INNOVATIONS PROJECT REPORT

Procedural Vine Generation

BACVA3

Stanislav Shcherbakov (d1113644)

Contents

Stanislav Shcherbakov
d1113644
BACVA3

"Stealing from one source is plagiarism,

stealing from many is research"- unknown source

Abstract

Using computer graphics in order to animate growing vines and trees is rapidly becoming a widespread technique. It is used in a vast amount of films, music videos, commercials, and games. This project is focusing on exploration and analysis of the means of vine creation and growth. As the result of the analysis the proof-of-concept tool for generation of procedurally animated vine will be implemented, via MEL script.

This report will be split into three major sections:

- **Introduction & Research** that will include background research, aims and objectives of the project.
- **Development** that will document the personal achievements
- **Evaluation** that will include the critical analysis of the produced work, evaluation, possibilities for further improvement and the set of drawn conclusions.

Introduction

Aims

The aim of this project is to research and document the currently existing means for producing animated vine growth.

Following the analysis of the collected reference, an implementation of a proof-of-concept Maya tool, will take place. The tool's aim will be to allow user easily implement the vine growth animation within a given scene.

Objectives

Research & document existing techniques of vine animation

- Investigate the topic using relevant sources, such as Books, Research Papers, Films and the Internet
- Critically approach each studied technique, find its strong and weak points and analyse how it relates to the personal solution.

Creation of a proof-of-concept Maya Tool

- Create a simple program that models still vines
- Proceed to creating a procedural process for animation of the vines
- Implement the vine shape change, according to the surface in the scene
- Create a graphical interface allowing users to get an easy control of the program by interactively changing the vine attributes.

Research

Natural Background

Vine animation being the simulation of the real nature process, provided a good starting point for the research. Looking into growth patterns of real vines and trees has been established as a first step towards achieving project's goals, following a belief that good simulation of a process can only be achieved through the thorough study of the original process.

The word vine is derived from Latin *vīnea,* and was originally referred exclusively to the grape bearing plant. More recently its use has been extended onto any similar climbing or trailing plants.

Certain plants always grow as vines others grow as vines only part of the time. Examples of the second type would be *poison ivy* and *bittersweet* that grow as low shrubs when there is no support available. As soon as there is support around, they turn into vines.

A vine's growth is based on long and flexible stems. This type of growth has two purposes. Primarily a vine uses support such as rock exposures and other plants, for growth, avoiding the need for investing energy into the creation of supportive tissue. The vine growth also allows plants to take over large areas quickly.

A climbing habit has evolved independently in different plant families, resulting a number of different climbing methods. Some climb by twining the stems around the support. Others climb via the use of clinging roots, or using tendrils. Specialized shoots, leaves or inflorescences could play the role of the tendrils. An odd group of vines exists, called *climbing ferns*. In this case instead of using a stem to climb, the plant uses fronds. These fronds unroll from the tip and theoretically never stop growing. The fronds sometimes form copses whilst unrolling over the supporting areas in face of plants, rocks and fences.

At this point of my research I had to choose one of the types of vine growth, in order to continue examining the subject, focusing on that type mostly. After looking at the different methods that are used by vine plants for climbing, I have chosen to look into the *root climbing* plants, as the most widely used in films and video productions.

Further research of *root climbing* plants, has brought me to the theme of *aerial* and *aerating* roots. These are the type of roots, which are notable for growing above ground. Vine plants such as ivy use aerial roots in order to cling to houses, neighbour plants, rocks or any other type of support.

An interesting example of the aerial roots being used as a support for the plant, are the Banyan trees, otherwise known as strangler fig. The tree begins as a small aerial root, which then grows down the trunk of the host tree, sprouting branches along its way, until finally reaching the ground and absorbing the minerals and water, eventually strangling the host tree. The hollow cylinder of aerial roots is used as a banyan tree trunk.

Traditional Approaches

Traditional approach to solving the problem of vine growth would have either been using *stop motion animation* or creating a full-scale *traditional animation*, depending on the specifics of the case.

Traditional Animation also known as cel / classical or hand-drawn animation is the oldest form of animation where each frame is drawn by hand. A rapid flipping through those frames produces an illusion of movement. Apart from creating traditional animated characters, objects and backgrounds, traditional animation is known as technique creating special effects such as smoke and lighting.

Stop motion animation is the technique that makes static objects appear to be moving. The effect is achieved by moving the animated object by a very small distance between each individual frame, producing the effect of motion. It is very similar to traditional animation in that sense. Before computer animation became widespread it was one of the most common ways of achieving special effects in films. Most notably used for such films as “Star Wars”, “Terminator” and “Robocop”, stop motion has also been called to when animating the tree movements for such films as the “Evil Dead” series. More recently stop motion animation has been resurfacing with such motion pictures as “Wallace & Gromit: The Curse of the Were-Rabbit” and “Corpse Bride”.

Implementation in Media

Looking through different ways of vine and tree animation being implemented in media has formed the next step for this project. Analysing the strong and the weak sides of existing solutions has helped in refining the objectives and goals. Below is the list, resulted from studying the relevant media and short criticism focusing on the high-quality and low-quality sides of each offered approach. The list is sorted by the years of media production.

Evil Dead 1 & 2 (1981, 1987)

While these films do not focus specifically on the growing trees and vines, a lot of attention has been paid to the role of the animated "monster trees", and vines that had a tendency to wrap themselves around the characters. This film in a way was my first inspiration for doing this project.

Stop motion animation has been used in the production of both films for the trees and vines animation. Though very well implemented, and at a time of the film's making, being the best way of going about producing this effect, today it looks dated, and is hardly believable. It also lacks any particular style. It still impresses the viewer with the amount of work that was put into it however.

- *Pros*- innovative at the time of making, impressive usage of stop motion animation technique.
- *Cons*- looks dated today, hardly very believable in its animation, lack of motion fluidity, and does not look real or stylized.

Tiamat- Gaia (1994)

This music video contains some hand drawn animation of a growing tree. It looks like the technique of sop motion was used in this video, combined with the basis of a hand drawn traditional animation.

Produced result looks great in the context of the video, however can only be used for this type of video, as it is very stylized, and lacks a smooth flowing effect in the animation.

- *Pros*- this technique fits the video style really well, shows good use of stop motion technique
- *Cons*- can not be applied to other videos with the same effect, unless the videos are in the same style.

What Dreams May Come (1998)

This film has some very impressive effects when it comes to plants. The whole world gets painted before the viewer's eyes.

