required for producing energy and for photosynthesis (part of chlorophyll)</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sulphur (S)</td>
<td>Required in protein and chlorophyll formation.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Molybdenum (Mo)</td>
<td>Required for enzyme activity and nitrogen fixation.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Manganese (Mn)</td>
<td>Required within chloroplasts.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Zinc (Zn)</td>
<td>Required for growth hormone and internode elongation.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Copper (Cu)</td>
<td>Required for respiration, protein synthesis, and chlorophyll production.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Iron (Fe)</td>
<td>Required for photosynthesis and respiration.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Boron (B)</td>
<td>Required for cell wall formation and reproduction.</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Chlorine (Cl)</td>
<td>Required for leaf turgor and photosynthesis</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Nickel (Ni)</td>
<td>Required for seed germination</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
Table 13: Nutrient ratios for major types of plants\(^1\) (Knecht & Goransson, 2004),

<table>
<thead>
<tr>
<th>Plant Type</th>
<th>N</th>
<th>P</th>
<th>K</th>
<th>Ca</th>
<th>Mg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deciduous(^1)</td>
<td>100</td>
<td>14.6</td>
<td>64.6</td>
<td>7.0</td>
<td>9.4</td>
</tr>
<tr>
<td>Coniferous(^1)</td>
<td>100</td>
<td>15.0</td>
<td>47.5</td>
<td>8.0</td>
<td>7.5</td>
</tr>
<tr>
<td>Herbaceous(^1)</td>
<td>100</td>
<td>14.3</td>
<td>68.3</td>
<td>8.3</td>
<td>8.7</td>
</tr>
</tbody>
</table>

Table 14: Nutrient ratios for wheat

<table>
<thead>
<tr>
<th>Nutrient (optimal percent concentration)</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
</tr>
<tr>
<td>----</td>
</tr>
<tr>
<td>Wheat</td>
</tr>
</tbody>
</table>

Table 15: General effects on colour due to plant nutrient deficiency (Römheld, 2012\(^1\); Uchida, 2000\(^2\))

<table>
<thead>
<tr>
<th>Nutrient</th>
<th>Leaf age</th>
<th>Effect(^1)</th>
<th>Colour(^2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calcium</td>
<td>Young</td>
<td>Chlorosis tip to base, necrosis if deficiency continues.</td>
<td>Pale yellow</td>
</tr>
<tr>
<td>Boron</td>
<td>Young</td>
<td>Chlorosis base to tip, necrosis if deficiency continues.</td>
<td>Light green</td>
</tr>
<tr>
<td>Copper</td>
<td>Young</td>
<td>Wilt with chlorosis then necrosis at tip</td>
<td>yellow</td>
</tr>
<tr>
<td>Zinc</td>
<td>Young</td>
<td>Necrotic spots, interveinal chlorosis</td>
<td>Yellow to white</td>
</tr>
<tr>
<td>Manganese</td>
<td>Young</td>
<td>Necrotic spots, interveinal chlorosis then necrosis</td>
<td>Yellow to white</td>
</tr>
<tr>
<td>Iron</td>
<td>Young</td>
<td>Chlorosis interveinal</td>
<td>Yellow to white</td>
</tr>
<tr>
<td>Sulfur</td>
<td>Young</td>
<td>Uniform Chlorosis</td>
<td>yellow</td>
</tr>
<tr>
<td>Phosphorous</td>
<td>Old</td>
<td>Chlorosis over all plant</td>
<td>Red/purple underside</td>
</tr>
<tr>
<td>Nitrogen</td>
<td>Old</td>
<td>Uniform Chlorosis then necrosis all over the plant</td>
<td>Pale yellow</td>
</tr>
<tr>
<td>Magnesium</td>
<td>Old</td>
<td>Mottling with interveinal chlorosis then necrosis</td>
<td>Yellow then red</td>
</tr>
<tr>
<td>Potassium</td>
<td>Old</td>
<td>Mottling with grey-green leaf, Brown spots,</td>
<td></td>
</tr>
</tbody>
</table>
Appendix D: Stage 2 Selection Detail

This appendix is provided to minimize the duplication of object selection amongst the coding examples. This appendix describes in detail the common objects that were selected for every coding example in this thesis.

Stage 2 VIRTUAL WORLD Domain Selection

The VIRTUAL WORLD domain selections are: World Object (Position), WorldInf (Background), 2D conversion - Polygon, Transform (Translation, Rotation, Scale), Light (Light position, Light Aim Direction), Material (Colour, Normal), Viewpoint (LookAt), Topography, and Time objects.

