Informatics

Sketch based L-Systems for Tree Modeling in Virtual Reality

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Media and Human Centered Computing

eingereicht von

Maximilian Steiner, BSc
Matrikelnummer 01041121

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Mag. Dr. Peter Kán
Mitwirkung: Mag. Dr. Hannes Kaufmann, Univ. Prof.

Wien, 2. Mai 2023

Maximilian Steiner

Peter Kán

Informatics

Sketch based L-Systems for Tree Modeling in Virtual Reality

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Media and Human Centered Computing

by

Maximilian Steiner, BSc
Registration Number 01041121

to the Faculty of Informatics

at the TU Wien

Advisor: Mag. Dr. Peter Kán
Assistance: Mag. Dr. Hannes Kaufmann, Univ. Prof.

Vienna, 2nd May, 2023

Maximilian Steiner	Peter Kán

Erklärung zur Verfassung der Arbeit

Maximilian Steiner, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 2. Mai 2023

Maximilian Steiner

Danksagung

Ich möchte meine aufrichtige Dankbarkeit ausdrücken für alle die mich während dieser Arbeit unterstützt haben. An erster Stelle, Univ. Prof. Mag. Dr. Hannes Kaufmann, für die Freiheit, mein eigenes Thema zu wählen, mir zu vertrauen und mir zu ermöglichen, meinen Absichten zu folgen. Ich möchte auch meine tiefste Wertschätzung an meinen Betreuer, Mag. Dr. Peter Kán, für Ihre Geduld, Unterstützung und unersetzliches Feedback ausdrücken. Ihre Einsatz daran, mir dabei zu helfen, meine Arbeit zu verbessern und meine Ziele auszurichten, war wirklich unschätzbar. Schließlich möchte ich meinen herzlichen Dank an meine Partnerin ausdrücken für Ihre unerschütterliche Unterstützung, Ermutigung und Verständnis während des gesamten Prozesses des Schreibens meiner Arbeit.

Ich bin tief dankbar für all Ihre Unterstützung und Anleitung.

Acknowledgements

I would like to express my sincere gratitude to my professor, Univ. Prof. Mag. Dr. Hannes Kaufmann, for providing me with the freedom to choose my own topic, trusting me and allowing me to follow my intentions. I would also like to extend my deepest appreciation to my supervisor, Mag. Dr. Peter Kán, for your patience, support and invaluable feedback throughout this journey. Your dedication to helping me improve my thesis and aligning my goals has been truly invaluable. Finally, I would like to express my heartfelt thanks to my girlfriend, for your unwavering support, encouragement and understanding throughout the entire process of writing my thesis. Your presence and support meant a lot to me and helped me to stay focused and motivated throughout this challenging time.

I am deeply grateful to all of you for your support and guidance.

Kurzfassung

Diese Arbeit präsentiert einen neuen Ansatz in Virtual Reality, um Baummodelle für digitalen Content zu erstellen, indem ein skizzenbasiertes Vorgehen mit einem prozeduralen System kombiniert wird. Skizzenbasierte Modellierung in Virtual Reality kann schnelle Iterationen in der Erstellung von Baummodellen unterstützen, während ein prozedurales System Realismus und eine Vielzahl von generierten Assets aus einem Modell ermöglichen kann. Um dieses Konzept zu demonstrieren, wurde im Rahmen dieser Arbeit ein Virtual-Reality-Prototyp zur Erstellung von Baummodellen entwickelt. Skizzengestützte Modellierung wird in Verbindung mit L-Systemen verwendet, um die realistische Aststruktur eines Baumes zu generieren. Um die Vorteile der Kombination von prozeduralen- und skizzengestützter Modellierung zu erreichen, wurde ein 2OL-System, ein spezialisiertes Lindenmayer System mit kontextbasierten Regeln und Parametern, adaptiert, um auf den Input des Benutzers zu reagieren. Diese Arbeit betrachtet die Herausforderungen, die Vorgaben des Users in die Parameter und Regeln des L-Systems einfließen zu lassen, um ein interaktives prozedurales System zu implementieren. Zusätzlich werden die Vor- und Nachteile von einer skizzenbasierten Herangehensweise in Virtual Reality beleuchtet. Eine Benutzerstudie wurde durchgeführt, um die Benutzerfreundlichkeit des Prototyps im Vergleich zu einer Standardsoftware für die Erstellung von Baummodellen, SpeedTree, zu untersuchen. Die Ergebnisse zeigten, dass der Prototyp in Bezug auf die Benutzerfreundlichkeit mit SpeedTree vergleichbar ist, jedoch den zusätzlichen Vorteil bietet, Baumformen schnell durch Skizzen zu generieren. Diese Forschung trägt zu den Bereichen Virtual Reality, Baummodellierung und skizzengestützter Modellierung bei, indem sie einen neuen Ansatz vorstellt, der L-Systeme zur schnellen Generierung parametrisierter Baummodelle nutzt.

Abstract

This thesis proposes a new method for creating tree assets in virtual reality for digital content that combines a sketch-based approach, with a procedural branching system. Sketch-based modeling in virtual reality offers the possibility of quick iteration and creation of models while a procedural system can aid realism but also provide a large variety of assets from one model. To demonstrate this concept, a virtual reality prototype for tree modeling that combines sketch-based modeling with L-Systems to generate a tree's branching structure is presented. To achieve the combined approach of sketch-based modeling and procedural generation of trees, a 2OL-System, a specialized Lindenmayer System that includes context sensitive rules and parameters, was implemented and adapted to react to the user's input. This work explores the challenges of implementing an interactive L-System by including the user's choices as parameters into the context-sensitive and parameterized generation rules of the system. Additionally the benefits and pitfalls are discussed that come with a sketch based interface in virtual reality. A user study was conducted to compare the usability of the prototype with a widely used tree modeling tool, SpeedTree. The results showed that the prototype is comparable to the level of usability as SpeedTree, but with the added benefit of quickly generating tree shapes through sketching. This research contributes to the fields of virtual reality, tree modeling, and sketch-based modeling by introducing a new approach utilizing L-Systems for the quick generation of parameterized tree models.

Contents

CHAPTER 1

Introduction

1.1 Motivation

"I felt the world needed a tool for the spontaneous invention of new virtual worlds that would express the stuff of the mind that was otherwise impenetrable. If you could conjure just the right virtual world, it would open up souls and math and love." [Lan17]

Lanier describes elegantly the fascination with virtual worlds and the possibilities they could provide. Evidently, his enthusiasm is reflected by the market. The demand for video games is growing constantly. Alone in the US, the industry was estimated to be 85.9 billion dollars [Vid21]. Virtual Reality (VR) used to play a smaller role in the video game market but as the technology advances, we can expect as stated by Alsop that: "The introduction of smaller and more fashionable devices, will support the increased adoption of VR by consumers and across industries." [Als22] This includes applications outside of the video game genre as well. But any VR device would be useless without content that can be experienced by the users. Therefore, also the market for software, targeted at digital asset creation, has grown and is expected to grow even more in the future: "The computer graphics software market value in the modeling & animation software segment increased between 2012 and 2020, and is expected to further increase to reach 390 million U.S. dollars by 2024."[Vai22] The market value of video games and the demand for new content for virtual environments lead to the conclusion that there is space for new tools in the subject of 3D modeling. Thus, there is a lot of research dedicated to improving the creative process in 3D-Modelling. One particular topic, although it is present in almost every virtual world, seems to have a lot of space for optimization: vegetation. Flora is a crucial element of convincing virtual worlds and the structure of plants and trees has fascinated scientists as well as artists for a long time. Our work explores a way to utilize virtual reality in combination with a sketch-based interface and an underlying procedural algorithm to improve the process of plant creation for virtual environments.

1.2 Problem Statement

Interactive media requires technology to be fast and responsive. Therefore, developers always had to restrict themselves concerning the size and quality of the assets used in their software [Pot10]. As the hardware specifications of computers and consoles have improved, more sophisticated applications with better graphics became possible. This enabled the first larger virtual worlds with advanced 3D graphics. But developers had to make compromises to ensure a smooth user experience due to the hardware limitations. Commonly the focus during development is to complete the relevant application features, before polishing the looks because performance always is an issue [DJ13]. With the possibility of larger worlds, the exploration of virtual environments became a bigger part of video games. Characters were not constricted to a certain path anymore and could move around freely. More attention had to be paid to the environment and details of the virtual world. Therefore, developers have started to focus more on the surroundings. Forest environments and virtual trees used to be either cardboard cut-outs or repetitions of the same low-resolution model. Recent games have adapted to findings of the positive effect on the perception of the game world when there is a significant amount of green vegetation and dedicated more resources towards creating virtual vegetation [TPC18]. An early example is the video game "Crysis" which many see as a milestone in computer graphics in video games. It impressed the audience with its highly detailed graphics as well as the immersive vegetation design including hand-created wind animations on every individual plant [Kus09]. Although humans tend to notice plants less, their beneficial influence is undeniable. The phenomenon that humans focus their attention on moving things like animals and seem to ignore vegetation around them is called plant blindness [All03]. This could be an explanation for the lack of attention to vegetation in video games. Counteracting this with detailed forest environments and a deliberate focus on designing green foliage could not only have positive effects on the quality of the virtual worlds as well as the user's comfort but could also raise awareness towards plant preservation.

But as resources are limited it is understandable that development teams keep the vegetation design in their worlds to a minimum. Especially in early phases of game development or rapid prototyping, the focus lies on gameplay and not on the surroundings. But a convincing environment can do a lot towards understanding how a game would feel in its finished version. Therefore, developers often do use slightly more sophisticated geometric models in the early stages of game development. But for vegetation modeling there seems to be a gap between the usage of copy and pasted tree models and high fidelity models that are very time-consuming to create and the existing tools that are cumbersome to use. The existing solutions have various usability barriers, for example high learning curves, that are obstacles to accelerated content creation. Concerning the creation of living and growing organisms as complex as plants, a static CAD-like approach as it would be used in designing buildings or furniture might be unfeasible. As Anastacio et al. state:

> "[..] most illustrators agree that available methods of constructing, editing and manipulating 3D models [...] do not lend to a natural interaction metaphor and force them to diverge from their preferred ways of thinking and working." [Sou05] As cited by Anastacio et al. [ASSJ06]

Sousa et al. agree and state that many of the approaches in existing software do not support our human way of thinking and working. Often an advanced state of the design process is mandatory to get started in this kind of software. As Chen et al. state:

> "3D modeling systems are cumbersome to use and therefore ill-suited to the early stage of design (unless the design is already well-formulated)." [CNX+08]

One possibility to address the mentioned shortcomings of existing software might be sketch-based modeling [SOJ11]. Several studies have shown that a more natural human-computer interaction, utilizing sketch-based interfaces, can benefit the concept design process [UWC90, JM92]. Therefore, a sketch-based modeling approach could improve the overall creative process of designing virtual vegetation and enable the use of more detailed models in the early stages of development.

Another approach developers often choose is to use procedural modeling in early project stages as demonstrated by Saldana et al. [SJ13]. With tools like Houdini FX [hou] one can quickly create virtual environments with just a few assets. Procedural elements are also crucial for modeling virtual plants. By incorporating algorithms derived from nature, the realism of created assets can improve. The improvements manifest in the approximation of real features using fractals, recreating realistic branching structures with L-Systems and the variety of models that can be created [TSD16]. A model created with a procedural approach can be parameterized, and through adjustments or randomization generate a variety of plants from one blueprint. This does not only save time and money in development, but data-driven vegetation can reduce the workload for graphics software rendering the models as demonstrated by the developers of the game "Horizon Zero Dawn" [San18]. But to create custom trees with a procedural approach, one needs an understanding of the biological nature of the tree to get a realistic result. Those rules are often not easy to understand. Chen et al. further evaluate that:

> "Rule-based systems are difficult for novice users to operate because they require not only specialized knowledge of biomechanics and biology for effective parameter specification." [CNX+08]

Therefore the software solution for designing 3D tree assets should also facilitate algorithms and parametric design without increasing the cognitive load on the user too much.

1.3 Aim of the work

Our work suggests that a modeling environment that encapsulates the mentioned principles of sketch-based modeling combined with procedural algorithms can be beneficial for the creation of digital vegetation. On top of that, immersion in virtual reality and setting the user at a close distance to the created tree models, could provide a beneficial level of authenticity and the advanced input devices present versatile options for interaction and manipulation techniques of the object as discussed by Hand [Han97]. To explore if a sketch-based approach of procedural vegetation elements is viable in virtual reality, our work focuses on the trunk and branch creation of a tree as the branch geometry is suited well for visualizing L-Systems. An application was developed with Unity for the Oculus Quest platform to study the concepts in practice. To gain meaningful insights into the interaction in virtual reality, current standards for virtual reality development were followed. The best practices proposed by the Oculus developer guideline were taken into consideration to implement the interaction techniques.