As far as I am aware Side Effects Houdini's L-systems have been used for producing all the stunning imagery of trees in this film. Taken out of the context of the film, this forms a very stylish piece of art.

- *Pros*- A great looking environment has been achieved by using Houdini. The effects look stunning even today. With tweaking a similar approach could be used in other media, to produce great results.
- *Cons*- There is little in terms of growth animation. Sometimes there is just too much going on at a given time, and the color palette of the plants seems to be overly vivid in places.

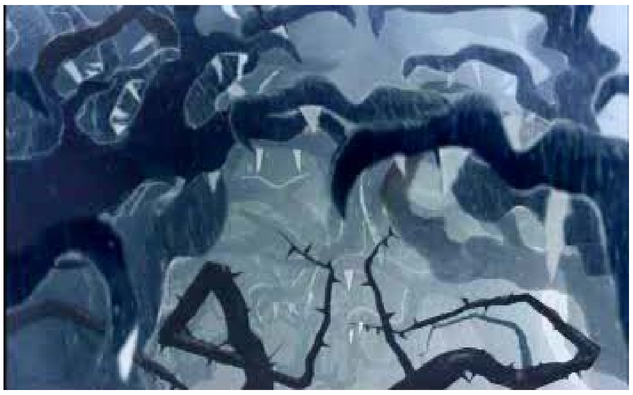
Marilyn Manson- The Nobodies (2000)

This video contains a couple of scenes in which characters get animated vines wrapped around them and finish being dragged into some kind of machine.

As far as could be told there was no CG animation involved in the making of the vines. They were all hand animated. By the looks of it the stop motion technique was not applied here, but rather the vines were just dragged towards the characters by a piece of string. Using a vast amount of rapid cuts, and gloom lighting, a this video achieves a very realistic look.

- *Pros*- good effect was achieved, low expenditures in setting it up.
- *Cons*- hardly suitable for longer duration shots.

Cradle of Filth- Ghost in the Fog (2000)



This video has some nice animated vines. They are however 2D, and most likely done as a postproduction via a program similar to Adobe After Effects.

It does however satisfy the needs of this video, and adds a nice style to the rest of it.

- *Pros*- *well* executed, stylized animation of the vines,
- *Cons*- perhaps slightly too cartoon-like, 2D

Radiohead - There there (2003)

Radiohead's video "There there" also deals with the concept of a "live" forest, which doesn't take kindly to strangers. There is not much on the subject of vines in this video, but it does deal with growth of trees.

As with the Evil Dead films mentioned above, this video has gone about solving the problem of animation with stop motion. Here however, it was not used in order to get the realistic looking animation, but rather to achieve the stylized effect, in which it succeeded really well.

- *Pros*- great use of stop motion animation technique, overall style.
- *Cons*- lack of fluid motion resulted because of the use of stop motion, has a very specific style and therefore cannot be successfully used in the media of a different style.

Brothers Grimm(2005)

This film has also used Side Effects Houdini in order to produce trees and has some great effects when it comes to animation of those. While it does not describe the growth of trees, there are some great vines/branches that snatch things away from the characters. While certainly looking very stylized, these branches have a great effect.

- *Pros*- impressive effects of tree animation, stylized look, with tweaking could be used in a variety of different styles.
- *Cons*- none was found, except for the lack of vine growth that makes this film slightly less relevant to my research.

Applications in Computer Graphics

L-Systems

One of the most common ways of describing the plant growth is the Lindenmayer system, more commonly referred to as L-system. This system presents a set of rules and symbols, mostly widely used for describing the growth processes in plant development, though they also have the potential for being used in outlining the morphology of variety of organisms.

L- systems were first introduced in 1968 by the Hungarian theoretical biologist and botanist, Aristid Lidenmayer (1925 - 1989).

L-system's central concept is based on rewriting. Using this technique, complex objects are defined by successively replacing parts of a simple initial object. A set of rewriting rules is used in this process.

A simple example of this is the snowflake curve proposed by Koch von Koch in 1905.

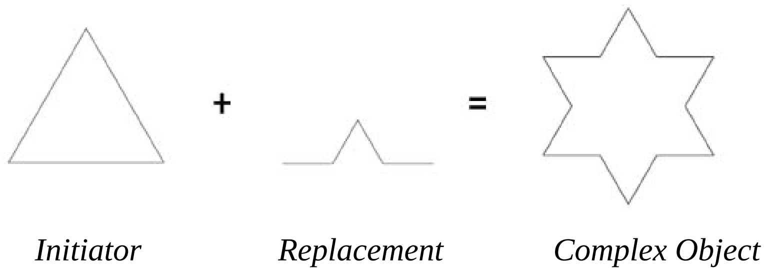


Figure 1

Further recursion results in more advanced structures, such as shown below.

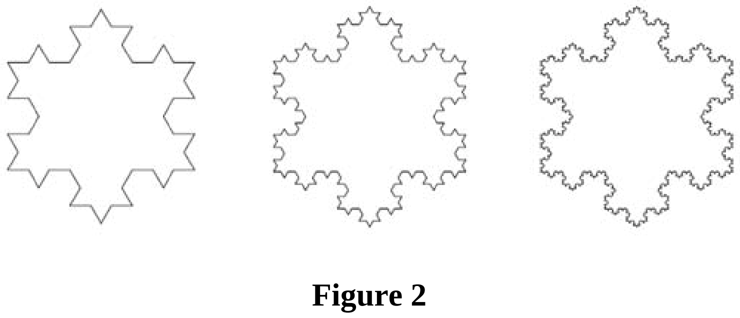


Figure 2

DOL- Systems

DOL- Systems present the simplest class of L-systems. They are deterministic and context free. They present a good opportunity for understanding the L-systems mechanisms of work.

Here is the simplest generic example for understanding these. Lets consider two letters a and b. For each we specify a rewriting rule. a -> ab stands for letter a being replaced by the string ab, subsequently b -> a means the replacement for the letter b with the letter a.

The process of rewriting begins from a string called an axiom. Assuming that b is the axiom, allows us to trace the process of further developments.

First step will cause b to be replaced by a, following the second rule. In the next step a will be replaced by ab, following our first rule.

