Erschien zuerst in: Informatik-Spektrum 20 (1997), Heft 4, S. 208-215

Erzeugung komplexer botanischer Objekte in der Computergraphik

Bernd Lintermann* Oliver Deussen†

Zusammenfassung Der Beitrag beschäftigt sich mit der Herstellung von 3D-Pflanzenmodellen. Ausgehend von traditionellen Methoden wird ein neues Verfahren vorgestellt, das über regelbasierte Objekterzeugung eine Reihe Modellierprobleme löst. Insbesondere wird gezeigt, wie hiermit strukturelle und geometrische Komplexität beherrscht und integrativ modelliert werden können.

Schlüsselwörter Regelbasierte Systeme, L-Systeme, Pflanzen, verzweigende Strukturen, geometrisches Modellieren, strukturelles Modellieren

Summary The present article deals with creating 3-d models of natural looking plants. Traditional approaches are compared with a new method using rule-based object generation. The method separates structural information into two parts. This helps to solve a number of modelling problems. We show how to control geometrical and structural complexity during modelling in the same system.

Key words Rule-based systems, L-systems, plants, branching structures, geometrical modelling, structural modelling

Computing Reviews Classification: I.3.4, I.3.6, I.3.7, J.5

1 Einführung

War vor zehn Jahren der Auftraggeber zum Bau eines Flugzeuges mit Plänen, Zeichnungen und hölzernen Prototypen zufriedenzustellen, so werden heute oftmals Präsentationen in Form von photorealistischen Computergraphiken, Animationen oder gar Simulationen in Echtzeit verlangt. Dabei wachsen die zu behandelnden Datenmengen mit den Qualitätsanforderungen.

So umfaßt die geometrische Beschreibung der Boeing 777 etwa 10 Millionen Polygone, eine Komplexität, die den amerikanischen Virtual-Reality-Forscher Fred Brooks Jr. jüngst veranlaßte zu fragen ([12]), ob bei durchschnittlichen Polygongrößen von unter einem Bildschirmpixel eine polygonale Repräsentation überhaupt noch adäquat ist.

Ähnliche Datenmengen entstehen auch bei architektonischen Objekten, aber insbesondere auch im Rahmen der Visualisierung medizinischer und naturwissenschaftlicher Daten.

Auch hat man es bei diesen Anwendungen längst nicht mehr nur mit kommerziell unbedeutenden Marktsegmenten zu tun. So entfielen 1995 auf den Computergraphik-Bereich 15% des IT-Gesamtumsatzes in den USA ([12]). Hiervon wird ein großer Teil in kommerziellen Präsentationen, Animationen und Werbe-Clips, in der Herstellung von Spezialeffekten und im Studiobereich umgesetzt.

Neue Probleme treten hinzu, wenn komplexe Datenmengen modelliert, also interaktiv erzeugt und verändert werden sollen. Neben entsprechender Speicherkapazität und Graphikleistung sind hier auch effiziente Algorithmen vonnöten, die es ermöglichen, komplexe Daten mittels überschaubarer Beschreibungen zu generieren.

Im folgenden wird die Herstellung natürlich wirkender botanischer Objekte als Beispiel für Modellierproblematik und effiziente Lösungsansätze im Umgang mit großen Datenmengen verwendet. Solche Pflanzenmodelle, angefangen von einfachen Blumen und Büschen bis hin zu komplexen Bäumen mit mehreren Millionen Polygonen, stellen aus der Sicht der Computergraphik eine besondere Herausforderung dar, weil sie einerseits aufgrund der Seherfahrung des Menschen äußerst kritisch “beäugt” und andererseits durch die anfallenden Datenmengen schwer zu erzeugen sind.

Benötigt werden diese Modelle für die Herstellung real wirkender computererzeugter Bilder und Animationen. So leben nicht nur Bilder für Werbezwecke, sondern beispielsweise auch CAD-Darstellungen aus dem Architekturbereich von einer natürlich aussehenden Umgebung. Pflanzenmodelle von hoher Qualität helfen hierbei, daß Bilder nicht als “poliert” empfunden werden, sondern gerade auch durch Modelle mit bewußt eingebauten Asymmetrien und Fehlern einen natürlichen Eindruck erwecken.