1.4 Methodological Approach

1.4.1 Research Question

This thesis aims to investigate the following research question:

How does a sketch-based, procedural modeling approach using L-Systems, coupled with immersion in Virtual Reality, influences the process of modeling a tree?

The motivation for this research question, as was laid out in the introduction, is to possibly facilitate the creation of digital tree assets for content creation in video games or virtual worlds. The constraints of maintaining small and reusable assets, inferred by the working practices of the industry, are that the resulting tree is parameterized and can be shown in multiple growth cycles.

1.4.2 Literature Research

For the literature review of our work, we researched various topics. Amongst others, they related mainly to the fields of L-Systems, tree modeling, sketch-based modeling, 3D interaction techniques, human-computer interaction, and virtual reality. We examined the current state of the art in each of these areas and how they relate to our research topic. L-Systems have been widely used in the generation of virtual plants and trees, and have been extended to model a wide range of natural phenomena. Tree modeling techniques have been developed to create more realistic and detailed trees in virtual environments. Sketch-based modeling has gained popularity in the field of 3D modeling, allowing for more intuitive and natural ways of creating 3D content. 3D interaction techniques have been developed to enable more natural and efficient interaction with virtual environments. Human-computer interaction (HCI) plays a critical role in the design and evaluation of virtual reality systems, and virtual reality itself has been widely used in fields such as entertainment, education, and therapy. Overall, the literature review provided the foundation for our research and helped us understand the current state of the art and identify potential areas for further investigation. Our findings are laid out in the related work in Chapter 2 and the aspects that have influenced this work the most are discussed in more detail in Chapter 3

1.4.3 Prototype Implementation

Following the literature research a prototype was implemented with the goal of realising a tree modeling application in VR that combines the creative approach of sketch-based modeling with an underlying procedural algorithm to give users an immersive experience while creating reusable and realistic digital tree assets.

1.4.4 User Study

A user study was designed to gather insights into the interaction and use of the virtual reality application. The study focused on the trunk and branch creation of a tree. The goals of the user study were to evaluate the effectiveness and usability of the virtual reality tree modeling application, and to assess the risk for simulator sickness in users. We were interested in exploring whether a sketch-based, procedural approach to modeling vegetation elements in virtual reality could be viable, and whether immersion in virtual reality and close proximity to the created tree models could provide a beneficial level of authenticity and versatility in terms of interaction and manipulation techniques. To investigate the research question and gain meaningful insights the participants had to complete a comparable task in the industry standard software for tree modeling: "SpeedTree"[spe], as well as in the prototype developed for this thesis. To compare and evaluate the usability of the two applications the participants filled out surveys after each task. The detailed methodology of the user study and the evaluation can be found in Chapter 4. The content of the questionnaires can be found in the appendix.

1.5 Contribution to the field of research

Our work discusses how sketch-based modeling could benefit the process of modeling trees in virtual reality with L-Systems. As discussed in the related work section there exist several solutions that address individual aspects of this question. The approach of letting users manipulate L-System via sketch-based modeling could provide help for crossing the bridge between time-intensive professional tree modeling with a steep learning curve that results in realistic-looking trees, and the creative benefits of quick prototyping. The use of sketch-based, procedural modeling with L-Systems in Virtual Reality has the potential to provide a more immersive and intuitive experience for tree modeling, which could be useful for researchers and practitioners in fields such as computer graphics, virtual reality, and landscape design. The development of a virtual reality application specifically for this purpose could also be a useful tool for researchers and practitioners in these fields, as it allows for the exploration of tree models in a more realistic and immersive environment.

Overall, the results of this master thesis, combined with the virtual reality application developed for this purpose, could contribute to the advancement of research in these fields and potentially lead to new and innovative approaches to tree modeling.

1.6 Structure of the work

We first present the theoretical background in Chapter 2, which is based on a review of the relevant literature on sketch-based modeling, procedural algorithms, virtual reality, and tree modeling. In Chapter 3 we describe the prototype and implementation of the virtual reality tree modeling application, diving deeper into the relevant concepts from related work including the technologies and algorithms used. Following this, we report on the user study conducted to evaluate the effectiveness and usability of the application and discuss the findings in Chapter 4. Finally, in Chapter 5 we discuss the limitations and results of our work, and offer suggestions for future research and development in this area.

This works methodology involves the following steps. First, a literature research was conducted concerning the topics L-Systems, tree modeling methods, sketch-based modelling, 3D interaction techniques, HCI and virtual reality. The prototype implementation followed based on the findings from the research and finally, a User Study was conducted.

CHAPTER 2

Related Work

As Stava et al. state: "Plant modeling approaches can be categorized into three principal classes: reconstructions from existing real world data, interactive modeling methods and procedural or rule based systems." [SPK+14] One goal of our work was to explore ways for manipulating the form and branching pattern of a tree, while maintaining a realistic growth model. Therefore, the focus of our work lay on the latter two categories. In this Section, we will discuss related groundwork that focuses on interactive modeling methods and procedural or rule-based systems as a basis for the prototype of our work.

2.1 Interactive Modelling Methods

Similar to the approach of our work there have been previous advances in combining sketch-based modeling with the creation of trees. Pruscinkiewicz et al.[PMKL01] dedicated research in this direction and created a tool to manipulate plant structures based on L-Systems with user-defined curves. Liu et al[LJLZ12]. were able to generate three-dimensional models from two-dimensional input sketches. They require two sketches for their algorithm to work. One sketch represents the side view, and the other sketch the front view of the tree. Okabe et al.[OOI06] generate three-dimensional tree models from two-dimensional sketches by assuming a tree spreads out its branches to gain the most sunlight. They rotate the sketched branches until there is the optimal distance between them to generate a three-dimensional model. Chen et al.[CNX+08] developed another very interesting approach to creating a three-dimensional tree from a user's sketch. The user draws a simple representation of the desired tree's branching structure and can even add boundaries for the tree's crown with a separate stroke. The sketch is then transformed into a branch model by inferring the depth of each branch tip by assuming the drawing is an orthographic projection. The data structure is a Markov Random field that can be converted to a Markov tree. By using self-similarity, sections of the tree are chosen and filled in until the model reaches the user-drawn boundaries

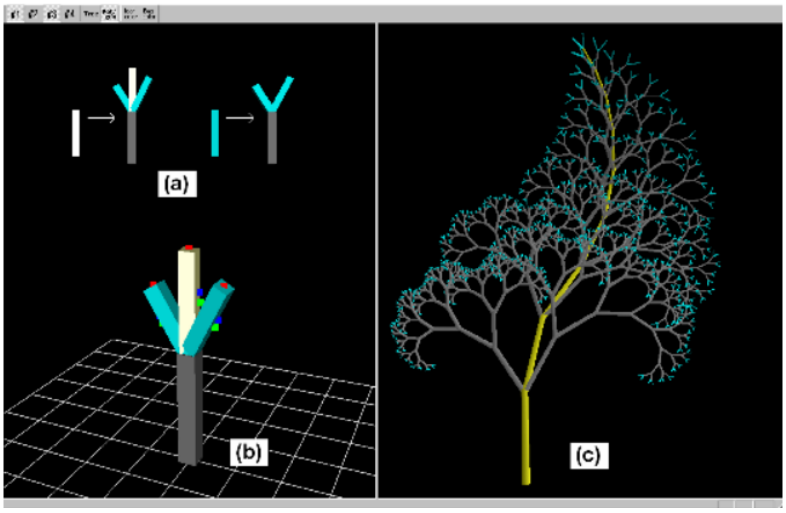


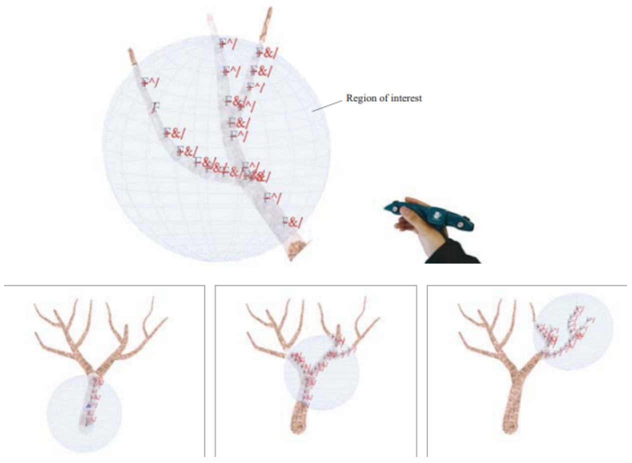
Figure 2.1: Example of a sketch-based L-System by Ijiri et al. [IOI06]

or a certain number of generations. Yuan and Huai argue that sketch-based modeling provides a lot of advantages for the task of modeling trees[YH21]. To overcome the problem of inferring three-dimensional structures from a two-dimensional sketch, they propose a prototype for virtual reality. With a head-mounted display and a controller, they implemented a tree sketching tool that can generate realistic-looking trees from a user-generated three-dimensional sketch. They achieve this by utilizing several algorithms to sample and optimise the user's input. But the resulting tree is not built upon a rule-based system. This means for the creation of multiple trees a user would need to create each tree individually. The prototype proposed in our work aims to implement the best of both worlds by combining sketch-based tree modeling with a rule-based system. There are several existing approaches that render a sketch-based tree from a user-created L-System that served as a reference.

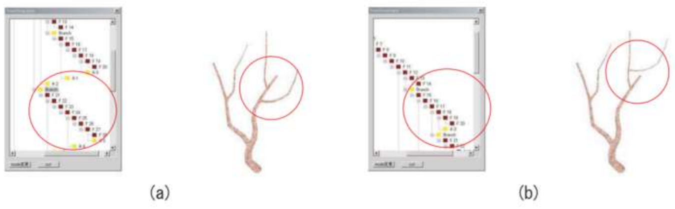
Ijiri et al. provide their users with two interfaces, one for defining the L-Systems rules, and the other for sketching the path of the trunk [IOI06]. The L-System is then generated accordingly along the trunk. The rules of the L-System are predefined. The user can change the length and direction of the resulting branches as shown in Figure 2.1.

In the prototype proposed by our work, the trunk and the branches are created in the same reference frame. The rules of the L-System are adaptable by the user directly by manipulating the sketch. No separate configuration is needed.

Onishi et al. have improved on the sketching part of the implementation by utilizing a 3D stylus to sketch the branches. This enables very finetuned control for the user and detailed manipulation is enabled like branch thickness. The L-System and its rules can also be modified within an extra window. The so-called "L-String" structure that defines the branching of the tree, is visualized in this second view as seen in Subfigure 2.2b [OMKK06]. But this process requires direct manipulation of the L-System string



(a) Sketching a tree with a 3D stylus by Onishi et al.



(b) Manipulating the L-System

Figure 2.2: Approach of modeling L-System with a 3D Stylus by Onishi et al. [OMKK06]

that might not be intuitive for newer users.

In our work, the manipulation of the L-Systems rules is not as granular but no second window is required and the user has a direct influence on the branching rules by manipulating the sketch. Exposed parameters are for example the rate of the branch and trunk growth as well as the consecutive branching and sub-branching rate. Our work takes a similar approach as the 3D stylus by utilizing a three degrees of freedom input method to sketch the tree.

The chosen user interaction methods for the prototype of our work are inspired by several insights in the field of 3D interaction techniques. When implementing gestures, for example, the deletion of a branch, correctly inferring the user's desired action is a common problem. One way to solve this problem is to provide the user with separate tools. But a sketching interface only maintains its practicality without an abundance of menus or buttons. There have been several strategies explored to implement more functionality into an interface while maintaining the sketching aspect. One possibility is gestural interfaces. An example would be the transformation strokes proposed by Severn et al.[SSS06] where a user can scale, rotate and translate objects by drawing a U shape. Tsang et al.[TBSR04] tried to solve it by introducing gestures for cutting and deleting nodes. When wanting to manipulate a drawn curve the user is supposed to mark it with an “N” gesture. Depending on how many strokes of the "N" shape the user completes a different action is taken. The first stroke of the "N", would mean, selecting, drawing 2 strokes (a "V" shape) means cutting and drawing the full "N" would delete the whole curve. When practised, this technique seems like an efficient way to manipulate the user’s sketch, but new users could easily forget the instructions and input unwanted actions.