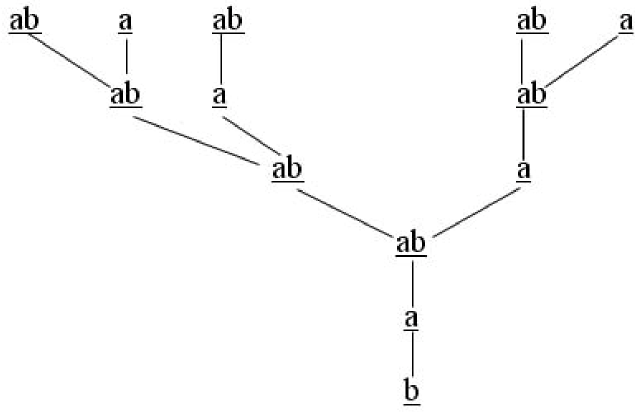


Figure 3

The string ab consists of two letters a and b both of which are replaced in the next step, resulting in a string aba. Further steps will produce strings abaab, abaababa etc. (**Figure 3**)

Graphic Interpretation of Strings

To turn L-systems into a versatile tool for plant modeling, several geometric interpretations have been introduced.

Logo system uses Turtle geometry to draw these interpretations. A state of the turtle is defined by a triplet *(x,y,a)*, where *x* and *y* are the Cartesian coordinates and a is an angle, that defines the direction in which the turtle is facing. If we also introduce the step size *d* and the angle increment *b*, then turtle can respond to the following commands:

F – moves forward by step size d, and draws a line;

F – moves forward by step size d, without drawing a line;

+ – turns right by an angle b;

- – turns left by an angle b;

Bracketed L-Systems

Up until the introduction of bracketed L-Systems the turtle interpreted each string in a “head-to-tail” manner. The results could produce complex figures, but they all consisted of a single continuous line.

Lindenmayer however introduced a notation for describing the branching process. Thus L-systems alphabet got extended by two symbols “[“ and “]”. The turtle interprets these symbols as follows:

[– pops the state from the stack, and makes it to be the current state of the turtle

] – pushes the current state of the turtle onto the stack

Existing Software Solutions

2D Solutions

There exist a number of programs for creation of vine growth as effect. They are all fairly effective, and can be used quite successfully for producing simple but effective animations. Macromedia Flash, often used for providing solutions for the website creation is perhaps the most notable of them.

Some 2D software allows user to go for a more advanced approach. Programs such as Adobe After Effects and Shake allow user to combine live footage with different effects in the process of postproduction. By using still, pre-drawn images of vines and a set of animated masks, a user can achieve quite impressive results.

This project however is more concentrated on the 3D side of computer graphics, so we will not focus on the 2D solutions, although it was felt that they were worth mentioning.

Side Effects Houdini

Houdini is a technical animation environment that has long been recognized as one of key contributors to feature film production pipelines. It presents a popular choice amongst technical directors working on high-end visual effects.

The part of Houdini, most relevant to this research is presented by the inbuilt L-systems. Placed under Surface Operators (SOP), the L-system primitive allows user to create an object, defined by a set of production rules. Controlled level of recursion allows user to specify the wanted complexity of the final object.

The production rules are based on the set of strings that follow Lindenmayer formulas described above, and are available for modification by the user at any given time. Houdini processes these strings and as a result generates an object in the 3D space.

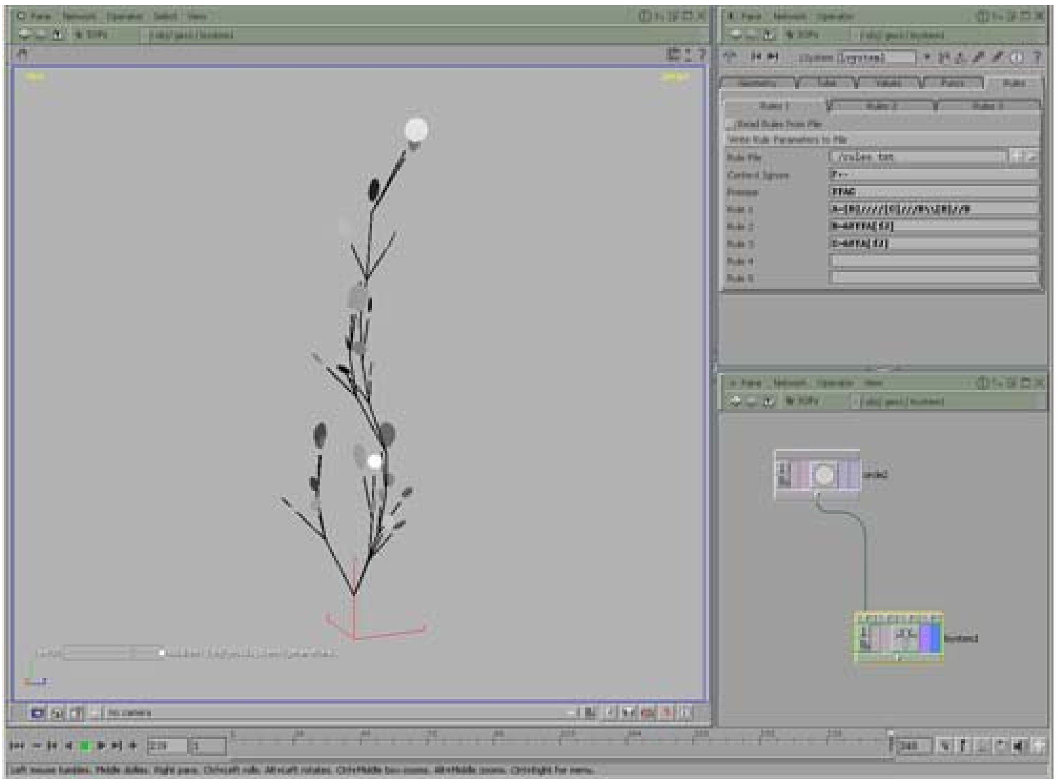


Figure 4

Initial creation option for Houdini's L-system geometry is a line representing each single branch, much like when using Turtle graphics. This can be altered, by using substitution with 3D geometric primitives (circles used for leaves **Figure 4**) or user-defined geometry. This aids in generation of a visually complex structures.

As well as being a powerful procedural modeller, Houdini also provides an option to animate the created model by setting keys to its attributes or via using run-time expressions. Houdini also allows user to light, texture and render the final result.

All of the above makes Houdini a very powerful tool for the creation of animated sequences of growing trees and vines. It does have one drawback however. While in the hands of an experienced used Houdini is able to produce great-looking results, Houdini suffers of being very complex software. Creating an impressive result requires an advanced knowledge of this package, and a large amount of tweaking. A novice user would have to spend a lot of time going through documentation, and spend a lot of time learning even the basics of this program. Although the documentation of Houdini is highly detailed, the lack of many good tutorials on the Internet makes the learning process even harder.