2 Pflanzenmodelle

Seit den sechziger Jahren werden algorithmische Beschreibungsverfahren für botanische Objekte untersucht ([18, 3]). Die Arbeiten von Aristid Lindenmayer bildeten hierbei die theoretischen Grundlagen für die ersten, im entferntesten natürlich wirkenden Pflanzen ([9, 10], Monographie siehe [17]).

2.1 L-Systeme

Die nach Lindenmayer benannten L-Systeme sind Textersetzungssysteme auf der Grundlage formaler Grammatiken. Die hiermit erzeugten Zeichenketten bilden eine Geometriebeschreibung der Pflanze und werden in einem zweiten Schritt graphisch interpretiert. Im Gegensatz zu Chomsky-Grammatiken werden Produktionen in L-Systemen nicht sequentiell, sondern parallel angewandt. Die Motivation für dieses Vorgehen entstammt der Biologie, da L-Systeme parallel ablaufende Prozesse, wie etwa die Zellteilung, beschreiben sollen. Tatsächlich hat die parallele Regelanwendung

*ZKM-Karlsruhe sowie Institut für Betriebs- und Dialogsysteme, Universität Karlsruhe, e-mail:linter@zkm.de, http://i31www.ira.uka.de/~linter

†Institut für Simulation und Graphik, Otto-von-Guericke-Universität Magdeburg, e-mail: deussen@isg.cs.uni-magdeburg.de, http://isgwww.cs.uni-magdeburg.de/~deussen

Einfluß auf die Mächtigkeit von L-Systemen. So gibt es kontextfreie L-Systeme, die keine äquivalente kontextfreie Chomsky-Grammatik besitzen ([17]).

Zur graphischen Interpretation der Zeichenketten bedient man sich der sog. Turtle-Graphik. Ein Zeichengerät, die Turtle, wird durch eine Folge textueller Kommandos im Raum bewegt und definiert hierdurch Strecken sowie Flächenumrandungen. Die üblichen Fahrkommandos sind in Tabelle 1 zusammengefaßt.

Tabelle 1: Turtle-Fahrkommandos bzw. Terminalzeichen von L-Systemen

F	Bewege Turtle um Länge d in Richtung des aktuellen Winkels α fort und zeichne eine Linie
f	Bewege Turtle um Länge d in Richtung des aktuellen Winkels α fort (ohne Zeichnen)
+	Drehe Turtle um Winkel δ bzgl. z-Achse
-	Drehe Turtle um Winkel $-\delta$ bzgl. z-Achse
&	Drehe Turtle um Winkel δ bzgl. y-Achse
∧	Drehe Turtle um Winkel $-\delta$ bzgl. y-Achse
\	Drehe Turtle um Winkel δ bzgl. x-Achse
/	Drehe Turtle um Winkel $-\delta$ bzgl. x-Achse
|	Wende Turtle nach rückwärts
[	Lege graphischen Zustand auf den Keller
]	Restauriere graphischen Zustand mittels Keller
{	Beginne Definition zu füllender Fläche
}	Beende Definition zu füllender Fläche

Über diese Kommandos kann mit der folgenden Grammatik der Busch aus Abbildung 1 erzeugt werden (Maximale Anzahl Produktionsanwendungen = 5, $\delta = 22.5$):

$$
\begin{aligned}
\omega &: A \\
p_1 &: A ::= [\&FL!A]/////'[\&FL!A]///////'[\&FL!A] \\
p_2 &: F ::= S/////F \\
p_3 &: S ::= FL \\
p_4 &: L ::= [''' \wedge \wedge \{-f+f+f-|-f+f+f\}]
\end{aligned}
$$

A ist das Axiom der Grammatik. Hiervon ausgehend erzeugt die Anwendung von Produktion p_1 drei Teilszenen ($[\&FL!A]$). Jede Teilszene wird gedreht (&), verschoben, ein Zweig wird erzeugt (F) und schließlich ein Blatt (Produktion p_4). Anschließend reproduziert sich die Szene durch Produktionsanwendung auf (A) selbst. Die Zeichen (!) und (´) werden zur Kontrolle der Farbe und des Stammdurchmessers verwendet.

Zur Erzeugung der Zeichenkette sei an dieser Stelle nochmals angemerkt, daß pro Generierungsschritt in einem L-System alle ersetzbaren Zeichen durch die rechte Seite der anwendbaren Produktion ersetzt werden. Nach Abbruch der Textgenerierung werden alle übriggebliebenen Nichtterminale gelöscht.