2.2 3D Interaction Techniques

Virtual Reality opens up new opportunities for implementing 3D interaction techniques as demonstrated by several approaches discussed in Section 2.1. Interaction design in virtual reality is a delicate process that requires careful consideration of several key aspects. One of the main challenges is balancing the need for immersion and interactivity with the need for usability and accessibility. VR experiences can be highly immersive, but this can also make them overwhelming or confusing for users who are not familiar with the medium. Designers must consider the cognitive load of the interface and ensure that the interactions are intuitive and easy to understand. Additionally, designers must consider the physical limitations of users, such as the range of motion and dexterity required to use certain input devices. Ensuring that the interface is comfortable and easy to use can help prevent fatigue or discomfort during long sessions. Another aspect to consider is the feedback provided to the user. In VR, feedback can come in many forms, including visual, haptic, and auditory feedback. Designers must ensure that the feedback is clear and effective, allowing the user to understand the consequences of their actions and providing a sense of agency.

Concerning user input in Virtual Reality, two general approaches in interaction technique design can be distinguished. One direction focuses on naturalism, replicating reality and the interaction it represents as close as possible. The other approach utilizes so-called “magic” interactions[BMR12]. On the one hand "magic" enhances the user’s natural capabilities through the 3D input devices On the other hand "naturalism" focuses on the immersion and engagement of the user. It seems alluring to design 3D tree modeling by providing the user with virtual woodworking tools. But naturalism in VR applications requires great care in design and implementation to have a beneficial impact on usability. A benefit of a naturalistic approach for sketch-based interfaces could be that it mitigates the problem of inferring the purpose of the user’s action. For example, the task of deleting part of a sketch is not trivial without having to choose a different tool as it was illustrated by Tsang et al.[TBSR04] as discussed in Section 2.1. In a naturalistic setting, the user would be provided with a scissor or eraser that can be selected, picked up and by activating the chosen object the connected action is performed. But sketching a tree and manipulating its growth has no practical counterpart so one would need to rely on "magic" interaction anyway. For our work, we chose to focus on “magic” interaction methods but the appeal of naturalistic methods as it was laid out here remains. The possible implications of an alternative naturalistic approach will be discussed in the future work Section 5.3. There are arguments for either approach, but the effect of both naturalism and magic depend heavily on the task and its execution. The manipulation tasks relevant to object manipulation are classified by Bowman et al.[BKLJP04] into a subcategory called spatial rigid object manipulation. This can be specified into 3 tasks that maintain the shape of the object: Selection, Positioning and Rotation. For our work, several decisions were made based on the nature of the task that will be discussed in the chapter "Implementation". During the user study, individual, unstructured feedback was collected concerning the concrete interaction techniques of the user interface.

2.3 Rule-based Systems

The work of Lindenmayer and Prusinkiewicz concerning Lindenmayer Systems (L-Systems) is crucial for the topic of modeling plants [Lin68]. At first glance, L-Systems are just a type of formal grammar. A simple string is rewritten according to a rule set. But they are exemplary for the use case of simulating plant growth. One reason for that can be explained by the process of their discovery. When observing plants, one cannot oversee recurring symmetry and thus beauty in their structure. This has not eluded the eyes of various scientists that researched the characteristics of plants. Pruscinkiewicz et al. tried to formally describe the patterns they found and researched additional factors that influence the growth of vegetation namely developmental algorithms, and self-similarity [PL12]. He collaborated with Lindenmayer who previously had his research focused on the behaviour of plant-based organisms. L-Systems were an effort by Lindenmayer himself to describe the characteristics of plant cells in a formal language. Together with Pruscinkiewicz he discusses the many applications of this seemingly simple idea in their book dedicated to this concept. As mentioned, Lindenmayer developed this new formal grammar while observing the reproduction behaviour of plant cells and as any new cell seemed to change and react individually, he formalized Lindenmayer Systems to capture this organic behaviour. The difference to existing concepts like Chomsky grammars is that in L-Systems the production rules are applied parallel and simultaneously. This proved to be an effective way of representing the complex structures of growing organisms. In the mentioned research, Pruscinkiewicz helped to advance this concept of cell growth to describe whole plants and even render them by interpreting this formal grammar with turtle graphics. Turtle graphics is part of the original Logo programming language, a simple way to draw 2D graphics by giving an imaginary turtle, instructions to move or turn, and optionally draw its path as illustrated in Figure 2.3[AD86]. Pruscinkiewicz used this simple technique in combination with L-Systems to render even complex plants realistically.

The topic of the creation of virtual vegetation has inspired several researchers to come up with ideas for creating plant models that have influenced the decisions in our work. As Niklas states to create realistic-looking plants we need to understand why they grow as they do [Nik86]. As plants do not show behaviour as humans or animals do, the term tropism was created to define the variables that can influence plants. The most prominent of those factors is probably the effect called phototropism which formulates the correlation of light and growth as described by Blaauw [Bla18].

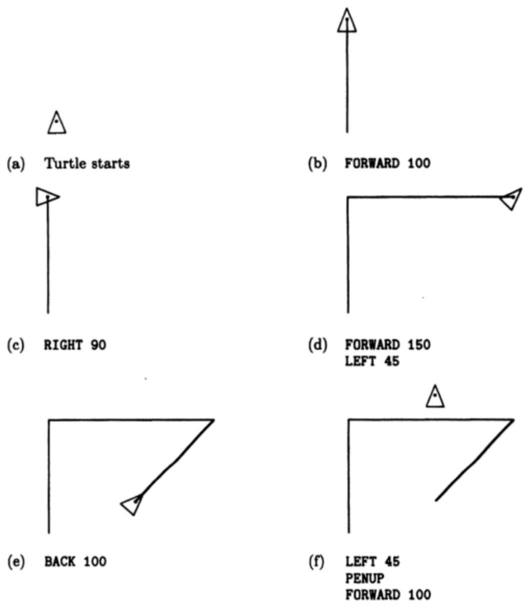


Figure 2.3: A sequence of turtle commands[AD86].

Several approaches have tried to include those findings in the creation of virtual vegetation. Hädrich et al. created a tool to enable users to model realistic climbing plants. In this framework, the influence of several tropisms on the plant can be manipulated by the user by placing attraction points on surfaces to generate growth of climbing plants around structures [HBDP17].

Pirk et al. have developed a fast method to modify a tree so it adapts to its surroundings [PSK+12]. They apply the variables of the environment and calculate how the tree should have grown, according to the forces from the various tropisms, according to the age of the individual branches. This results in a tree that looks like it has been grown in the environment it was placed in. The advantage of this approach is that any tree model can be taken, even very sophisticated ones, and be adapted to different circumstances.

Ochoa demonstrated that tropisms can also be used in combination with L-Systems [Och98]. By using the L-System as genotype in a genetic algorithm and the tropism as a fitness function Ochoa was able to generate convincing plant models stating: "Lindenmayer systems constitute a highly suitable encoding for artificial evolution studies". This supports the assumption that L-Systems are a suitable framework to aid the modeling of plants.

Stava et al. have developed another interesting framework to generate realistic-looking procedural tree models and can even render different types of trees [SPK+14]. They used tropisms and other recent insights from biology to formulate parameters for their procedural model. Existing tree models from a laser scan or another tool like Speed

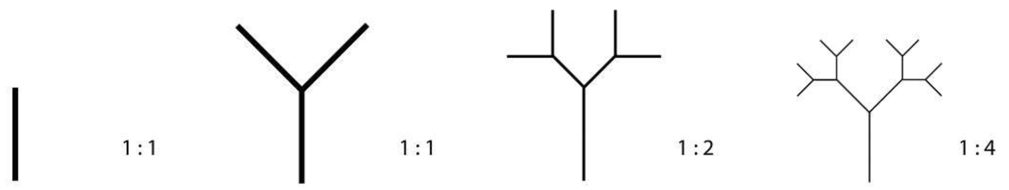


Figure 2.4: Example of a simple L-System

Tree or Xfrog can be fed into their system[spe, xfr]. With a Markov Chain Monte Carlo optimization, they produce a general rule system. By using a similarity function with a maximization algorithm they optimize the parameters for their procedural model to find a fitting description. Guo et al. take a different approach and use deep learning to generate an L-System representation of a 2D input [GJB$^+$20].

On the basis of the related work presented, this thesis proposes an approach that utilizes a sketch-based modeling approach in VR with L-Systems as the underlying procedural or rule-based system. For the implementation of such a prototype, it is clear that the theory behind 3D interaction methods and L-Systems are crucial building blocks. 3D interaction methods are key in enabling users to directly manipulate and shape 3D models in a more intuitive and natural way, while L-Systems provide a powerful rule-based system for generating complex structures from simple instructions.

2.4 Lindenmayer Systems

In this Section, we will discuss the theory and related work of the underlying L-System that was adapted for the prototype to fit the task of modeling three-dimensional tree structures in VR. L-Systems, or Lindenmayer systems, are a type of formal grammar that was introduced by the biologist Aristid Lindenmayer in the 1960s. As Pruscinciewicz states the reason why he was compelled to advance this concept together with Lindenmayer himself, the characteristics of L-Systems that reach in many fields like biology, mathematics and informatics are fascinating for scientists [PL12] . They are commonly used to model the growth and development of complex structures, such as plants, bacteria, and crystals. The idea behind L-Systems is the iterative and parallel application of a set of production rules to a starting symbol, generating a sequence of symbols that represent the structure of the modeled object. This sequence can then be interpreted graphically or geometrically to produce a detailed and visually compelling representation of the original system. One of the most popular methods for interpreting L-Systems is through turtle graphics see Figure 2.3, where the turtle moves and turns on a plane based on the symbols generated by the L-System, resulting in a drawing see Figure 2.4. There are various types of L-Systems, the simplest form is a deterministic and context-free L-System called DOL-System. But they can be adapted towards context-sensitive and parametric L-Systems, which introduce additional rules and parameters to the basic L-System structure. These extensions allow for even more complex and detailed modeling of natural systems. As Pruscinciewicz descibes, L-System consist of a triplet: $G = \{V, w, P\}$ G is the alphabet of symbols, w is the axiom where $w \in V$ and P is the set of production rules where $P = a \to X$ and $a \in V, X \in V*$. a is called the predecessor and X the sucessor. [PL12] The axiom of a L-System consists of a string where each character relates to a drawing instruction for the renderer. A L-System could consist of the string $w = ABABA$ where "A" could mean: “move forward”, and "B": “draw a line” which would then result in a straight line that is two units long. With each iteration, each symbol in the axiom is compared with the predecessors in the ruleset. If any match, the symbol is replaced with the successor and the algorithm continues with the next symbol. If the symbol does not match any predecessors it remains the same. For example, the rules could be:

1. $p_1 : A \to AB$
2. $p_2 : B \to A$

With one iteration the axiom change would be the following:

1. $w_0 : ABAB$
2. $w_1 : ABAABA$

As mentioned turtle graphics can be used to visualize the axiom of an L-System. This is done by defining rules for the turtle on how to interpret each letter of the axiom. Typically turtle graphics are used in two dimensions. For example, the turtle can receive the instructions as shown in Figure 2.3 A crucial part of a realistic tree structure is when and how often branching occurs. One solution could be to just use the data structure of a rooted directed tree to store the information of the L-System. But to keep the simplicity and elegance of this system Lindenmayer himself proposed an alternative solution: Bracketed L-Systems[Lin68]. The idea was to represent an axial tree using strings with brackets. To traverse this tree the symbols can be seen as a stack of instructions. For example the axiom: $w : AB[AB]AB$ would translate to the instructions:

1. "A" : move
2. "B" : draw a line
3. "[" : push state on the stack
4. "]" : pop state from the stack

When this string is interpreted by turtle graphics it would mean that the turtle stores its position and rotation at every [and returns to the previous state at every].

For this work, the basic theory of L-Systems was adapted and elements of bracketed, pseudo, context-sensitive and parametric L-Systems were used. In the next chapter, we will discuss the details and how they influenced the prototype.

CHAPTER 3

Tree Modelling with Sketch-based L-Systems in Virtual Reality

As mentioned in the Related Work Section the focus of this thesis lies on the combination of two approaches towards generating digital tree assets: Interactive modeling and procedural or rule-based systems. In this chapter, we will discuss the proposal of combining sketch-based interfaces as a basis for an interactive tree modeling application with L-Systems as the underlying algorithm as a solution that includes the best of both worlds.

3.1 Sketch-based Modelling

One goal of this work is to evaluate how sketch-based modeling could benefit a tree modeling application in virtual reality based on L-Systems. In this chapter, we will discuss why a sketch-based approach was chosen as a solution for the interactive modeling of tree assets.