Xfrog

Xfrog is a Windows-based organic modelling and animation software. It comes both as a standalone program, and as integrated software for such packages as Maya and Cinema4D. While I did not have an opportunity to test it myself research showed that it is quite widely used, and the results achieved with it are quite good.

According to the official website, Xfrog offers a set of components, each having a different function. Those are used to describe the structural and geometrical specifics of the plant. The geometry of the plant is generated following the description set by each of these components, which are mapped to a graph consisting of nodes, much like Houdini.

Xfrog also provides an opportunity of animating these components, by changing their parameters, such as number of branches, crookedness, randomness, gravity, etc. Following these techniques, convincing trees/vines can be modelled and their growth animated.

Also being compatible with Maya, this program allows user to further animate produced trees within Maya by affecting them with Dynamics.

Overall this looks like a great solution, it does however imply the need of purchasing and learning a new package, which can be both time consuming and expensive. I also have not been able to find any information on this interaction of this program’s output geometry, with the existing geometry of the scene. It looks like a user would have to manually set up all the interactions, in each single case.

Maya Paint Effects

Another way of going about creating growing vines on a surface of an object or in space is via using Maya's Paint Effects. The mechanics of their creation process is quite simple. A user creates an object in 2 dimensions, which automatically gets converted into a 3D object by the software, as a post-processing effect. This results in the process often being referred to as a 2½ D system.

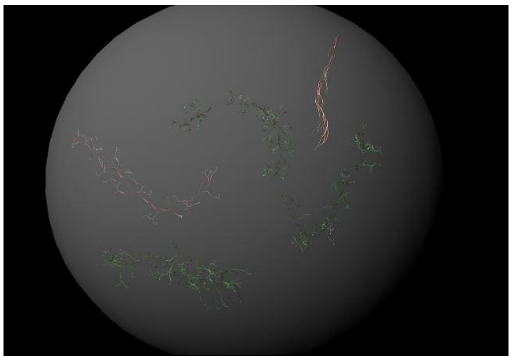


Figure 5

Paint effects contain a number of different types of brushes for creation of vines. Each one has a different look, but essentially they are very similar in the ways of their parameters and attributes. Each type allows user to change such attributes as density of the vine, presence of leaves and flowers, etc. The vine can also be animated by key framing certain attributes, or adding expressions to them. The results are produced very quickly, and are good for speedily needed solutions.

Paint Effects however do have some limitations and drawbacks. To start with, in their default state they can be rendered only by the Maya Software renderer. This causes a problem for those wishing to render their projects using a different option. They are also not included in ray-tracing calculations, and will not be rendered at all, if placed behind a semi-transparent object.

There is a solution to these problems of course, which presents itself in the fact that Maya allows users to convert Paint Effects into Polygon meshes. This does have a drawback of resulting mesh often being highly dense, which results dramatic increase of rendering times, and overall scene size.

Another noticed problem is the fact that zooming with camera on the vines produced by means of Paint effects, clearly reveals the fact that they are made out of individual bits, rather than being one object (**Figure6**)

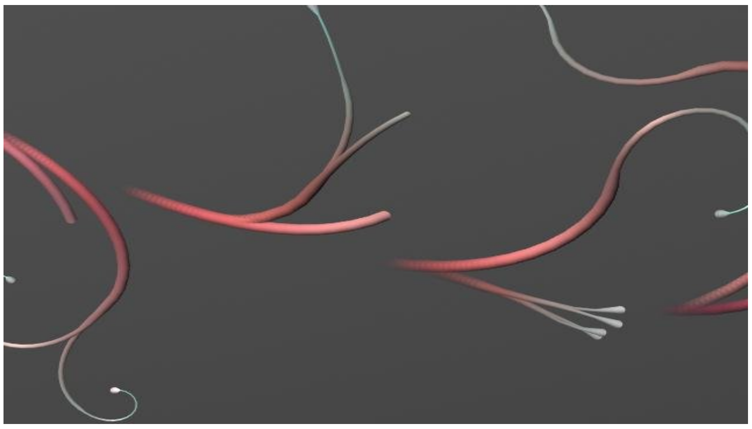


Figure 6

Paint effects are also looked down at by many people in computer graphics industry, as not being of a high enough quality, which is true to an extent. While being a good solution for a fast creation of vines and trees, the overall result suffers from looking rather generic, and can nearly always be distinguished as Maya's Paint effect. This can be corrected by a lot of tweaking however, and overall Paint Effects are capable of satisfying a lot of user's requirements.

Research Conclusions

Analysing the research, it is pretty safe to say that vines, as well as any growing plants, present a complex structure that requires a careful thought and many different approaches in order to be represented carefully in the medium of computer graphics.

In order to achieve realism in the behavior of the vines many factors must be taken into the account, such as different environmental factors that directly affect the their shape and appearance. For example a differently shaped surface calls for the vine to grow differently, otherwise the final result will hardly look convincing, with vines passing through the surface, rather than repeating its contours. Other factors must also be taken into the account such as general size of the produced vines, the angles at which they branch and the general direction of its flow.

The techniques presented above offer a variety of solutions that allow users to solve this type of problem. Analyzing and comparing them results in noting the strengths and weaknesses of each approach.

Turtle graphics provide a great starting ground for a user, and allow them to explore the processes of grammar parsing. Using them however only produces 2D results, which are quite primitive.

Houdini allows user to take a leap forward and create professional imagery. Being a very powerful tool it allows user to have a great control over the produced work, however it does fall short in cases where a novice user requires speedy results.

As mentioned above, the chance of testing Xfrog did not present itself. Forming a judgment upon the found tutorials and descriptions, though shows that aside from it being a great, and simple tool for creation of animated trees, flowers and vines, it has little to offer to user where the interaction between the created vines, and the scene environment is required.

Maya’s Paint Effects seem like possibly being the best solution for a novice user, however they do present some problems that have been described above. They are often discarded my advanced users as being “not worthy”, and they do not fully automate the process of creating the animated vines on the surface. This means that a new user would still have to spend some time, going through the brush’s attributes, in order to find the ones needed to be changed in order to get strokes animated.

Analysing the means of creating animated plant growth has provided a valuable experience and an understanding of the concepts and processes behind these systems. It has also exposed the strengths and weaknesses of existing solutions. With these conclusions in mind it is time to progress onto the development stage of this project.

Development

Exploring Houdini

The first step towards achieving my goals of creating a proof-of-concept Maya vine creation tool was to get a better understanding of the general processes behind the creation of animated plants.