Stage 2 FORM Domain Selection

The FORM domain list selections (includes one of the hierarchy the selection came from and the target object or property) are:


<table>
<thead>
<tr>
<th>Molybdenum²</th>
<th>Old</th>
<th>Uniform chlorosis, necrotic spots</th>
<th>Yellow, brown spots</th>
</tr>
</thead>
<tbody>
<tr>
<td>interveinal and necrosis near tips and margins</td>
<td>yellow interveinal</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
• Plant trait (TO:0000387) – anatomy and morphology trait 
  (TO:0000017) – Plant structure anatomy and morphology trait 
  (TO:0000839) – multi-tissue plant structure anatomy and 
  morphology trait (TO:0000836) – plant organ anatomy and 
  morphology trait (TO:0000736) – plant axis anatomy and 
  morphology trait (TO:0000739) – stem anatomy and morphology 
  trait (TO:0000361) - stem elongation property (TO:0006036),

• Plant trait (TO:0000387) – anatomy and morphology trait 
  (TO:0000017) – Plant structure anatomy and morphology trait 
  (TO:0000839) – multi-tissue plant structure anatomy and 
  morphology trait (TO:0000836) – plant organ anatomy and 
  morphology trait (TO:0000736) – plant axis anatomy and 
  morphology trait (TO:0000739) – stem anatomy and morphology 
  trait (TO:0000361) - stem length property (TO:0000576),

• Plant trait (TO:0000387) – anatomy and morphology trait 
  (TO:0000017) – Plant structure anatomy and morphology trait 
  (TO:0000839) – multi-tissue plant structure anatomy and 
  morphology trait (TO:0000836) – plant organ anatomy and 
  morphology trait (TO:0000736) – plant axis anatomy and 
  morphology trait (TO:0000739) – stem anatomy and morphology 
  trait (TO:0000361) - stem diameter property (TO:0020083),

• Plant trait (TO:0000387) – anatomy and morphology trait 
  (TO:0000017) – Plant structure anatomy and morphology trait
(TO:0000839) – multi-tissue plant structure anatomy and
morphology trait (TO:0000836) – plant organ anatomy and
morphology trait (TO:0000736) – plant axis anatomy and
morphology trait (TO:0000739) – stem anatomy and morphology
trait (TO:0000361) -stem colour property (TO:0000056),

o Plant trait (TO:0000387) – anatomy and morphology trait
(TO:0000017) – Plant structure anatomy and morphology trait
(TO:0000839) – multi-tissue plant structure anatomy and
morphology trait (TO:0000836) – plant organ anatomy and
morphology trait (TO:0000736) – plant axis anatomy and
morphology trait (TO:0000739) – stem anatomy and morphology
trait (TO:0000361) -Stem Angle object (TO:0000577),

o Plant trait (TO:0000387) – anatomy and morphology trait
(TO:0000017) – Plant structure anatomy and morphology trait
(TO:0000839) – multi-tissue plant structure anatomy and
morphology trait (TO:0000836) – plant organ anatomy and
morphology trait (TO:0000736) – plant axis anatomy and
morphology trait (TO:0000739) – stem anatomy and morphology
trait (TO:0000361) – stem strength object (TO:000051)

• Plant anatomical entity (PO:0025131) – plant structure (PO:0009011) –
multi-tissue plant structure (PO:0025496) – plant organ (PO:0009008) –
collective plant organ structure (PO:0025007) – shoot system
(PO:0009006) – shoot axis (PO:0025029) – shoot apex (PO:0000037) –
phyllome primordium (PO:0025128) – vascular leaf primordium (PO:00000017) – vascular leaf (PO:0009025) – Petiole object (PO:0020038),

- Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – cardinal part of a multi-tissue plant structure anatomy and morphology trait (TO:0000837) – cardinal organ part anatomy and morphology trait (TO:0000754) – stalk anatomy and morphology trait (TO:0000764) – petiole anatomy and morphology trait (TO:0000765) – petiole length property (TO:0000766)

- Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – cardinal part of a multi-tissue plant structure anatomy and morphology trait (TO:0000837) – cardinal organ part anatomy and morphology trait (TO:0000754) – stalk anatomy and morphology trait (TO:0000764) – petiole anatomy and morphology trait (TO:0000765) – petiole shape property (TO:0000767)

- Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – cardinal part of a multi-tissue plant structure anatomy and morphology trait (TO:0000837) – cardinal organ part anatomy and morphology trait (TO:0000754) – stalk anatomy and morphology trait (TO:0000764) – petiole anatomy and morphology trait (TO:0000765) – stalk anatomy and
morphology trait (TO:0000764) – petiole anatomy and morphology trait (TO:0000765) – petiole size property (TO:0000768)

- Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – cardinal part of a multi-tissue plant structure anatomy and morphology trait (TO:0000837) – cardinal organ part anatomy and morphology trait (TO:0000754) – stalk anatomy and morphology trait (TO:0000764) – petiole anatomy and morphology trait (TO:0000765) – petiole colour property (TO:0002707)

- Plant anatomical entity (PO:0025131) – plant structure (PO:0009011) – multi-tissue plant structure (PO:0025496) – plant organ (PO:0009008) – collective plant organ structure (PO:0025007) – shoot system (PO:0009006) – phyllome (PO:0006001) - Leaf object (PO: 0025034),

  - Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – multi-tissue plant structure anatomy and morphology trait (TO:0000836) – plant organ anatomy and morphology trait (TO:0000736) – phyllome anatomy and morphology trait (TO:0000747) – leaf anatomy and morphology trait (TO:0000748) – leaf lamina anatomy and morphology trait (TO:0000829) – leaf length property (TO:0000135),

  - Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait
(TO:0000839) – multi-tissue plant structure anatomy and morphology trait (TO:0000836) – plant organ anatomy and morphology trait (TO:0000736) – phyllome anatomy and morphology trait (TO:0000747) – leaf anatomy and morphology trait (TO:0000748) – leaf lamina anatomy and morphology trait (TO:0000829) – leaf width property (TO:0000370),

- Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – multi-tissue plant structure anatomy and morphology trait (TO:0000836) – plant organ anatomy and morphology trait (TO:0000736) – phyllome anatomy and morphology trait (TO:0000747) – leaf anatomy and morphology trait (TO:0000748) - leaf thickness property (TO:0000258),

- Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – multi-tissue plant structure anatomy and morphology trait (TO:0000836) – plant organ anatomy and morphology trait (TO:0000736) – phyllome anatomy and morphology trait (TO:0000747) – leaf anatomy and morphology trait (TO:0000748) – leaf attitude (TO:0000824) - Leaf Angle object (TO:0000206),

- Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait
(TO:0000839) – multi-tissue plant structure anatomy and morphology trait (TO:0000836) – plant organ anatomy and morphology trait (TO:0000736) – phyllome anatomy and morphology trait (TO:0000747) – leaf anatomy and morphology trait (TO:0000748) leaf lamina anatomy and morphology trait (TO:0000829) – Leaf Lamina Joint Bending object (TO:0002688)

- Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – multi-tissue plant structure anatomy and morphology trait (TO:0000836) – plant organ anatomy and morphology trait (TO:0000736) – phyllome anatomy and morphology trait (TO:0000747) – leaf anatomy and morphology trait (TO:0000748) – leaf lamina anatomy and morphology trait (TO:0000829) - leaf colour property (TO:0000326),

- Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – multi-tissue plant structure anatomy and morphology trait (TO:0000836) – plant organ anatomy and morphology trait (TO:0000736) – phyllome anatomy and morphology trait (TO:0000747) – leaf anatomy and morphology trait (TO:0000748) - leaf shape property (TO:0000492).
• Plant anatomical entity (PO:0025131) – plant structure (PO:0009011) – whole plant (PO:0000003) – plant embryo (PO:0009009) – Seed object (PO: 0009010),
  o Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – multi-tissue plant structure anatomy and morphology trait (TO:0000836) – seed anatomy and morphology trait (TO:0000184) – seed weight (TO:0000181) – seed quality (TO:0000162) – seed size property (TO:0000391) – seed length property (TO:0000146)
  o Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – multi-tissue plant structure anatomy and morphology trait (TO:0000836) – seed anatomy and morphology trait (TO:0000184) – seed weight (TO:0000181) – seed quality (TO:0000162) – seed size property (TO:0000391) – seed width property (TO:0000149),
  o Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – multi-tissue plant structure anatomy and morphology trait (TO:0000836) – seed anatomy and morphology trait (TO:0000184) – seed weight (TO:0000181) – seed quality
(TO:0000162) – seed size property (TO:0000391) – seed thickness property (TO:0000304)

- Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – multi-tissue plant structure anatomy and morphology trait (TO:0000836) – seed anatomy and morphology trait (TO:0000184) – seed quality (TO:0000162) – seed colour property (TO:0000486),

- Plant trait (TO:0000387) – anatomy and morphology trait (TO:0000017) – Plant structure anatomy and morphology trait (TO:0000839) – multi-tissue plant structure anatomy and morphology trait (TO:0000836) – seed anatomy and morphology trait (TO:0000184) – seed shape property (TO:0000484).