Sketch-based modeling origins in the field of Human-Computer Interaction (HCI) and is widely recognized that it can be an essential part of the design process as even highly detailed products start with a sketch. As stated by Olsen et al. hand-drawing is an effective way of communicating ideas in their earliest stages[OSSJ09]. Any conference room is equipped with a whiteboard or flipcharts so speakers can underline their arguments with drawings. There are a lot of arguments from the HCI perspective for sketch-based modeling. Kazmi et al. state it could benefit beginners as the learning curve is not quite as steep as it is with traditional windows, icons, menus, pointers approach (WIMP)[KYZ14]. Current state-of-the-art modeling tools have an abundance of features that allow 3D creators to perfect their models, but it takes a long time to master them. An expert artist knows where to find what tool to perform the exact action they need and can manipulate their objects to the smallest detail. For beginners, this is often overwhelming and in the early stages of an idea might not be the best approach. Sometimes quick and dirty serves the design process better, and this is where sketch-based modeling comes into play. Sketching is less precise than the traditional computer-aided design (CAD) modeling process, but it allows for quick prototypes and iterations.

The research concerning sketch-based modeling covers many fields. One big aspect is overcoming the mentioned imprecision of the sketch-based approach with the aid of computer vision and machine learning. This branch of research focuses on how to infer meaning from the user's sketches. This is especially challenging when the user wants to create a 3D object with 2D sketches. Edge and object detection are exemplary problems for this task. Machine learning can be applied to better detect the user's drawing or learn about the user's inputs and try to predict what the intended action is.

Plants are an organic and ever-changing structure therefore, our work argues that precision in modeling vegetation is not the most important aspect. A sketch-based approach for creating foliage could be ideal as this mitigates a lot of the mentioned issues. By implementing the sketch-based aspect in VR, the 2D to 3D inference problem is solved as the user now has three dimensions for sketching.

Another interesting problem concerning sketch-based modeling is the tool selection. When sketching the user often has a limited choice of tools. A traditional sketch on paper consists of just pencil strokes and nothing else. The more tools are used to create the sketch, the more it turns into a drawing, and the more knowledge is required to complete it. Traditional modeling tools provide a lot of functionality with complex menus. A sketch-based approach would want to avoid that and other solutions needed to be found.

As mentioned computer vision and machine learning are sometimes applied to infer the meaning of the user's actions. But the problem of having to interpret the user's sketch is also due to the input methods used. As Bowman et al. stated: "A virtual reality application may allow a user to place an object anywhere in 3D space, with any orientation—a task for which a 2D mouse is inadequate."[BKLJP04]. The added versatility of user input provided by virtual reality can be used to tackle many of the problems introduced by sketch-based modeling without having to rely on computational expensive algorithms. By implementing 3D interaction methods, the user is enabled to give additional meaning to the sketch. This simplifies the interpretation that is required to understand the user's sketch.

To summarize, the advantages of using sketch-based interfaces for virtual reality tree modeling are numerous. They allow for a more natural and intuitive way of designing tree structures, leveraging the users' existing drawing skills. It enables real-time interaction and feedback, enabling quick and easy adjustments to the model. In the prototype of this work, these benefits are enhanced by the incorporation of a rule-based system. For this work, we adapted an L-System algorithm to take in sketch-based inputs and generate tree models that match the user's intentions. The result is an intuitive and flexible tool that combines the best of both worlds to enable the creation of realistic and intricate tree structures in virtual reality. In the following chapter, we will discuss how the L-System was adapted to fit a sketch-based approach in VR.

3.2 Interactive L-Systems

The core idea of the prototype is to use a sketch of a tree's trunk and branch, created in virtual reality, to generate an L-System that then spawns the desired tree. By combining the immersion of VR and the flexibility of sketch-based modeling the user should be empowered to create the desired tree. The underlying algorithm should serve as a tool to finish the model, without overwhelming the user. In this chapter, we will delve into the details of the L-System algorithm and how it could be applied to the use case of generating complex tree models from 3D sketches. While sketch-based interfaces in VR provide an intuitive and natural way of creating tree structures, the resulting sketches are 3-dimensional in nature. To generate realistic and visually appealing 3D models from these sketches, L-Systems are incorporated as a rule-based system for tree modeling. However, L-Systems require specific rules to generate complex branching patterns and other tree characteristics. Context-aware L-Systems are a possibility to address this. For example, from the users' sketch, the branching-off rate can be inferred. Additionally, parametric L-System rules can be derived from the geometry of the sketches to ensure that the generated tree models have a high degree of similarity to the original sketch. In this chapter, we will discuss the aspects of L-Systems that would allow such an approach.

3.2.1 Context-sensitive L-Systems

Compared to context-sensitive L-Systems, context-free L-Systems suffer from a few drawbacks. One problem of L-Systems, related to the task of generating trees, is that they can quickly go into exponential growth, which does not accurately reflect the growth of real-world trees. This is due to the fact that traditional L-Systems do not take into account the complex relations between the different parts of a system. In the biological example of trees, the growth depends on the interaction between various components such as leaves, branches, and the trunk. To overcome this limitation, alternative approaches such as growth functions or context-sensitive L-Systems have been developed. These approaches allow for a more realistic representation of tree growth. For this work, we focused on a context-sensitive L-Systems approach to solving the tree's growth as it seemed more practical to incorporate the user's input into the ruleset as opposed to creating a growth matrix from the sketch.

Context-sensitive L-Systems are a variation of the classical L-Systems. These systems consider the context of the current symbol in their production rules, leading to the generation of more complex and, if applied right, more realistic structures. For the task of sketch-based tree modeling in VR, context-sensitive L-Systems have several properties that make them ideal for generating tree models from 3D sketches as they can simulate different growth rates across the tree. The syntax of context-sensitive production rules is the following: $p_1 : a_l < a > a_r \rightarrow X$. Meaning that the predecessor a is only matched and replaced by the successor X if all conditions are met.

Now an additional problem is introduced that multiple rules could have similar predecessors. The rules should be applied so that the context-sensitive production rules are applied before the general ones. For example the production rule $p_1 : A < B \rightarrow AA$ would trigger before the rule $p_2 : B \rightarrow AB$ as it is more specific. This was crucial for the implementation in the prototype, as by favoring context-sensitive rules in the generation, the aspects that are influenced by the users' choice, which are of context-sensitive nature, are more prominently resembled in the generated model. One interesting application of context-sensitive production rules is to send signals through a system, similar to how biological organisms operate. As Prusinkiewicz et al. demonstrate, two simple production rules can achieve that: $p_1 : B < A \rightarrow B$ and $p2 : B \rightarrow A$. [PL12] For the axiom $w : BAAAAAAA$ the B symbol would wander one step to the right for each iteration. But context-sensitive production rules in L-Systems not only allow for the transmission of signals through the system, they also provide the means to adjust the growth rate of the generated structures.

The tree trunk's growth can be viewed as a concrete example of the application of context-sensitive L-Systems. One could implement the rule $p_1 : A \rightarrow AA$ to grow the tree, where A stands for a segment of the tree trunk. But this would mean that it would grow exponentially as each A segment would be doubled again in the next iteration. This could be solved by implementing a buffer character B, that replicates instead of the original symbol and a blocker character C which are both skipped by the turtle graphics. By swapping out the initial rule with different rules that state $P : \{p_1 : B \rightarrow BB, p_2 : B < B < B \rightarrow A, p_3 : A > B < A \rightarrow C\}$ a ruleset is achieved that only grows every third iteration and only at the tip of the branch. The iterations would look the following:

1. $w_0 : AB$
2. $w_1 : ABB$
3. $w_2 : ABBB$
4. $w_3 : ABAB$
5. $w_4 : ACABB$
6. $w_5 : ACABBB$
7. $w_6 : ACABAB$
8. $w_7 : ACACABB$

Note that here we can observe the importance of specificity in context-sensitive L-Systems. This example only works because rule p_2 and p_3 supersede rule p_1.

Prusinkiewicz et al. differentiate between 1L-Systems and 2L-Systems[PL12]. 1L-Systems only employ one-sided conditions, while 2L-Systems use two-sided conditions. The use of one-sided conditions in 1L-Systems could become complicated, as the mixing of the $>$ and $<$ symbols could lead to confusion. In contrast, 2L-Systems allow for greater specificity in defining the conditions for rule application, but are also more constricting in their definition. Therefore, for the context of this thesis 2L-Systems will be used where any side of the condition can be left empty, meaning that the context on this side does not matter to the production rule. As it was laid out in this Section, context-sensitive L-Systems provide a lot of versatility towards generating more sophisticated structure as they can consider the surrounding of each symbol. But they still generate rather stiff structures that often lack characteristics we can observe in nature. This is where parametric L-Systems come into play.

3.2.2 Parametric L-systems

Parametric L-Systems offer a solution to the mentioned shortcomings of context-sensitive L-Systems and allow the generation of a lot more versatile structures. Traditional L-Systems are limited when it comes to the interpretation via turtle graphics, as the symbols only allow the use of a unit length. The symbol A might be interpreted by the turtle to move X in the forward direction. But if a different length is required for example to close a triangle with a right angle it would need to add a new symbol for each required distance. Additionally, the rules to fulfil these requirements would get very complex very quickly. This was solved by introducing parametric L-Systems [Pru90] that add parameters next to the symbols. Parametric L-Systems are another variant of L-Systems that allow for the interpretation of symbols using parameters. Unlike context-free L-Systems, which use a fixed set of rules to interpret symbols, parametric L-Systems use a set of parameters to define the interpretation of symbols. These parameters can define the interpretation of the symbols towards the size, shape, and orientation of the generated model, as well as the application of the production rules. By defining the interpretation of symbols using parameters, parametric L-Systems enable the creation of detailed and intricate structures, such as plants and trees, that might not have a one-length unit line as their basis. The parameters can be referenced in the production rules by introducing conditions. When Σ is the set of parameters and $C(\Sigma)$ is a logical expression using the parameters, then the production rule P with the successors and predecessors $v \in V$ where V is the alphabet of the L-System, can be expanded with the condition like so: $v_0(\Sigma) : C(\Sigma) \rightarrow v_1(\Sigma)$. This is best demonstrated with the same use-case of trunk growth rate from Section 3.2.1. The starting axiom $w : AB(1)$ can serve for this example. The integers in the brackets are the parameters. The rules for an example of non-exponential growth could now be: $p_1 : B(x) : x < 3 \rightarrow B(x+1)$ and $p_2 : B(x) : x >= 3 \rightarrow AB(1)$ which would result in the trunk growing only every 3rd iteration.

3.2.3 Parametric 2L-Systems

To complete the function set of L-Systems that were used for this work, parametric L-Systems can be combined with 2L-Systems into so called 2OL-Systems. 2L-Systems are context-sensitive L-Systems with two-sided conditions as discussed in Section 3.2.1. The parametric definition of rules and axiom can now be extended with before and after conditions. Combining context-sensitive with parametric L-Systems in this way opens up a variety of new possibilities. A parametric 2L-System is now able to have context-sensitive rules, that consider the context before and after each symbol in the axiom. It can define parameters that can be referenced by the graphical interpreter, for example, turtle graphics, to produce more complex shapes with non-unit length sections. And finally, the parameters can also be referenced as a condition for the succesor, resulting in more complex rewriting rules. As an example the production rules $p_1 : B(x) \rightarrow B(x+1)$ and $p_2 :< A(x) < B(y) : y >= 3 \rightarrow A(x * 0.5)B(1)$ for the axiom $w : A(1)B(1)$ can be introduced. The iteration over the axiom would look like this:

1. $w_1 : A(1)B(2)$
2. $w_2 : A(1)B(3)$
3. $w_3 : A(1)A(0.5)B(1)$
4. $w_4 : A(1)A(0.5)B(2)$
5. $w_5 : A(1)A(0.5)B(3)$
6. $w_6 : A(1)A(0.5)A(0.25)B(1)$

In this definition, the character A could represent a branch section and the context-sensitive rule p_2 will only add another branch segment when the A character is at the final position. This is assured by the context-sensitive rule that it has to be followed by a B character. Parameters are used to increment the y and divide the x variable. The x variable can be used by turtle graphics to draw non-unit lengths, for example with the instruction: *"A(x)": pendown, move x, penup.* Additionally the parameters are used to ensure that the tree does not grow exponentially by checking the condition: $B(y) : y >= 3$ before adding another A segment.