One of the first steps was to get familiar with a book called “The Algorithmic Beauty of Plants” by Przemyslaw Prusinkiewicz and Aristid Lindenmayer. The book describes the L-systems in detail, and provides a great starting point for exploration.

Houdini, due to its inbuilt L-system primitive, has been the package chosen for the L-systems explorations. Initial tests produced some structures that although reminded trees, had absolutely nothing to do with the knowledge of the program. Simply by substituting rules with similar ones one can get interesting results from Houdini, without even having much knowledge of the package.

When one gets the hang of the syntax of rules however, everything starts to make a lot more sense, and the results become completely controlled by the user, allowing for creation of complex objects of the required shape, look and animated behaviour.

In my case the lack of experience with this package showed in the results produced, however since Houdini only served as an experimentation and learning tool these results were acceptable.

Exploring Houdini proved to be an important personal experience that helped in understanding of the principle of the concept behind the L-systems. Houdini has proved to be very complex software, advanced learning of which in itself could become a separate project.

Initial Plans

Initial plan was to produce a Maya tool, using the MEL scripting language allowing a novice user to quickly and easily create a set vines in a given scene, and if chosen by the user follow the shape of the surface of a specific object in a scene.

In order to proceed with this problem it was decided to break it down into smaller problem chunks that would need solving, successively as the project developed. The key breakdown was implemented as follows:

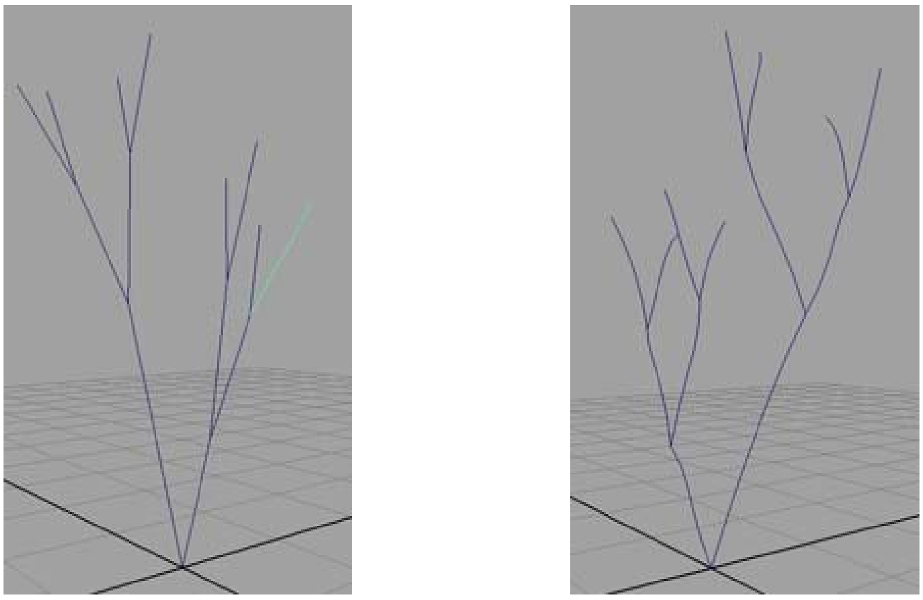
- Create a simple recursive curve generator that would form a tree/vine shape.
- Proceed and procedurally key the correct fames on the correct parts of the vine that will automatically produce the animation of the vines.
- Make the vine aware of its surroundings.
- Create a Graphical Interface, allowing user to easily interact with the program without the need to resort to the code every time an attribute requires a change.

First Step

First step after splitting the problem was implemented in the following way. Browsing the Internet showed numerous resources thoroughly explaining the implementation of such a generator. The one that used as a starting point for this project is located at this address - http://www.fundza.com/mel/tree/tree.html

It produces a simple tree that is made out of separate EP curves. While it served as a good base, it was not producing quite the wanted results. After the script being carefully analysed, it was rewritten it in a more suiting style for the project, adding some useful features on the way, and avoiding the excessive code that was not doing any good for the program, in the final code.

For example the original code was producing curves that only had two control vertices, which made the tree look very rigid and unnatural. The final code corrected that by rebuilding the curves and introducing two extra control vertices in the middle that were displaced to produce a more naturally flowing result.



Original **Modified**

Figure 8

Original code also had a lot of unnecessary conversions from angles to radians, and back, which did not do anything for the program. Many of those were excluded. There also have been some if statements that could never happen, therefore those were omitted in the final version as well.

While the initial bit has worked out quite well, it was felt that some volume needed to be defined. Otherwise the vines could not get rendered and presented little interest. In order to achieve this, a circle per branch of this structure, was added and placed at the base of corresponding branches. This has produced the result that seen on **Figure 8** below. Each new level received a set of smaller circles compared to the previous, creating the effect of the braches getting smaller closer to the end of the vine.

In order to achieve the volume, each of these circles got extruded along the corresponding curve, with a scaling factor, equal to the scaling factor of each circle, making sure that the branches became progressively smaller.

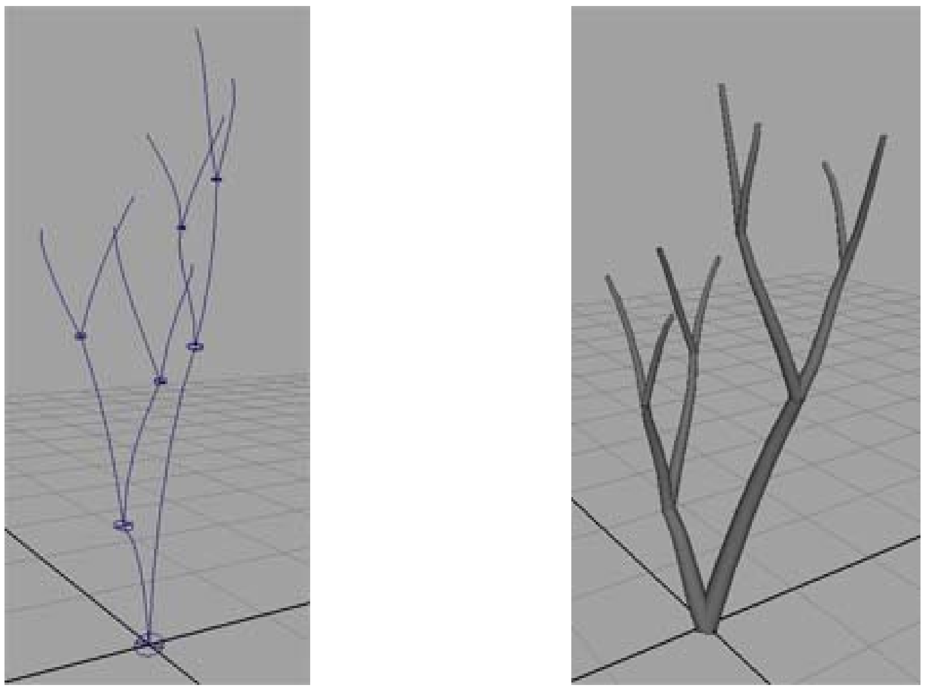


Figure 8

Step Two

To create animation of vines growing, it was decided to animate the extrusion values. This is a relatively simple procedure to implement even without the use of MEL script, so it was fairly quick to implement it, however the initial result could hardly be called satisfactory. All branches started the extrusion at the same time, which did not produce the wanted effect of growing, but rather just looked strange.