Durch Erweiterungen wie parametrisierte und stochastische Regelanwendung sowie durch kontextsensitive Grammatiken können weitere Klassen von Objekten erzeugt und über-

Abbildung 1: Mit L-System hergestellter Busch (aus [17])

dies eine Reihe biologischer Mechanismen und auch Umgebungseinflüsse [16, 14] simuliert werden.

Es zeigt sich aber auch, daß das Arbeiten mit L-Systemen Nachteile mit sich bringt:

- L-Systeme sind sehr abstrakte Repräsentationen natürlicher Objekte. Es ist schwer, Regelsysteme für spezifische Pflanzen herzustellen. Dies kann für praktisch interessante Modelle auch nur über Ausprobieren geschehen, da die automatische Generierung (gegeben die Geometrie, gesucht die erzeugende Grammatik, auch bekannt als Inferenzproblem) nicht auf komplexe Modelle anwendbar ist.
- Geometriedaten werden nur über den Turtle-Graphics-Mechanismus erzeugt. Dies ist im Falle komplexer Geometrien umständlich. Übliche graphische Modelliermethoden für solche Objekte arbeiten über Polygonnetze oder Freiformflächen und enthalten beispielsweise texturierte Oberflächen.
- Algorithmische Vervielfältigung ist nicht möglich. An vielen Stellen in der Natur werden Objekte jedoch über Plazierungsalgorithmen angeordnet. Beispiel hierfür ist das Verfahren des goldenen Schnitts (typisches Beispiel: die Anordnung von Samenkapseln einer Sonnenblume) oder Plazierung durch Wettbewerb um Licht. Soll eine solche Anordnung erzeugt werden, muß bei der Verwendung von L-Systemen auf externe Programme zurückgegriffen werden.

Insbesondere die intuitiv schlecht zu erfassende Arbeitsweise mit L-Systemen führte zu weiteren Modellieransätzen speziell für Bäume, die im folgenden skizziert werden.

2.2 Parametrische Beschreibungen

Parametrische Beschreibungen haben ein generisches Modell als Grundlage, das über eine Menge von Parametern modifiziert werden kann. Die veränderbaren Eigenschaften des Modells werden hierbei auf einen praktisch relevanten Teil eingeschränkt, freilich um den Preis einer geringeren Flexibilität.

Solche Parametersätze können z.B. biologischen Ursprungs sein. So stellten De Reffye und andere ([4]) ein Modell auf der Basis von wachsenden Knospen vor, Holton entwickelte

ein Modell auf der Grundlage von Strängen ([8]), die von der Wurzel zu jeder Stelle des Baums wachsen. Die Anzahl der Stränge in einer Verzweigung bestimmt den Abzweigungswinkel sowie die Verjüngung und Länge des Astes - eine ähnliche Beschreibung hat übrigens schon Leonardo da Vinci (siehe [13]) entwickelt und damit die Proportionen von Bäumen analysiert.

Relativ grobe parametrische Beschreibungen werden auch in AMAP ([2]), einer kommerziellen Bibliothek von Baummodellen und Erzeugungsverfahren, verwendet. Bei Angabe von Typ und Alter eines Baums wird die entsprechende Geometrie erzeugt.

3 Ein kombiniertes Verfahren

Uns interessierte die Frage, wie man die Mächtigkeit von Regelsystemen mit der intuitiv leicht zu erfassenden Arbeitsweise parametrischer Ansätze verbinden kann (siehe hierzu auch [11, 5]).

Unser Ansatz arbeitet ähnlich L-Systemen mit einem Regelsatz. Im Unterschied zu L-Systemen werden hiermit aber nicht textuelle Symbole erzeugt, die anschließend graphisch interpretiert werden, sondern Bausteine.

Bausteine haben vieles mit den Objekten einer objektorientierten Programmiersprache gemein. So enthalten sie eine Menge von Daten und Methoden, um geometrische Daten zu erzeugen, aber auch Algorithmen, die andere Bausteine vervielfältigen.

3.1 Aufbau des Regelsystems

Das Regelsystem wird durch einen gerichteten Graphen (den Prototyp- oder *p-Graph*) beschrieben. Die Knoten des Graphen sind Baustein-Prototypen, seine Kanten symbolisieren Instantiierungsregeln, die dazu verwendet werden, einen temporären Baum aus Baustein-Instanzen (wir nennen ihn *i-Baum*) aufzubauen. Die Baustein-Instanzen im i-Baum erzeugen in einem zweiten Schritt die Geometriedaten.