Note that we were able to achieve the growth behaviour of growing only at the tip of every third generation with all three types of specialised L-Systems we discussed here. But parametric L-Systems needed one less rule to achieve it and, 2OL-System also needed 1 less rule, but also added a way to add additional context to the axiom string. Using 2 OL-Systems proved to be the most concise way of describing the applied production rules of the L-System used for this prototype.

3.2.4 Node rewriting

For completeness, one final aspect of L-Systems should be mentioned at this point. The operation mode of all the examples of turtle graphic interpretation of L-Systems presented so far, can be characterized as "edge-rewriting" as defined by Prusinkiewicz et al [Pru90]. But there exists the possibility of simplifying the rules of L-Systems by implementing node-rewriting. This means that symbols can be replaced with certain structures or blueprints that are defined beforehand. This way some complexity can be kept out of the string of the L-System. In praxis, the desired model might contain some recurring but complex structure. By encapsulating the complex object into a "node" and defining a way in and out for the turtle graphics, the complexity of the structure can be kept out of the definition of the L-System. In the Section3.3, we will discuss how some aspects of node rewriting were applied in the prototype to refer back to the user's sketch.

3.3 Implementation

As discussed in section 3.1 this work argues that sketch-based tree modeling can offer several advantages over traditional modeling techniques. Additionally, implementing this technique in a virtual reality environment could enhance the user experience by providing a more immersive and intuitive interface. The use of L-Systems for generating the tree structures also could offer practical benefits, such as the ability to create complex branching patterns and simulate realistic growth processes. To evaluate such an approach a virtual reality prototype was developed for this work. The goals of the prototype and the following user study were to investigate the usability of L-Systems and sketch-based modeling in the context of creating tree models. In this section, the VR prototype developed for this work is discussed. The goal of this tool is to provide users with an immersive interface for creating and manipulating virtual trees using simple hand-drawn sketches. By leveraging the benefits of VR technology and L-Systems, users should be able to easily generate diverse and realistic tree structures. The decisions that influenced the design of the prototype will be explored in this section. Through this exploration, we hope to provide a better understanding of the capabilities and limitations of the prototype.

3.3.1 Protoype

Before we discuss the usage of the prototype and how users can sketch the desired tree we will discuss the terminology to get a common understanding of the sketching process. The sketched structure can be described as a rooted tree. For the terminology of this section we will follow the example of Prusinciewicz et al [PL12]. Sketching of a tree in the VR prototype of this work begins at the root or base node, while a node located at the top of the tree is referred to as a terminal node. The nodes are connected by edges, which are called branch segments in the biological context. The segments connecting two successive nodes are called internodes, with the exception of the final segments, which are referred to as apex.

3.3.2 Features

The user starts sketching in the prototype by drawing from the root node with the use of a VR controller. Users can draw lines in the air which are then connected with same-width lines resembling the branch segments, forming the initial structure of the tree. These nodes can be dragged around or deleted to modify the internodes and the tree's shape. A second line can be added to any internode, to create a branch and achieve the desired branching structure. The drawn sketch is located on a platform floating in front of the user. The platform can be dragged around and positioned by the player for a comfortable drawing angle. At any stage of the sketch, the user can render the tree and see how it grows, based on their sketch. The actual tree is located behind the floating structure. Any changes made to the sketch will be reflected on the next rendering step of said tree, giving the user immediate feedback on their design choices. The user can cycle through the stages of the tree, allowing them to see how the sketch affected its growth and refine their design as they go. The rendered tree is based on the sketch, and replicates the sketch's structure using an underlying L-System that is derived from the geometrical drawing created by the user. This results in a three-dimensional tree with self-similar elements derived from the sketch. The branches' orientation can be controlled with an additional lever placed on the floating sketching platform. The lever will change the incremental offset of the branches. An option to scrap the whole sketch and start fresh was added as well.

3.3.3 Usability Design

The interaction design was influenced by the developer guide of oculus quest [ocD] as well as the architecture constraints of the XR interaction kit of Unity. As it is suggested by the XR interaction toolkit, the user interacts with the world by pointing at objects and selecting them. The interaction methods are, as proposed by the oculus quest developer's handbook, aimed to resemble the standards the users are accustomed to and are mainly done by using the trigger buttons. When pressing the triggers, there are two specialized actions the user can perform when hovering over an object with the x-ray originating from the controller: Selection and Activation. Selecting is done by pressing the grip trigger on the side of the controller. There are two ways to activate an object. Either first, a selection is done by pressing the according button, whilst pointing at the object. Then the object counts as selected and additional actions can be taken by activating it. Or one could activate an object directly by pressing and holding the selection button. The application is designed in a way that any action can be done with either controller. This way left- or right-handed users should have the same experience. As mentioned an activation is performed while an object is selected, which is done with the main trigger on the back of the controller. This enables the development of multiple interaction methods with no distinction necessary between the left and right controllers. The user can draw the trunk of the tree by pressing the main trigger of the controller, meaning they activate the apex node of the tree, adding another branch segment to the top of the tree. When confirmed, a new apex section of the trunk, consisting of two internodes and a line connecting them, the branch segment, is added at the controller's position. When the user has confirmed at least two trunk sections, a new branch can be added by selecting a previous node. The internodes can be moved by selecting and dragging them or be erased by pressing the trigger again and "activating" them. A line renderer is used to indicate where the next branch section would be drawn should the user confirm the input.

As discussed in the related work section it was demonstrated that versatile input methods and 3D interaction techniques can improve the sketching process. Virtual Reality is a powerful technique for interactive modeling. The new input methods that are enabled by recent technology for VR including hand-held controllers, haptic feedback gloves, and motion capture sensors, allow users to interact with virtual objects in a more natural and intuitive way, enhancing the immersive experience. One of the key advantages of utilizing

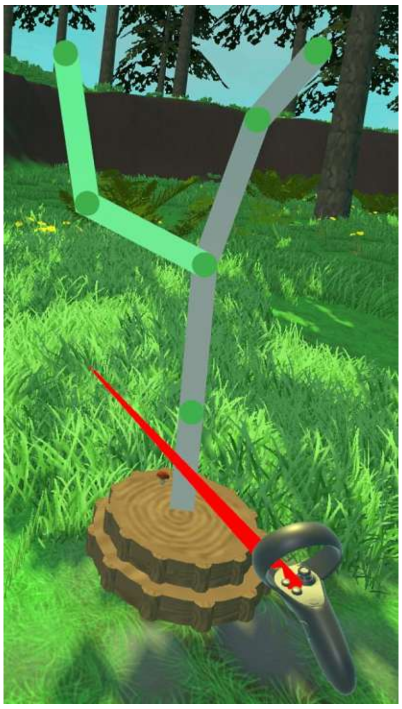


Figure 3.1: Sketching platform of VR Prototype

virtual reality for sketch-based tree modeling is the immersive experience it provides. By putting on a VR headset, users can enter a virtual environment where they can move around and interact with the model in a more natural and intuitive way. This allows them to gain a better understanding of the scale and proportion of the tree, and to make more accurate and informed design decisions. Additionally, the virtual environment can be designed to be visually appealing and soothing, creating a calming atmosphere that can reduce stress and improve focus. This can be particularly beneficial for professionals who spend long hours working on tree models, as it can help to prevent eye strain and mental fatigue. Being set in virtual proximity to the model can save time and resources by eliminating the need for physical prototypes or multiple design iterations.

Overall, virtual reality provides a powerful tool for tree modeling that can improve accuracy, efficiency, and user experience. But as discussed in Section 2.2 designing for interaction in virtual reality requires a nuanced approach by taking into account the unique affordances and constraints of the medium, as well as the needs and limitations of the users. But thankfully the wheel does not have to be reinvented. As Bowman et al.[BKL04] lay out, manipulation techniques for virtual reality can be categorized into metaphors see Figure 3.2. In this section, we will discuss the metaphors used for the prototype and the decisions behind them.

As mentioned the sketching implementation uses just the position of the controller. But the selection is done with the ray casting technique. The interaction between the user

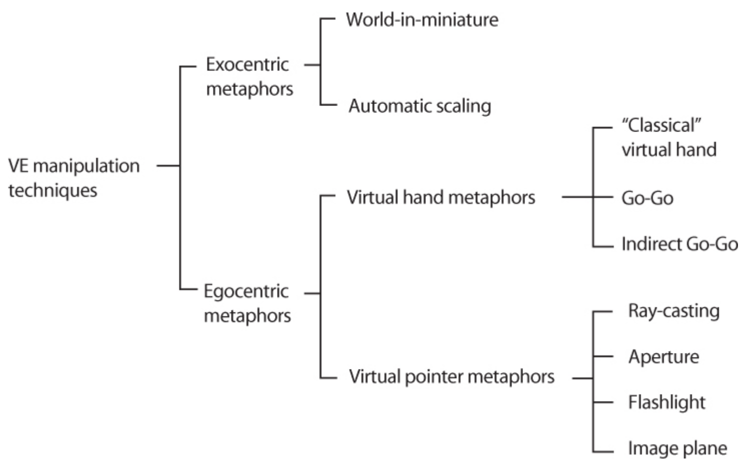


Figure 3.2: Virtual Environment Interaction Metaphors by Bowman et al. [BKL04]

and the nodes is of a sketch-based nature and as discussed in Section 3.1, precision is not the most important aspect for this type of application. Therefore, the simple point with ray-cast interaction was chosen. Once selected the node "flies" to the user's controller and sticks to it, until the trigger is let go.

When the user is face to face with a fully grown tree in virtual reality, this can result in scaling problems. To tackle the scaling issue, a sketching platform was used that provides a smaller blueprint version of the final tree that can be dragged around and manipulated, see Figure 3.1. This approach was inspired by existing manipulation methods for virtual reality similar to the "world in miniature"[SCP95] technique see Figure 3.2.

Movement is a complex issue in virtual reality applications, as it can have a significant impact on the user's sense of immersion and comfort. Teleportation methods are a common solution to this issue, as they allow users to quickly move around virtual spaces without inducing motion sickness or discomfort. However, these methods can also be disorienting and may disrupt the user's sense of spatial awareness. For the prototype, teleportation methods were avoided and instead the users are provided with a movable sketching platform that can be placed in various locations to minimize the amount of physical movement required. This approach allows for maintaining the user's sense of immersion while also reducing the potential for discomfort and disorientation.

3.3.4 L-System

In the previous section, the interaction between the user and the sketch was explored, as well as the use of 3D interaction techniques to enhance this interaction was discussed including the implications of these techniques for virtual reality applications, where the user's sense of presence and immersion is crucial. Moving forward, we will discuss how the geometry of the created sketch impacts the rendered tree in our prototype.

As stated part of the goal of this thesis was to evaluate the usability of L-Systems as a base algorithm for a sketch-based tree modeling approach. Since sketch-based tree modeling involves an interactive process, any implementation using L-systems as a base needs to be interactive as well. In section 3.2 we discussed the more intricate elements of L-Systems and in this section, we will lay out how those aspects were implemented to allow for an interactive L-System influenced by the user. We will divide the task of sketching and rendering a tree into several subtasks and discuss the implications on the prototype as well as the underlying L-System.