Solving this seemed as an easy task at first, as all that needed doing, was telling program to extrude the bottom branches first, the next level branches second, and so on. The problem appeared however, in the naming convention of these branches, caused by the implementation specifics of the original modelling script.

A script would run the set number of recursions producing the first number of branches, and then it would run through the next set of recursions producing the mirrored branches. After which it returned to the first set of recursions again. Thus the branches did not follow the standard naming convention, but rather depended on a mathematical formula for binary trees.

Thus the two bottom branches would be called

curve1;

curve(2^recursions);

the next four branches would follow the following naming conventions

curve[1+1];

curve [(1 + 1) + [(2^(recursions - 1) – 1]];

curve[(2^recursions) + 1];

curve[(2^recursions) + 1 + [(2^(recursions - 1) – 1]];

This style progressed. With each new level the number of new branches was increasing, and the formula was slightly changing to accommodate the new recursion step. While being relatively easy to work out on paper, I found it hard to implement into the code, so the alternative way was sought, in which the correct branch numbers for each level calculated by formulas outside of the program, were assigned to separate arrays in order to divide these levels in the final code. The result worked quite well, with only limitation being the fact that the recursions now can only be used up until 5, since I have not calculated the branch order for recursion levels higher than that.

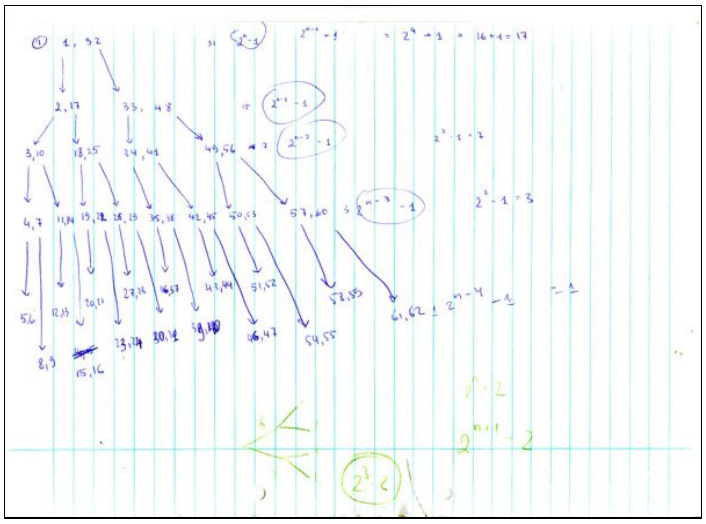


Figure 9

Step Three

Making the vine to be aware of the surface on which it had to grow was perhaps the most nerve-wrecking task during this project. At the beginning the solution was sought in the area of direction vector changing. When something came in the way of the vine, direction vector would be changed to a new value. After the object was passed, the vector would be reset to the original state.

A couple of tests were done with a plane. They proved satisfactory, but only allowed the vines to avoid that plane. This was certainly a good achievement, but it failed when trying to make the vine shape to mimic the shape of an object.

The process was controlled by a small set of commands in the program. They looked the area in which the plane was located, by checking the world space coordinates of its corner vertices. Then if the vine was intersecting the plane, it was told to alter its the direction vector, which caused the vine to bend around the plane (**Figure 10**)

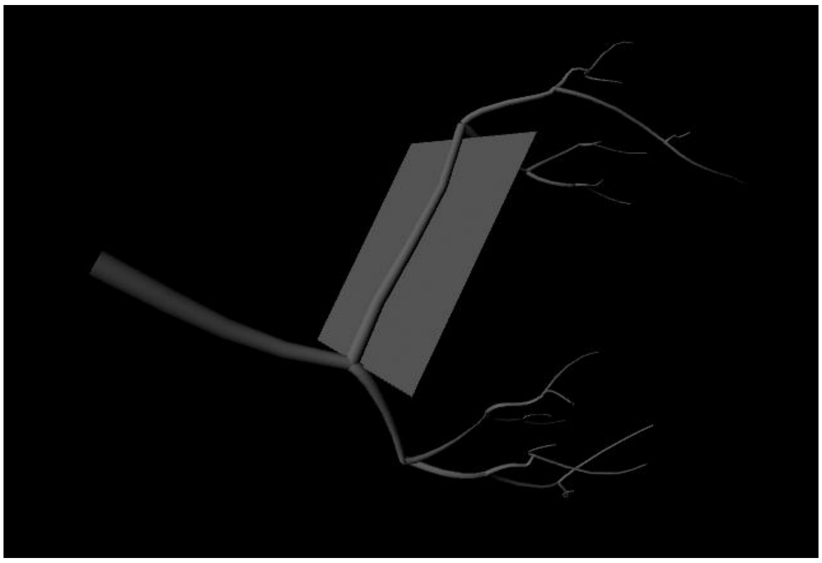


Figure 10

This concept worked quite well for avoiding collision of vines and objects in the scene, by using a similar concept to that of calculating collisions in games. Placing a simple plane in front of the object would cause the collision to be avoided, as the plane bent around the plane.

While this certainly worked, it did not really follow the original idea, in which it was intended to implement the vines that would change shapes according to the surface below. Around this time another idea has presented itself to me. Remembering the lectures from the first year of this course, I have recalled of projectCurve command.

This proved to be the turning point for the project, as everything started to finally fall into place. The final script worked in the following way:

- An object was placed into the scene.
- The vines were drawn somewhere in the space near the object.
- The vines were projected onto the object.
- Projected curves were duplicated in order to create fully functional curves, and all other curves in the scene were deleted.
- The curves were further tweaked to create a smoother flow, and also create some offset from the object, so that the vines would not look flat.
- The extrusion circles were moved to the corresponding bases on the new curves.
- Extrusion and animation were applied to the new set of curves.