Auf diese Weise wird strukturelle Information in zwei Teile zerlegt: als Topologie des Graphen und in Baustein-Parameter. Diese Separation wird sich im folgenden als nützlich erweisen.

Die Instantiierungsregel zu einer gerichteten Kante im p-Graph lautet umgangsprachlich so: Wurde der Prototyp des Quell-Bausteins kopiert, um eine Instanz im i-Baum zu bilden, so tue dies auch mit dem Zielbaustein und verbinde beide durch eine Kante.

Ist der Quell-Baustein der Kante ein Multiplikator-Baustein, so wird im i-Baum nicht eine, sondern es werden mehrere Instanzen des nachfolgenden Bausteines erzeugt. Deren Anzahl wird hierbei durch die Parameter des Multiplikator-Bausteins festgelegt.

Ähnlich wird mit Rekursionen verfahren. Jeder Baustein besitzt einen Parameter, der die maximale Rekursionstiefe beschreibt, bis zu der er instantiiert wird. Rekursionen im p-Graphen können jetzt bei der Herstellung des i-Baums zu Teilbäumen aufgelöst werden. Die Umwandlung erlaubt es, geschachtelte Rekursionen zu definieren; überlappende Rekursionen sind theoretisch auch möglich, werden aber hier nicht benötigt und auch nicht unterstützt.

Folgendes Agorithmenschema beschreibt das Umwandlungsverfahren. Ein Parameter n eines Bausteins A ist hierbei mit $A : n$ angegeben:

```
Algorithmus ErzeugeTeilbaum(Baustein Q)
begin
  Erhöhe Q:Rekursionstiefe um eins;
  if (Q:Rekursionstiefe < Q:MaximaleRekursionen) dann
    Instantiiere Q;
    für jeden Zielbaustein Z von Q begin
      if Q ist Multiplikator-Baustein
        dann für i von 1 bis Q : n ErzeugeTeilbaum(Z);
        sonst ErzeugeTeilbaum(Z);
    end
  end
end
```

Die oben angegebene Instantiierungsregel entspricht am ehesten einer kontextfreien Produktion der Form $A ::= A_i B^n$, wobei A und B den Baustein-Prototypen entsprechende Nichtterminale, A_i Terminalsymbol der Instanz von A und n Multiplikationsfaktor ist.

Das Modellieren mit Bausteinen ähnelt also dem Arbeiten mit kontextfreien Grammatiken. Auf der anderen Seite ist n als Parameter des Bausteins A gespeichert und wird von diesem bei der Regelanwendung gesetzt. Dieser Mechanismus ist mit Grammatiken nicht nachzubilden.

3.1.1 Ein Beispiel

Folgendes Beipiel soll den Prozeß verdeutlichen: Baustein-Prototyp A sei die Wurzel, Prototyp B ein Geometriebaustein, der einen Stamm erzeugt. C sei ein Multiplikator-Baustein, der im i-Baum jeden Nachfolger verdreifachen soll und selbst keine Geometrie produziert. D soll einen Teil eines Zweiges erzeugen, bei ihm wird die maximale Rekursionstiefe auf drei gesetzt.

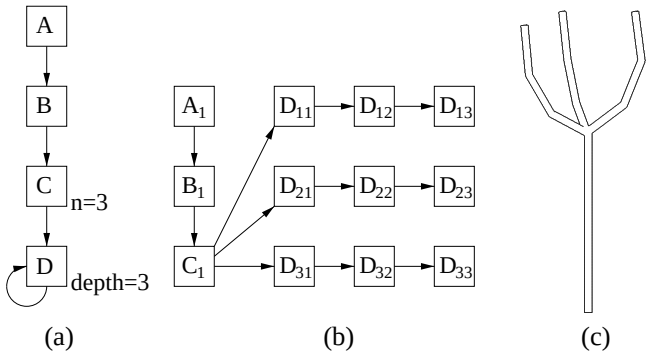


Abbildung 2: Geometrie-Erzeugung: a) Der Benutzer definiert den p-Graph; b) der p-Graph wird in den i-Baum transformiert. c) Die Instanzen erzeugen die Geometriedaten.