The first step of sketching a tree is the creation of the trunk. Storing the sketch in the L-System and rendering the desired shape is relatively straightforward as long as the number of iteration steps performed on the L-System is less than the number of sections of the trunk the user has drawn. For the tree to "grow" into the drawn shape, the user will have to step through each stage by triggering the growth action. In the context of L-Systems, we will call the growth cycles iterations. For the first iteration, the axiom would just consist of the character "A". The trunk consists of at least one branch segment, from the root of the tree to its apex. This is stored in a dictionary by saving the position of each node of the sketched trunk to an identifier, that will be referred to by the L-Systems alphabet. For the prototype, the character "A" was chosen as an identifier for the trunk segments. This storage of the sketch's geometry is already a small deviation from the standard L-Systems. As discussed in Section 3.2.2 parametric L-Systems can be used to draw non-unit size segments with the turtle graphics. But as the lengths of each segment in our prototype do not follow any predefined rules, but are sketched by the user, this approach is inapplicable. Therefore, we borrow elements from the concept of node rewriting as discussed in section 3.2.4 by storing positional information defined by the user as "blueprints" for the turtle graphics in the dictionary. To represent the growth of the trunk we will introduce the first rule to the L-System. To make sure the L-System does not grow exponentially we will introduce a context-sensitive rule as discussed in Section 3.2.1. For this, to work we will have to introduce a second character: "B". This character acts as a buffer and marks the end of the trunk. With this approach, we can make sure new segments are only added to the tip of the tree. In the notation from context-sensitive L-Systems this rule would look like this: $> A > B \rightarrow AA$. Note that the first element of the selector is empty, as we do not care what comes before the A. We only care that A is followed by B, therefore being at the tip of the trunk so we replace it with "AA". The axiom "AB" would then iterate the following:

1. AAB
2. AAAB
3. A[...]AB

After n iterations, the axiom of the L-System representing the sketched tree would now be n-times the character "A". This is simple enough as long as n is smaller than the size of the list of positions stored in the dictionary for the character "A". The axiom of the L-System representing the full trees sketch would now be n-times the character "A" followed by the final "B", where n is the number of branch segments the user has drawn for the trunk. To render the tree for each "A", the renderer just has to iterate over the positions of the users' sketch. This is done by looking up the list of nodes in the dictionary for the character "A". For the first iteration, the renderer would look for the first character in the mentioned dictionary, and draw the first branch segment from the root node to the relative position stored in the dictionary at position 1. But we want the renderer to be stateless, so it does not have to keep track of the number of iterations. To store the positional information in the L-System we can utilize parameterized L-System. We would need to adapt our initial context-sensitive generation rule to include parameters. The mix of parameterized and context-sensitive L-Systems is called Parametric 2L-Systems as discussed in Section 3.2.3. To include parameters the initial context-sensitive rule could be changed like this:

$$< A(x) < B \rightarrow A(x)A(x+1)$$

With the changed initial axiom of: $A(0)B$ the iterations would now look the following:

1. A(0)A(1)B

2. A(0)A(1)A(2)B

3. A(0)[...]A(n)B

To render this axiom, turtle graphics was adapted for this prototype to work in three dimensions, similar as described by Prusinciewicz et al. [PL12] with the difference of using quaternions, as they are generally used in Unity3D for rotations. The turtle renderer iterates through the axiom, and when encountering an $A(x)$ symbol, moves to the next node by looking up its position in the dictionary under the key "A" at position x of the list of nodes that are stored there and then draws a line connecting them. But once n grows larger than the trunk sections drawn by the user, we would run into an out-of-range error as there are not that many positions stored in the array. Thankfully the self-similar attributes of plant-like structures allow us to just repeat the structure from the base of the trunk again. Observations in the user study have shown that repeating the trunk structure will still result in the overall desired shape most of the time. This is solved in the implementation by using the modulo operation with the size of the array stored in the dictionary. The rule defined for the turtle graphics to render the trunk in pseudo-code would be:

Algorithm 3.1: Turtle graphics pseudo code.

```
Data: Dictionary<Char,List<Transform> > dictionary, int iteration
switch $c$ do
    case "A" do
        transform = dictionary[c][iteration mod dictionary[c].Size];
        position = transform.position;
        rotation = transform.rotation;
        drawSegment(position,rotation);
    end
end
```

Now let us say the user wants to add a branch to the trunk. The geometry of the branch itself can be stored easily, by just introducing another character, for example, "C" and adding an entry for it in the dictionary. Now the positions can be looked up in a similar manner to the trunk. For the branch growth we would just need to add a similar rule as we have for the trunk but with the new character:

$$< C(x) < B \rightarrow C(x)C(x+1)$$

But the problem remains when and where the renderer should apply this branch to the tree. As discussed in Section 2.4 brackets can be used in L-System to implement branching patterns. This would mean that the turtle renderer saves its position and rotation when a [bracket in the L-System occurs, and returns to it once it reads a closing bracket]. For this prototype, it had to be solved how the users' choice of where the branch was added can influence this mechanism. By adding one more rule we can branch off exactly where the user indicated it on the sketch. Depending on what node the user chose to branch off from, for example in case the user chose the node after the third branch segment: node A(n=3). The rule for the generation of the L-System and adding the branch at the desired position would look like this:

$$< A(x) < A(x+1) : x == n \rightarrow A(x)[C(0)B]A(x+1)$$

But this would generate only one branch for the whole tree. For this prototype, the approach was chosen to repeat the branch at the same interval where the user chose to branch off initially. If the user as in our example chooses the node at position n=3 to add a branch, the branch will be readded every 3 segments of the trunk. To implement this the rule just needs a small adjustment by utilizing the modulo operation:

$$< A(x) < A(x+1) : x \mod n == 0 \rightarrow A(x)[C(0)B]A(x+1)$$

The final adjustment for the underlying L-System is to allow sub-branching. Again the approach was chosen to use the users' initial choice of where to branch off, as a

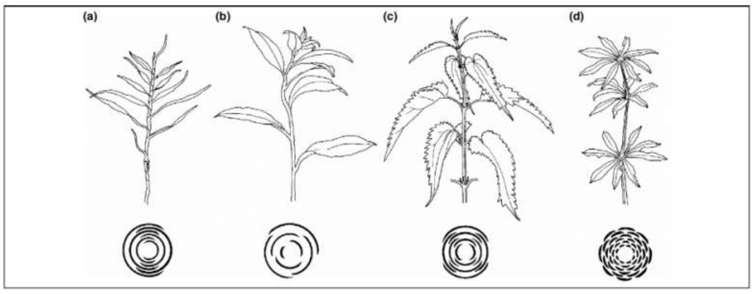


Figure 3.3: Phyllotactic patterns as demonstrade by Kuhlemeier et al [Kuh07]

sub-branching rate. If we look at the example again with the chosen branch off node n=3, it would mean that the branch would create subbranches at every 3rd branch segment and those subbranches would branch off after 3 branch segments as well. This behaviour can be encapsulated by reusing the initial branching off rule for the characters referring to the branch:

$$< C(x) < C(x+1) : x == n \rightarrow C(x)[C(0)B]C(x+1)$$

Now the only problem remaining was that the branches would always be oriented in the same direction. This would look weird as plants, as we can observe them in nature, would not have their branches all reach in the same direction. This behaviour of plants, arranging the lateral organs is called "phyllotaxis". Phyllotaxis refers to the arrangement of leaves, flowers, and other plant structures on a stem or branch. There are two main purposes of phyllotaxis in plants: Firstly space efficiency: One purpose of phyllotaxis is to allow plants to efficiently use the limited space available to them. By arranging leaves and other structures in a specific pattern, plants can avoid overcrowding and maximize their use of available resources such as water and nutrients. This helps plants to grow and reproduce effectively in their environments. Secondly, optimization of sunlight absorption. One purpose of phyllotaxis is to optimize the absorption of sunlight for photosynthesis. By arranging leaves and other plant structures in a specific pattern, plants can maximize their exposure to sunlight while minimizing shading between neighbouring leaves. This allows for efficient use of available light energy, which is essential for plant growth and survival. There are different strategies to solve this as illustrated in Figure 3.3 by Kuhlemeier et al [Kuh07]. One possibility is to alternate the branches, left and right by 180°, similar to ferns. When observing phyllotaxis in nature, one interestingly finds a reoccurring number. As demonstrated by the last pattern (d) in the illustration one of the most efficient layouts can be achieved by using a divergent angle of approximately 137.6°. This number can be calculated by applying the golden ratio to the 360° of a full circle. This could be one explanation for why the golden ratio can be found so often in fauna and flora. Thus as an initial diverging branch angle for this prototype, the golden angle was chosen. But, if desired, the user can change this with the lever on the floating platform. The usage of quaternions for the turtle graphics orientation benefits this approach as this angle can just be applied to the turtle before it branches off. The initial orientation of the turtle is calculated by transforming it so that it looks towards the next node stored in the dictionary. Now the quaternion with a rotation of 137.6° can be applied on the turtle's forward axis and the desired diverging branch angles are achieved.

In summary, the implementation of a sketch-based tree modeling prototype using L-Systems in VR has been a challenging yet rewarding process. The adapted L-System algorithm provided a flexible and efficient framework for generating complex tree structures from simple sketches, and the virtual reality environment enabled a more immersive and intuitive user experience. As discussed in this chapter, throughout the development process, various design decisions were made to balance functionality, performance, and usability, and these decisions were informed by the existing literature and user feedback. In the next chapter, the results and analysis of the user study are presented, which aimed to evaluate the effectiveness and user experience of the prototype.

CHAPTER 4

Results

4.1 User Study

To investigate the research question of this thesis: "How does a sketch-based, procedural modeling approach using L-Systems, coupled with immersion in Virtual Reality, influence the process of modeling a tree?" a within-subject user study was conducted. In the study, users had to complete a task under two conditions. In the first condition, they had to model a given tree in the industry standard software for tree modeling "SpeedTree" [spe]. In the second condition, they had to generate the same tree in the prototype developed for this thesis. The tool SpeedTree has a very complex user interface. It is an example of the classic windows, icons, menus, pointers approach (WIMP)[KYZ14] for applications. As described in the getting started section of the documentation for speed tree one starts modeling a tree by defining and chaining together several generators, see Figure 4.2. Those can be adjusted by changing the parameters of the generator. But the parameters can get complex and getting the desired output can take a long time, especially for inexperienced users. After the initial setup of the generators, one can also customize the branches by dragging around the nodes, or even by hand-drawing them. In comparison, the VR application developed for this work focuses on the drawing aspect and tries to infer the parameters from the user's actions.

To compare the usability of the two applications the participants were given this task and a rough time limit that they should loosely stick to. The task was to model the trunk and branches of an example tree provided as an image, see Figure 4.3 . The example tree given is a japanese maple tree that has beautiful branching patterns. This should make it interesting as well as challenging to recreate said tree. The participants were provided with a short tutorial for both tools so they could focus on the task quickly. During the SpeedTree as well as the prototype attempt, the participants could ask questions and the study supervisor tried to aid them as long as they were purely technical questions. After both attempts they filled out a standard usability questionnaire: the System Usability

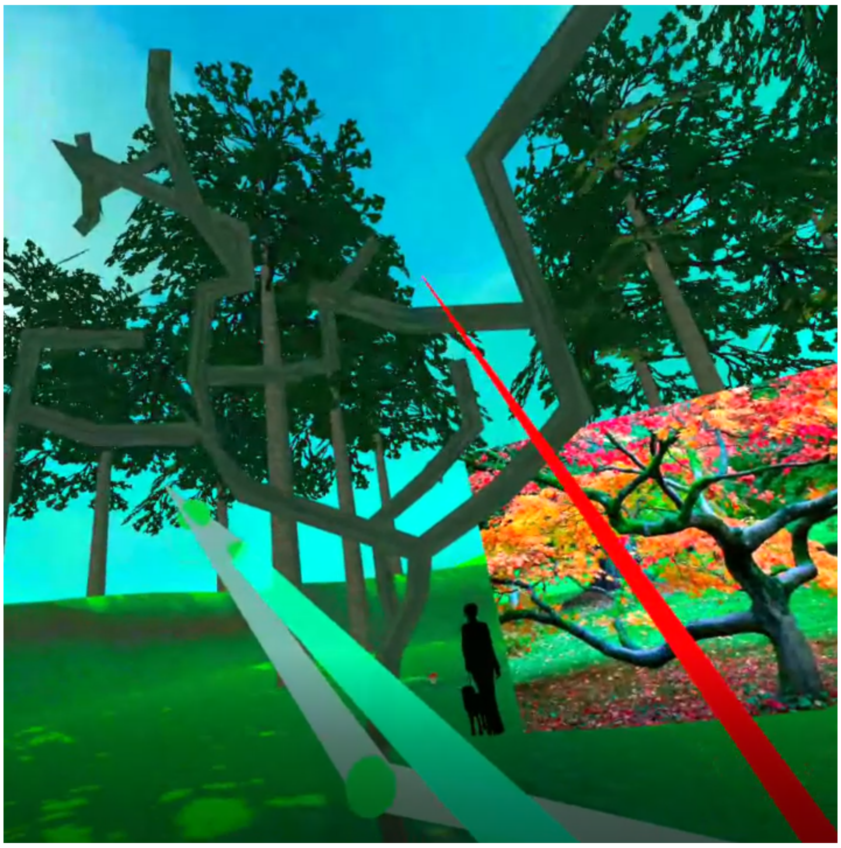


Figure 4.1: Attempt of a participant to recreate a maple tree from a reference image.

Scale (SUS)[B+96], and answered five custom qualitative questions related to the task. By including the SpeedTree attempt we have a baseline for modeling a tree with a standard industry tool and compare this to the prototype of our work. To monitor the effects of the VR application the participants also had to fill out the Simulator Sickness Questionnaire(SSQ)[KLBL93]. Additionally, all participants filled out a pre-screening to get insights about the study demographics as well as influencing factors like previous experience or physical barriers. All the participants signed a consent form which can be found in the appendix.