This technique has worked out exactly the way that it was planned, therefore concluding the most important part in creation of the tool.

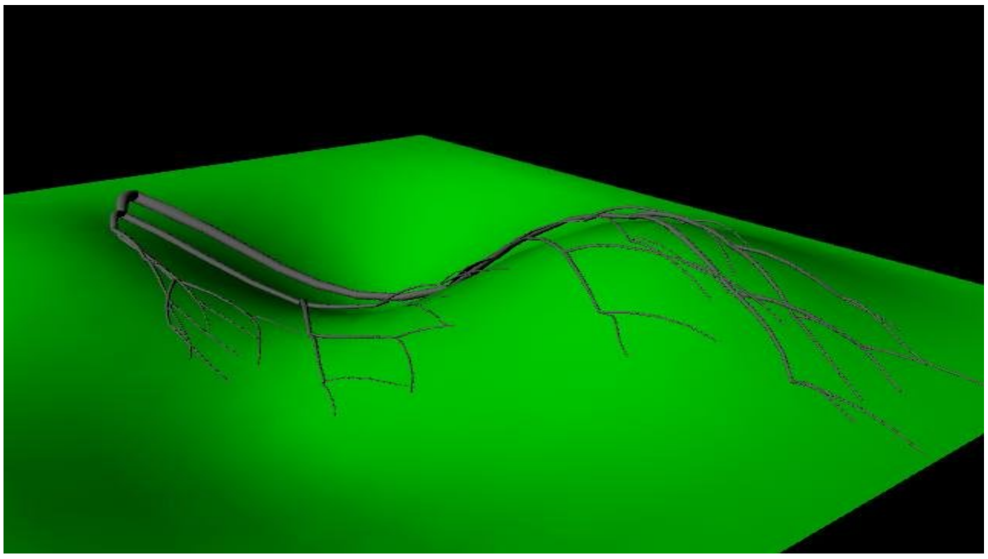
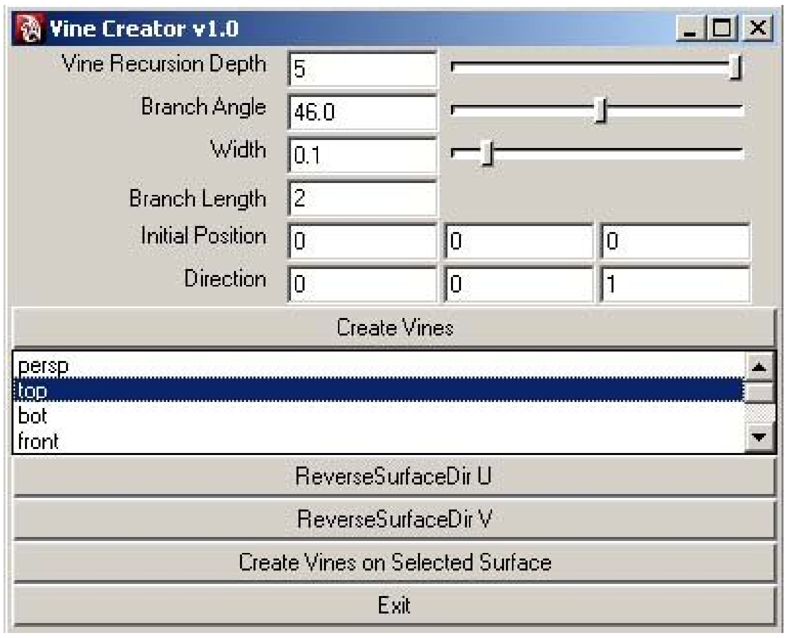


Figure 11

Step Four

Step four was to create the user interface that would allow the novice user to work with this tool, without having to face the code. This proved to be perhaps the part that was easiest in terms of implementation. At the same time it was an important touch that separated this tool from being a bunch of lines, into a fully functional interactive instrument capable of being used by people, who have no knowledge of MEL scripting.

The interface was created to be as user-friendly as possible. Once the script is sourced a user is introduced to the following window that allows user to specify parameters for the vine creation.



Vine Recursion Depth: 5
Branch Angle: 46.0
Width: 0.1
Branch Length: 2
Initial Position: 0 0 0
Direction: 0 0 1

persp
top
bot
front

Figure 12

The aim of this interface was to keep everything pretty self-explanatory. The options at the top allow user to specify the attributes of the vine they are creating. The “Create Vines” button just creates a set of animated vines, ignoring the objects. Options below it allow user to specify which camera to use for projecting the vines onto a selected surface. “Create Vines on Selected Surface”, projects the curves from the chosen camera onto the surface of the object. “Exit” quits the program.

Buttons responsible for Reversing Surface Direction help when the vines project onto the plane in the following way:

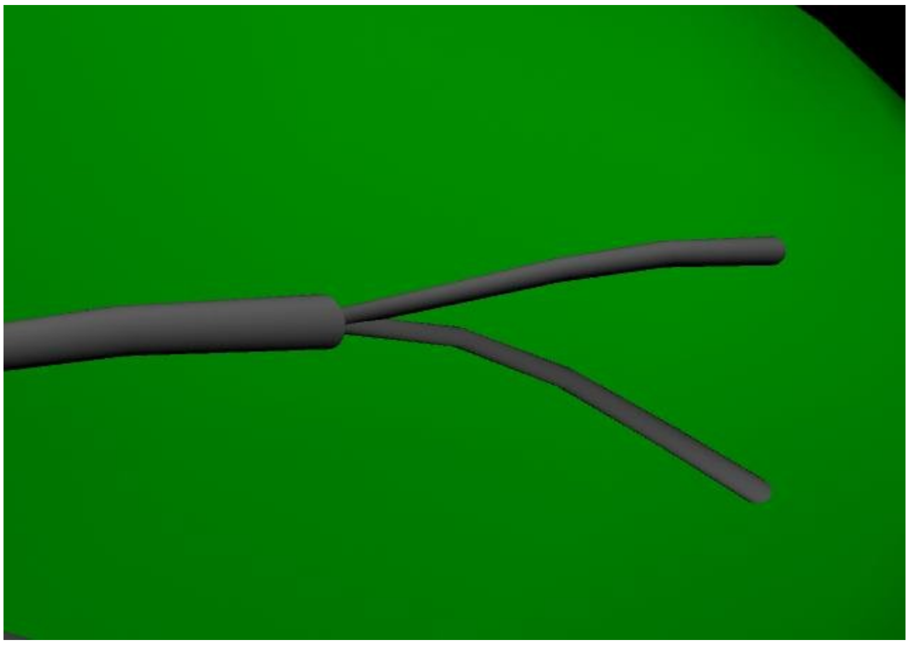


Figure 13

To correct such result user simply has to click one of the ReverseSurfaceDir buttons, depending on the position of the object and reapply the creation of the vines, after which normal result is expected.