Stage 2 FUNCTION Domain Selection

The FUNCTION domain selections (includes one of the hierarchy the selection came from and the target object or property) are:


- Occurrent (BFO:0000003) – process (BFO:0000015) – Biological_process (GO:0008150) – Growth (GO:0040007) –
Developmental growth (GO:0048589)-meristem growth (GO:0080112) –
vegetative meristem growth (GO:0010448)

- *Plant structure development stage* (PO:0009012) – *plant organ
development stage* (PO:0025339) – *phyllome development stage*
(PO:0025579) – *Leaf development stage* (PO:0001050)

- *Plant structure development stage* (PO:0009012) – collective plant
organ structure development stage (PO:0025338) – *Shoot system
development stage* (PO:0025527),

- *Occurrent* (BFO:0000003)- *Process* (BFO:0000015) –
*Biological_process* (GO:0008150) – *Locomotion object*(GO:0008150),

- *Occurrent* (BFO:0000003)- *Process* (BFO:0000015) –
*Biological_process* (GO:0008150) – *Response to Stimulus*
(GO:00050896) – *Response to chemical* (GO:0042221) – *Response to
Nutrient object*(GO:0007584),

- *Occurrent* (BFO:0000003) – *Process* (BFO:0000015) –
*Biological_process* (GO:0008150) – *Response to stimulus*
(GO:00050896) – *Response to chemical* (GO:0042221) – *Response to
Nitrogen compound object*(GO:1901698)

Stage 2 ENVIRONMENT Domain Selection
The Environment domain selections (including one of the hierarchy the selection came from and the target object or property) from the plant environment ontology (EO:0007359) are:

- **Abiotic (EO:0007191)** - Physical (EO:0007316) or ecological environment (EO:0007064) - *Wind environment* object (EO:0007382) (note, we needed to add wind direction and velocity to our demonstration, only air temperature was in the EO)


- **Abiotic (EO:0007191)** - *Seasonal* (EO:0007027) – *Summer season* object (EO:0007036).

- **Abiotic (EO:0007191)** - *Seasonal* (EO:0007027) – *Spring season* object (EO:0007037).

- **Abiotic (EO:0007191)** - *Seasonal* (EO:0007027) – *Autumn season* object (EO:0007038).


• Abiotic (EO:0007191) – Chemical regimen (EO:0007189) - Nutrient regimen object (EO:0007241) - Mineral nutrient regimen (EO:0007239) - Micronutrient regimen (EO:0007240) – Chlorine nutrient regimen (EO:0007244),

• Abiotic (EO:0007191) – Chemical regimen (EO:0007189) - Nutrient regimen object (EO:0007241) – Mineral nutrient regimen (EO:0007239) - Micronutrient regimen (EO:0007240) – Zinc nutrient regimen (EO:0007309),

• Abiotic (EO:0007191) – Chemical regimen (EO:0007189) - Nutrient regimen object (EO:0007241) – Mineral nutrient regimen (EO:0007239) - Micronutrient regimen (EO:0007240) – boron nutrient regimen (EO:0007273),

• Abiotic (EO:0007191) – Chemical regimen (EO:0007189) - Nutrient regimen object (EO:0007241) – Mineral nutrient regimen (EO:0007239) - Micronutrient regimen (EO:0007240) – Molybdenum nutrient regimen (EO:0007253),

• Abiotic (EO:0007191) – Chemical regimen (EO:0007189) - Nutrient regimen object (EO:0007241) – Mineral nutrient regimen (EO:0007239) - Micronutrient regimen (EO:0007240) – Manganese nutrient regimen (EO:0007387),

• Abiotic (EO:0007191) – Chemical regimen (EO:0007189) - Nutrient regimen object (EO:0007241) – Mineral nutrient regimen
(EO:0007239)- *Micronutrient regimen* (EO:0007240) – *Copper nutrient regimen* (EO:0007261),

As part of Stage 2, linkages are described. For example, Table 3 shows such linkages between the FORM, FUNCTION, and ENVIRONMENT domains.