In Bild 2(a) ist der p-Graph dargestellt, in Teilbild (b) der resultierende i-Baum: Es werden, entsprechend dem Vervielfältigungsparameter von C, drei Teilbäume aller Nachfolger erzeugt (beginnend mit den Instanzen D_1, D_2, D_3) und mit der von C erzeugten Instanz C_1 verbunden. Da auf D eine Rekursion definiert ist, werden entsprechend der maximalen Rekursionstiefe drei Instanzen D_{i1}, D_{i2} und D_{i3} erzeugt. Insgesamt wird Prototyp D also neun Mal instantiiert.

3.2 Geometrieerzeugung

Im zweiten Schritt wird nun aus dem i-Baum die Geometrie erzeugt (Bild 2(c)). Hierzu wird der Baum in Tiefensuche abgearbeitet, wobei jeder Baustein aufgefordert wird, Geometrie entsprechend seiner Erzeugungsmethode zu produzieren. Erzeugt werden je nach Baustein Dreiecke oder Punktlisten. Letztere werden anschließend trianguliert. Dieses Verfahren ist insbesondere dort interessant, wo verzweigende Strukturen mit geschlossenen Oberflächen erzeugt werden sollen.

Der Leser wird sich vielleicht fragen, wie das individuelle Aussehen der Geometrie der neun Instanzen von D in Bild 2(c) erreicht werden kann. Generell werden zur Geometrieerzeugung lokale Koordinatensysteme verwendet, die von Multiplikator-Bausteinen für jeden Nachfolger individuell bestimmt werden. Das führt auch zum generellen Unterschied zwischen Baustein-Prototypen und -Instanzen: In Prototypen werden Parameter meist als Bereiche definiert, aus denen die einzelnen Werte für die Nachfolger-Instanzen durch Interpolation errechnet werden. Neben den Koordinatensystemen können so auch die Größe und andere Eigenschaften festgelegt werden.

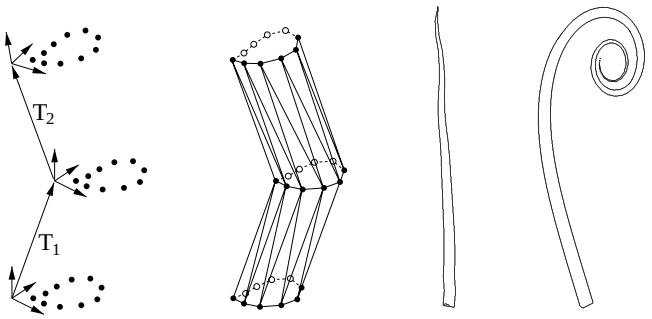
Ist der Parameterbereich $[v_0, v_1]$, so wird der Wert des i-ten Nachfolgers aus n durch $v_i = i(v_1 - v_0)/(n-1)$ bestimmt. Bevor das System solch einen Wert verwendet, kann eine vom Benutzer beliebig definierbare Funktion angewendet werden. Auf diese Weise ist es auch möglich, zufallsgesteuerte Faktoren zu erzeugen.

4 Bausteine

Da wir anstrebten, eine möglichst mächtige Bausteinbasis zu erzeugen, war eine interessante Frage, welche Bausteintypen hierfür nötig sind. Es war eine Reihe von meist praktischen Faktoren, die uns schließlich zu den folgenden Typen kommen ließen.

Die ersten vier Bausteintypen erzeugen Geometrie:

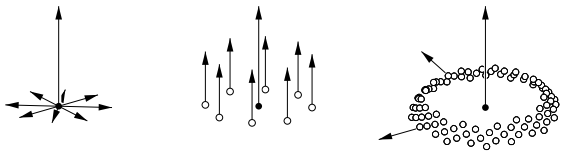
- **Simple:** Bausteine dieses Typs haben nur Grundfunktionalität. Sie können geometrische Primitive (Kugel, Zylinder, Quader) erzeugen und die Geometrie nachfolgender Bausteine transformieren.
- **Horn:** Wird verwendet, um Stämme und Äste zu erzeugen. Die erzeugte Geometrie hat die Form eines Kegels und kann durch eine Reihe von Parametern verändert (z.B. gebogen) werden.



(a) (b) (c)

Abbildung 3: Geometriedefinition eines Horn-Bausteins: a) Punktmengen werden definiert über relative Transformationen zueinander; b) die Oberfläche wird durch Triangulation erzeugt. c) Zwei Beispiele.