Evaluation

Analysis

In order to conduct the evaluation of the project I have gone back and looked at the set aims. This has helped in drawing conclusions about the success of this project.

Initial purposes have been as follows:

- Research & document existing techniques of vine animation
- Create a proof-of-concept Maya tool

Let us start with the first one. Broad research has been conducted in the field of expertise of this project, using a number of digital and printed sources, including books, research papers and Internet. All related information was thoroughly studied and documented, and proved to be a very important and enlightening process. Approaching the project from this direction has definitely proved to be a correct step. And even though a lot of the received information was not used in the production of the final tool, this knowledge will definitely prove to be very valuable for the future projects.

Critically evaluating different techniques used in media has helped in drawing conclusions about the type of technique that would likely present the user with the best and simplest solution. Finding strong and weak points of each piece of work has also helped in formulating the needs for the creation of the final tool.

Looking into existing commercial software allowed seeing the techniques that were used by those, in order to implement similar problem solutions. Analysing these techniques and their implementations in the software enabled to produce a personal opinion on the matter, and helped in defining the further direction of the project was taking in the face of the tool implementation.

Overall I feel that this objective has been achieved quite well. Having been an important step it provided me with a lot of information needed to successfully carry out the second part.

Constructing the proof- of- concept Maya tool for creating animated vines has been the second aim of this project. Using the approach of breaking the problem up into a series of consecutive smaller steps has helped to simplify the task, and use a structured approach for its solution.

Doing some early scripting tests helped to get the hang of MEL scripting, and to establish the possible problems and their solutions, some of which did come useful later when the code got more complex. It is felt that the final program has fulfilled its purpose rather well, and although still being rather crude and unrefined, it is capable of producing some adequate results.

The movie files generated, show the results produced via the program. The vine structure and growth, although being rather generic does resemble that of a real vine. It does react to its environment, and can be used within the pre-created scenes, therefore fulfilling the objectives stated in the **Introduction**.

With project concentrating on a study of the subject, and production of a proof-of-concept tool based on the analysed research, rather then on creation of a beautiful imagery, it can be seen that the output of the tool does accomplish its task.

<u>Work Extensions/ Improvements</u>

This project may be extended in a couple of places. First and most important feature that would be introduced to this project with extra time would be the usage of L-systems. As of now vine generation process is based on a simple binary tree recursive method, which does create satisfactory results with higher recursions, but unfortunately fails in creating a completely natural looking vine. With the use of L-systems this project could be taken onto a new level of advancement, and could perhaps in time even made into a fully functional commercial software tool.

Another possible improvement for the future would be to implement the automatic wrap around Maya's Polygons as well as Maya's NURBS. Polygons in Maya do not allow curves to be projected onto their surface, however this problem can be simply solved in a somewhat similar fashion to the one used in games to calculate collisions. While at the moment it can be manually implemented by the user placing a NURBS surface of similar shape and a slightly larger volume onto the polygon and projecting the curves onto that, a great improvement for the future, would be to automate this process, therefore reducing the workload for the user.

Suppressing the Errors and Warnings, by introducing safeguards into the code, would be another thing that I would like to get improved in the future. While this tool runs stable when used correctly, it does tend to produce a lot of Warning messages when user attempts projecting onto the surface without selecting one. A simple code to check if anything is selected in the scene would be sufficient to solve this problem.

Implementing procedural texturing for the vines would also be another improvement to consider, as it would add realism to resulting imagery.

Adding supplementary geometry representing leaves and flowers is yet another thing that could add to the realism of the final image.

Conclusion

Overall I think this project proved to be a success. Using the analytical processing of the gathered information, has aided the development of the final solution. The animated vines, produced by the tool, having started with a couple of really simple curves progressed into advanced and complex structures that carry a strong resemblance to the real vines.

Personal experience gained by undergoing this project, proved to be very important. My scripting skills have improved, general problem solving techniques have undergone a change for the better. Ability to approach problems in a structured and organised way has been gained. Research involved in the making of the project proved to be a lot more extensive than was originally planned, which was also a great improvement, over the original arrangement, since it allowed me to look at my problem from a different perspective.

This project has also proved to me, my own capability of coding, and solving programming related problems, something that I have always been afraid to do, and avoided. I am glad that I did not shy away from this project, and carried through with it until the end, although admittedly there was a point when thoughts of switching to something that I had a stronger side for, appealed to me.

Acknowledgements

Special thanks to the people that have provided their support during the production time of this project, by offering valuable advice and suggestions:

- Ian Stephenson
- Jay Brindle
- Sebastian Huart
- Daniel Barrett

References

Books

- *“The Algorithmic Beauty of Plants”- Przemyslaw Prusinkiewicz, Aristid Lindenmayer;*
- *“Plant growth analysis”- Hunt, Roderick;*
- *“A colour atlas of plant structure”- Bowes, Bryan G. ;*
- *“Botany”- Larry Hufford*

Papers

- *“Parametric L-Systems and Their Application to the Modelling and Visualisation of Plants” - James Scott Hanan*
- *“Natural Systems: Simulated Tree Growth”- Kevan Shorey*
- *The Artificial Life of Plants Przemyslaw Prusinkiewicz, Mark Hammel, Radomır Mech*
- *“On Vertex-Vertex Systems and Their Use in Geometric and Biological Modelling” -Colin Smith*

Websites

- http://algorithmicbotany.org/
- http://www.highend3d.com/
- http://forums.cgsociety.org
- http://efg.cs.umb.edu/gallery/NFSPlants
- http://directory.google.am/Top/Computers/Artificial_Life/Lindenmayer_Systems/
- http://del.icio.us/dvmorris/lsystems
- http://www.fundza.com/index.html
- http://www.13dots.com/forum/index.php?showtopic=4545
- http://www.joncrow.com/tutorials/xsi_tuts/growing_flowers/grow_flower.htm
- http://www.tumbletruss.com/index.html
- http://en.wikipedia.org
- http://www.biologie.uni-hamburg.de/b-online/e28_3/lsys.html
- http://spanky.triumf.ca/www/fractint/lsys/plants.html
- http://www.math.okstate.edu/mathdept/dynamics/lecnotes/node20.html
- www.google.com