4.2 Study Results

The study participants include 7 people, 2 females and 5 males. The participants filled out the SUS both for the prototype of our work and the standard industry tool SpeedTree. The SUS was evaluated according to Brooke [B+96] and results in a number between 0 and 100 where 68 would be the average score of systems that have been tested by this scale. So if a system scores above 68 it should score above the 50% quantile compared to other systems. Additionally, the results were analysed with the Wilcoxon Signed-Rank

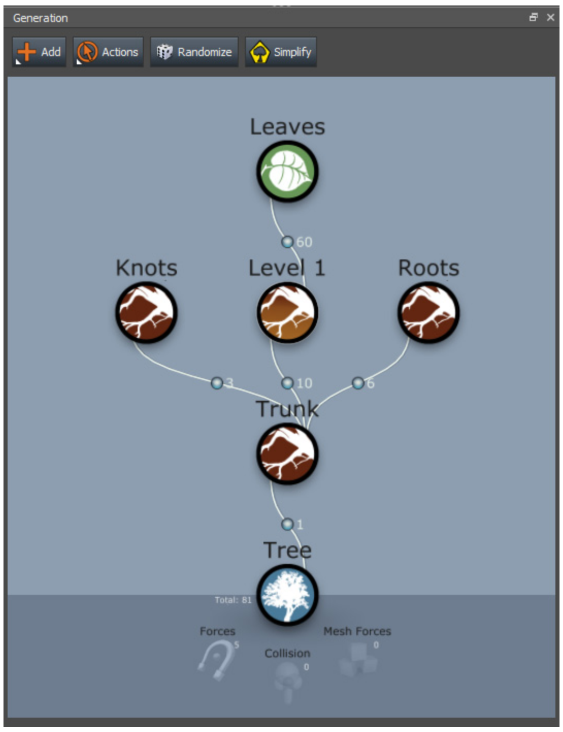


Figure 4.2: The generator window of the SpeedTree application. [spe]

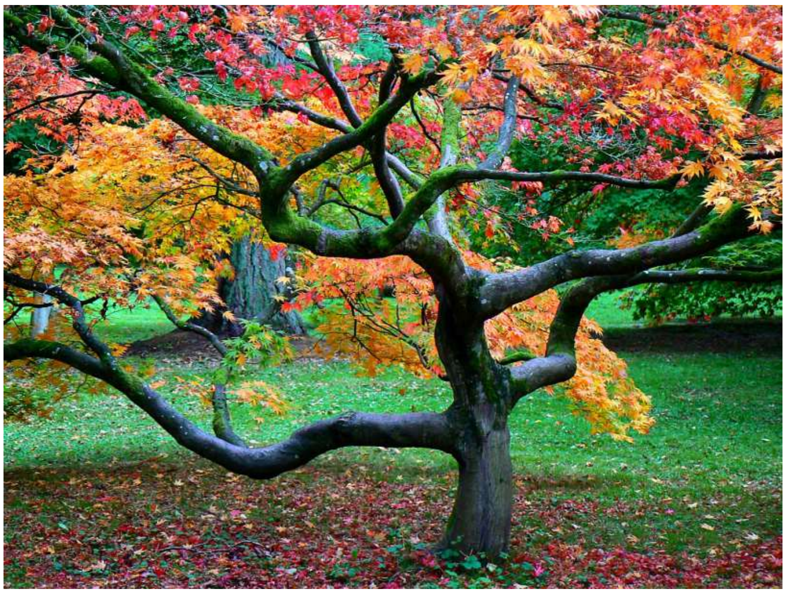


Figure 4.3: An image of a japanese Maple Tree from [Bri10], provided to participants as reference.

test as it is well suited for a within-participants study [Con99]. As suggested by Bimberg et al. the Simulator Sickness Questionnaire was conducted before and after the VR part of the study [BWK20]. The answers from before the test can then be subtracted from the final result to remove pre-existing circumstances that might have nothing to do with the effects of the prototype.

4.2.1 System Usability Scale

The SUS (System Usability Scale) was evaluated according to Brooke[B+96]. Every Question had a scale with 5 options from 1 (Strongly Disagree) to 5 (Strongly Agree). To get a total score of 1-100, where 68 would be the average score of all applications, one adds all uneven questions but shifts the score by -1. Meaning the answer to Question 1 "Strongly Agree" would be scored as a 4. And "Strongly Disagree would be scored as a zero. The sum is then added to the sum of all even questions, where each individual score is subtracted from 5. Meaning on Question 2 the answer "Strongly Agree"(5) would be scored as 0(5-5) and "Strongly Disagree"(1) as a 4(5-1). The added sums that would now have a possible maximum score of 40 are then multiplied by 2.5 to transform the score in the range of 0-100 for readability.

The average score from the participants of this study for SpeedTree was 62.50 which was considerably lower than the overall score for the VR application which was 86.07. The Wilcoxon Signed Rank test with a confidence interval of 95% suggests that there is a significant difference between the scores of both applications with a Z-value of -2.117 and a p-value of 0.047. The mean of all the scores for the Speed Tree evaluation was under the mentioned average of 68 and is a lot lower than the score of the prototype of this work. For most questions, when analysed individually, the results did not indicate a significant difference between the usage of the SpeedTree tool and the prototype of this thesis in the specific aspect of that question. Nonetheless, two questions, 7 and 10, stuck out with relatively clear results and several results showed some interesting trends. The other questions of the SUS did not show any significant differences in the usage of the two applications in the specific aspect the questions targeted. Below we will now discuss the mentioned surbey questions that showed interesting trends.
Question 2 - "I found the system unnecessarily complex." All the participants answered that they disagree or strongly disagree in favour of the VR prototype in the question: "I found the system unnecessarily complex.". It might suggest that the complexity of the prototype was well chosen.
Question 7 - "I would imagine that most people would learn to use this system very quickly." For the question: "I would imagine that most people would learn to use this system very quickly" all but one participant answered this question more positively for the VR prototype. This suggests that the learning curve for the tool SpeedTree is indeed quite high and might not be suitable for beginners.
Question 9 - "I felt very confident using the system." No one chose the option disagree or strongly disagree for the statement "I felt very confident using the system." for the VR prototype. This might suggest that the participants felt more confident compared

SSQ	Before	After	Difference
Nausea	17.72	12.27	-5.45
Oculumotor disturbance	9.75	10.83	1.08
Disorientation	0.00	17.9	17.9
Total	11.75	14.96	3.21

Table 4.1: Results of the Simulator Sickness Questionaire
negligible (< 5), minimal ($5 - 10$), significant ($10 - 15$), concerning ($15 - 20$), bad(>20)

to the SpeedTree tool.
Question 10 - "I needed to learn a lot of things before I could get going with this system." Question 10 in the System Usability Scale was answered more positively for the VR prototype by all the participants. This question falls into a similar category as question 7 and would suggest that the entry barrier for new users is lower for our prototype than for the tool SpeedTree.

4.2.2 Simulator Sickness Questionaire

The Simulator Sickness Questionaire(SSQ)[KLBL93] was conducted to observe any physical effects the virtual prototype might have on the test subjects according to Kennedy et al. The SSQ was originally developed for military flight simulators but was widely adopted to quantify physical effects like simulator sickness, or cyber sickness in VR applications. The test subjects have to answer for a range of symptoms and how strongly they experience them. The answer possibilities range from None(0), over Slight(1) and Moderate(2) to Severe(3). Additionally to an overall score, there are subcategories for related symptoms: nausea (N), oculomotor disturbance (O) and disorientation (D) as visualized in Figure 4.4 by Bimberg et al. [BWK20]. Each category is then multiplied by a constant factor that normalizes the score so they can be compared. The total scores can be interpreted by using the following categories: negligible (< 5), minimal ($5 - 10$), significant ($10 - 15$), and concerning ($15 - 20$) symptoms. Above 20 is considered "bad" [SKD97].

The time the participants spent wearing the HMD during the study was limited to 20 minutes as it is recommended in many health guidelines to keep VR usage times under 30 minutes per session. The SSQ was taken before and after the participants completed the task in the VR application. When we take a look at the results of the test taken after the session we can find significant effects that were experienced by the participants. For nausea the score of 12.27 was relatively high which would mean an overall significant perception of the symptoms. The symptoms related to oculomotor disturbance were perceived as minimal with a score of 10.83 on average. With a score of 17.9, Disorientation was perceived more widely, falling into the concerning category. The total score of 14.96 also falls in the category of significant symptoms. But as Bimberg et al. [BWK20] point out, those scores do not factor in the symptoms the participants might have had beforehand. Therefore, for this user study, an additional SSQ was filled

	Weight		
SSQ Symptom	N	O	D
General discomfort	1	1	
Fatigue		1	
Headache		1	
Eyestrain		1	
Difficulty focusing		1	1
Increased salivation	1		
Sweating	1		
Nausea	1		1
Difficulty concentrating	1	1	
Fullness of head			1
Blurred vision		1	1
Dizzy (eyes open)			1
Dizzy (eyes closed)			1
Vertigo			1
Stomach awareness	1		
Burping	1		
Total	[1]	[2]	[3]

$$\text{N} = [1] \times 9.54$$
$$\text{O} = [2] \times 7.58$$
$$\text{D} = [3] \times 13.92$$
$$\text{TS} = ([1] + [2] + [3]) \times 3.74$$

Figure 4.4: Evaluation of the Simulator Sickness Questionaire

out before the VR attempt. If we subtract the initial scores from the final result, we may get a more complete picture of the actual effect the application had. The difference between before and after the attempt can be seen in Figure 4.1. Subtracting the scores from before the attempt does change the resulting categories except for the symptoms related to disorientation. The new calculated categories are nausea -5.45 (negligible), oculomotor disturbance 1.08 (negligible), disorientation 17.9(concerning), and total 3.21 (negligible). For nausea, some participants scored the related symptoms even lower than before the test, thus resulting in a negative score of -5.45, but of course, we can not assume that this positive effect was caused by the prototype.

CHAPTER 5

Conclusion

5.1 Summary

The purpose of the master thesis was to explore the use of sketch-based, procedural modeling with L-Systems in Virtual Reality for tree modeling. A user study was conducted with a small number of participants to examine the impact of this approach on the modeling process. The results of the study showed that the sketch-based, procedural modeling approach using L-Systems in Virtual Reality had a positive influence on the tree modeling process. However, due to the small sample size of the study, the findings should be interpreted with caution and further research is needed to confirm the results.

5.2 Limitations

5.2.1 Prototype

The scope of the prototype was chosen to only enable the user to draw the trunk and one branch to reduce complexity. This served two purposes. First, it should ease the user into to concept of creating a shape based on an L-System. By keeping the initial L-System simpler, the resulting tree should be more easily recognized to be based on the user's sketch. In this way, they should be able to see the patterns faster. Adding only a single branch reduces the possibilities, and could be implemented in the future for more sophisticated branching structures. Secondly, for testability we decided to keep the feature as simple as possible, to make clearer deductions from the user study. With too many possibilities for the user, it would be hard to narrow down what affected the usability of the prototype.

5.2.2 User study

Due to the circumstances of the user study and the required sanitary safety measures at that time, it was not possible to gather as many participants as would have been optimal. It was only possible to include as many people in the study, as the test involved attempts on not just one prototype but also the competitor's software. Hence it took the participants around one hour to complete the study. This means that the findings of this study might not be very representative, but they still could show us some interesting trends.

5.3 Future Work

5.3.1 VR Application

The graphics for the prototype were kept simple. This was to reduce visual complexity for the user as well as to improve the testability of the prototype. VR Applications need to be very performant to reduce lag and avoid giving the user symptoms of simulator sickness. But with more time and resources we believe it could be beneficial to use advanced graphics to improve the immersion of the VR experience. The resulting tree could also be improved by including more realistic bark textures, and leaves. One step to improve the realism of the resulting structure would be to implement scaling of the width of the branches and trunk. There are very interesting algorithms that can be applied to the tree's shape. For example, Leonarda da Vinci is credited to have discovered that trees conserve their diameter at different heights [Elo11]. This could be applied to generate a realistic-looking diameter of the trunk and branches. More 3D Interaction methods could be developed in the future to control other parameters of the L-System as the modern VR Controllers are more versatile than the 3D Stylus used by Onishi et al [OMKK06].

Another future improvement could be focused on the visualisation of the tree growth. In the prototype of this work, the growth steps of the L-System are discrete, but by interpolating between two growth steps, continuous growth could be achieved.

5.3.2 L-System

The prototype developed for this thesis exposes the rules of the L-Systems by parameters, but not all were made accessible to the user. A few parameters were set as constants to make sure the resulting L-System resembles a tree-like structure. In the future, those parameters could also be exposed to the user. Although some of the parameters were not used in the prototype, because they may be overly complex, it demonstrates how L-Systems can be adapted to the user's wishes further by parameterizing more elements of the algorithm. For future work, it would be interesting to expose even more parameters to the user and develop interaction techniques to manipulate more of them.

5.3.3 Usability

In the future, it could be possible to add blueprints of L-System examples to give the user a structure to start from. Those could include examples of different tree types and shapes. This tree "phenotype" could be selected together with matching bark texture and leaves.

One benefit of L-Systems is that the complex shapes of trees can be stored in a simple set of instructions and parameters. This could be used to generate a wide range of tree models from one user-created L-System. To enable this we could define a shared tree data model based on L-System. Then one could use separate tools for first creating the L-System in a sketch-based approach as proposed by this work, and secondly use another application to generate life-like tree from the data structure. This could be used in virtual environments to cheaply store and render lots of realistic trees.

The sketching could be advanced further by introducing Bézier curves. By placing 3 points, 4 including the starting node, the user could draw intricate curves for smoother tree shapes. The 4-point definition of Bézier curves would enable easily storing them and rendering them with node rewriting by looking them up in the dictionary, as it is already implemented for the user-drawn nodes in the prototype.

5.3.4 Sketch-based modeling

We believe the future of sketch-based modeling is an exciting one, as advancements in technology are paving the way for new hardware and input methods. These developments promise to significantly enhance the user experience and streamline the modeling process. With virtual reality, users will have the ability to create and manipulate 3D models in a fully immersive environment, allowing for a more natural and intuitive design process. AI technology also holds immense potential for sketch-based modeling, as it can be used to automate and optimize certain aspects of the process, reducing the time and effort required by users. As these technologies continue to evolve, the possibilities for 3D modeling are limitless, and it is exciting to consider the new and innovative ways that sketch-based modeling will be used in the future.

Summing up the work on the prototype and the insights of the user study has made us believe that sketch-based modeling, combined with virtual reality, and "naturalistic" as well as "magic" interaction methods, could be a powerful tool to improve and speed up content creation. We are looking forward to even more immersive user experiences in the future.

List of Figures

List of Tables

List of Algorithms

Bibliography

[AD86] Harold Abelson and Andrea DiSessa. *Turtle geometry: The computer as a medium for exploring mathematics*. MIT press, 1986.

[All03] William Allen. Plant Blindness. *BioScience*, 53(10):926–926, 10 2003.

[Als22] Thomas Alsop. Virtual reality (vr) - statistics & facts. `https://www.statista.com/topics/2532/virtual-reality-vr/#dossierKeyfigures`, 2022. Accessed: 2022-20-03.

[ASSJ06] Fabricio Anastacio, Mario Costa Sousa, Faramarz Samavati, and Joaquim A Jorge. Modeling plant structures using concept sketches. In *Proceedings of the 4th international symposium on Non-photorealistic animation and rendering*, pages 105–113, 2006.

[B+96] John Brooke et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[BKL04] Doug A. Bowman, Ernst Kruijff, and Joseph J LaViola. *3D User Interfaces: Theory and Practice*. Addison Wesley Professional, [Place of publication not identified], 1st edition edition, 2004.

[BKLJP04] Doug Bowman, Ernst Kruijff, Joseph J LaViola Jr, and Ivan P Poupyrev. *3D User interfaces: theory and practice, CourseSmart eTextbook*. Addison-Wesley, 2004.

[Bla18] Anton Hendrik Blaauw. Licht und wachstum. Technical report, Veenman, 1918.

[BMR12] Doug A Bowman, Ryan P McMahan, and Eric D Ragan. Questioning naturalism in 3d user interfaces. *Communications of the ACM*, 55(9):78–88, 2012.

[Bri10] Brian Robert Marshall. Japanese maple, westonbirt arboretum, 2010. [Online; accessed April 16, 2022, published under Creative Commons Licence].

[BWK20] Pauline Bimberg, Tim Weissker, and Alexander Kulik. On the usage of the simulator sickness questionnaire for virtual reality research. In *2020 IEEE conference on virtual reality and 3D user interfaces abstracts and workshops (VRW)*, pages 464–467. IEEE, 2020.

[CNX+08] Xuejin Chen, Boris Neubert, Ying-Qing Xu, Oliver Deussen, and Sing Bing Kang. Sketch-based tree modeling using markov random field. In *ACM SIGGRAPH Asia 2008 papers*, pages 1–9. 2008.

[Con99] William Jay Conover. *Practical nonparametric statistics*, volume 350. john wiley & sons, 1999.

[DJ13] Vivianette Ocasio De Jesus. *Impact of graphical fidelity on a player's emotional response in video games*. PhD thesis, Purdue University, 2013.

[Elo11] Christophe Eloy. Leonardo's rule, self-similarity, and wind-induced stresses in trees. *Physical review letters*, 107(25):258101, 2011.

[GJB+20] Jianwei Guo, Haiyong Jiang, Bedrich Benes, Oliver Deussen, Xiaopeng Zhang, Dani Lischinski, and Hui Huang. Inverse procedural modeling of branching structures by inferring l-systems. *ACM Transactions on Graphics (TOG)*, 39(5):1–13, 2020.

[Han97] Chris Hand. A survey of 3d interaction techniques. In *Computer graphics forum*, volume 16, pages 269–281. Wiley Online Library, 1997.

[HBDP17] Torsten Hädrich, Bedrich Benes, Oliver Deussen, and Sören Pirk. Interactive modeling and authoring of climbing plants. In *Computer Graphics Forum*, volume 36, pages 49–61. Wiley Online Library, 2017.

[hou] Sidefx. `https://www.sidefx.com/products/houdini/`. Accessed: 2022-07-15.

[IOI06] Takashi Ijiri, Shigeru Owada, and Takeo Igarashi. The sketch l-system: Global control of tree modeling using free-form strokes. In *International symposium on smart graphics*, pages 138–146. Springer, 2006.

[JM92] DL Jenkins and RR Martin. Applying constraints to enforce users' intentions in free-hand 2-d sketches. *Intelligent Systems Engineering*, 1(1):31–49, 1992.

[KLBL93] Robert S Kennedy, Norman E Lane, Kevin S Berbaum, and Michael G Lilienthal. Simulator sickness questionnaire: An enhanced method for quantifying simulator sickness. *The international journal of aviation psychology*, 3(3):203–220, 1993.

[Kuh07] Cris Kuhlemeier. Phyllotaxis. *Trends in plant science*, 12(4):143–150, 2007.

[Kus09] Alexander Kusternig. *Real-time rendering of dynamic vegetation*. Vienna University of Technology Austria, 2009.

[KYZ14] Ismail Khalid Kazmi, Lihua You, and Jian Jun Zhang. A survey of sketch based modeling systems. In *2014 11th International Conference on Computer Graphics, Imaging and Visualization*, pages 27–36. IEEE, 2014.

[Lan17] Jaron Lanier. *Dawn of the new everything: A journey through virtual reality*. Random House, 2017.

[Lin68] Aristid Lindenmayer. Mathematical models for cellular interactions in development i. filaments with one-sided inputs. *Journal of theoretical biology*, 18(3):280–299, 1968.

[LJLZ12] Jia Liu, Zhiguo Jiang, Hongjun Li, and Xiaopeng Zhang. Easy modeling of realistic trees from freehand sketches. *Frontiers of Computer Science*, 6(6):756–768, 2012.

[Nik86] Karl J Niklas. Computer-simulated plant evolution. *Scientific American*, 254(3):78–87, 1986.

[ocD] Meta quest, user input. https://developer.oculus.com/resources/bp-userinput. Accessed: 2023-03-13.

[Och98] Gabriela Ochoa. On genetic algorithms and lindenmayer systems. In *International Conference on Parallel Problem Solving from Nature*, pages 335–344. Springer, 1998.

[OMKK06] Katsuhiko Onishi, Norishige Murakami, Yoshifumi Kitamura, and Fumio Kishino. Modeling of trees with interactive l-system and 3d gestures. In *International Workshop on Biologically Inspired Approaches to Advanced Information Technology*, pages 222–235. Springer, 2006.

[OOI06] Makoto Okabe, Shigeru Owada, and Takeo Igarashi. Interactive design of botanical trees using freehand sketches and example-based editing. In *ACM SIGGRAPH 2006 Courses*, pages 18–es. 2006.

[OSSJ09] Luke Olsen, Faramarz F Samavati, Mario Costa Sousa, and Joaquim A Jorge. Sketch-based modeling: A survey. *Computers & Graphics*, 33(1):85–103, 2009.

[PL12] Przemyslaw Prusinkiewicz and Aristid Lindenmayer. *The algorithmic beauty of plants*. Springer Science & Business Media, 2012.

[PMKL01] Przemyslaw Prusinkiewicz, Lars Mündermann, Radoslaw Karwowski, and Brendan Lane. The use of positional information in the modeling of plants. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 289–300, 2001.

[Pot10] Robin Potanin. Forces in play: The business and culture of videogame production. In *Proceedings of the 3rd International Conference on Fun and Games*, pages 135–143, 2010.

[Pru90] Przemyslaw Prusinkiewicz. Visualization of botanical structures and processes using parametric l-systems. *Scientific visualization and graphics simulation*, 1990.

[PSK+12] Sören Pirk, Ondrej Stava, Julian Kratt, Michel Abdul Massih Said, Boris Neubert, Radomír Měch, Bedrich Benes, and Oliver Deussen. Plastic trees: interactive self-adapting botanical tree models. *ACM Transactions on Graphics (TOG)*, 31(4):1–10, 2012.

[San18] Gilbert Sanders. Between tech and art: The vegetation of horizon zero dawn. GDC 2018, 2018.

[SJ13] Marie Saldana and Christopher Johanson. Procedural modeling for rapid-prototyping of multiple building phases. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 5:W1, 2013.

[SKD97] Kay M Stanney, Robert S Kennedy, and Julie M Drexler. Cybersickness is not simulator sickness. In *Proceedings of the Human Factors and Ergonomics Society annual meeting*, volume 41, pages 1138–1142. SAGE Publications Sage CA: Los Angeles, CA, 1997.

[SOJ11] Faramarz Samavati, L Olsen, and JA Jorge. *Sketch-based interfaces and modeling*. Springer, 2011.

[Sou05] MC Sousa. Computer tools for the science illustrator-what do you need. In *The Guild of Natural Science Illustrators (GNSI), Annual Conference*, 2005.

[spe] Speed tree. `https://store.speedtree.com/`. Accessed: 2022-18-08.

[SPK+14] Ondrej Stava, Sören Pirk, Julian Kratt, Baoquan Chen, Radomír Měch, Oliver Deussen, and Bedrich Benes. Inverse procedural modelling of trees. In *Computer Graphics Forum*, volume 33, pages 118–131. Wiley Online Library, 2014.

[SSS06] Aaron Severn, Faramarz Samavati, and Mario Costa Sousa. Transformation strokes. In *SBM*, pages 75–81, 2006.

[TBSR04] Steve Tsang, Ravin Balakrishnan, Karan Singh, and Abhishek Ranjan. A suggestive interface for image guided 3d sketching. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 591–598, 2004.

[TPC18] Minh-Xuan Truong, Anne-Caroline Prévot, and Susan Clayton. Gamers like it green: The significance of vegetation in online gaming. *Ecopsychology*, 10(1):1–13, 2018.

[TSD16] Julian Togelius, Noor Shaker, and Joris Dormans. Grammars and l-systems with applications to vegetation and levels. In *Procedural Content Generation in Games*, pages 73–98. Springer, 2016.

[UWC90] David G Ullman, Stephen Wood, and David Craig. The importance of drawing in the mechanical design process. *Computers & graphics*, 14(2):263–274, 1990.

[Vai22] Lionel Sujay Vailshery. Computer graphics software: modeling and animation segment market size from 2012 to 2024. `https://www.statista.com/statistics/269255/computer-graphics-software-market-value-in-the-digital-video-segment/`, 2022. Accessed: 2022-20-03.

[Vid21] Video games in the us - market size 2005–2027. `https://www.ibisworld.com/industry-statistics/market-size/video-games-united-states/`, 2021. Accessed: 2022-20-03.

[xfr] Xfrog inc. `https://www.xfrog.net/`. Accessed: 2022-18-08.

[YH21] Qi Yuan and Yongjian Huai. Immersive sketch-based tree modeling in virtual reality. *Computers & Graphics*, 94:132–143, 2021.

Appendix

The code of the prototype, the questionnaires from the user study, the consent form as well as the results can be found on this github repository:

https://github.com/mps2209/MasterThesis