- **Blatt:** Hiermit werden alle Arten von Blättern definiert. Die Form kann eingestellt werden, Blätter können gefaltet oder an den Rändern gewellt werden.
- **Baum:** Sequenzen dieser Bausteine werden benutzt, um Bäume zu erzeugen (siehe unten). Erzeugt wird ein Horn, das hier aber im Gegensatz zum Horn-Baustein mit anderen Parametern, wie beispielweise Tropismen, in seiner Erscheinung verändert werden kann. Nachfolgende Baumbausteine können als Verzweigung multipliziert werden.



(a) (b) (c)

Abbildung 4: Orientierung von multiplizierten Bausteinen: a) Hydra; b) Ring; c) Phiball.

Die zweite Gruppe bilden Multiplikator-Bausteine:

- **Hydra und Ring:** Diese Bausteine multiplizieren ihre Nachfolger stern- oder ringförmig.
- **Phiball:** Der Phiball ordnet seine Nachfolger nach dem Verfahren des goldenen Schnitts an (Genaueres siehe [1]). Wir haben gängige Verfahren, die eine Anordnung auf Kreisscheiben ermöglichen, auf Kugelausschnitte erweitert, was eine wesentlich größere Flexibilität erlaubt.

Die Bausteine der letzten Gruppe werden dazu verwendet, die erzeugte Gestalt zu verändern und Tropismen (Wuchsrichtungen) zu definieren:

- **Freiformdeformation (FFD):** Hiermit kann auf der Gesamtheit oder einem Teil der Daten eine Freiformdeformation ausgeführt werden. Eine Deformation von Teildaten ist beispielsweise dann sinnvoll, wenn die Zweige eines Baums verformt werden sollen, aber nicht die Blätter oder Nadeln.

- **Welt:** Die räumliche Ausrichtung der Geometriedaten von Blatt- und Baum-Bausteinen kann durch Photo- und Gravitropismen beeinflußt werden. Normalerweise zeigen die Gradienten der zugehörigen Felder in positive bzw. negative z-Richtung. Durch einen Welt-Baustein können diese Definitionen durch beliebige vektorwertige Funktionen ersetzt werden.

In Bild 5 sind die graphischen Symbole der Bausteine dargestellt. In den folgenden Beispielen wird zusätzlich ein Wurzelbaustein verwendet, der durch ein Kamerasymbol gekennzeichnet ist.

(a) (b) (c) (d) (e) (f) (g) (h) (i)

Abbildung 5: Graphische Symbole der Bausteine: a) Simple; b) Horn; c) Blatt; d) Baum; e) Hydra; f) Ring; g) Phiball; h) FFD; i) Welt.

4.1 Kantentypen

Die generelle Bedeutung einer Kante des p-Graphen wurde bereits dargestellt. Allerdings gibt es Fälle, die besonders berücksichtigt werden müssen. Etwa muß bei einem Baum-Baustein zwischen Nachfolgern unterschieden werden, die nur einmal erzeugt werden, und solchen, die als Zweige vervielfältigt werden sollen. Letztere werden durch ein “Branch-Link” (Kennzeichnung: B) angezeigt, während im ersten Fall der normale Kantentyp, der “Child-Link”, verwendet wird.

In rekursiven Graphen kann es ferner sinnvoll sein, nach Abbruch der Rekursion einen Baustein zu instantiieren. Soll das geschehen, wird ein “Leaf-Link” (L) verwendet.

5 Beispiele

Anhand von drei Beispielen möchten wir das Modellieren mit den Bausteinen veranschaulichen. Das erste Beispiel ist ein Löwenzahn, eine Blume, die strukturelle und geometrische Komplexität in sich vereint. Danach wird die Herstellung eines Baumes zeigen, daß sich die Methode auch hier angewenden läßt. Schließlich wird der Busch aus Bild 1 noch einmal modelliert, um Unterschiede und Gemeinsamkeiten mit L-Systemen zu verdeutlichen.

5.1 Löwenzahn

Der Aufbau des Modells eines Löwenzahns orientiert sich an dessen natürlicher Struktur. Zuerst wird mit einem Horn-Baustein ein Haar eines Schirmchens erzeugt (Bild 6(a)). Dazu wird der Baustein an die Kamera angehängt, was zur Folge hat, daß eine Instanz erzeugt wird, die zur Produktion eines Haares aufgefordert wird.

Die geometrische Beschreibung des Haares besteht aus der triangulierten Oberfläche eines Kegels. Da im folgenden sehr

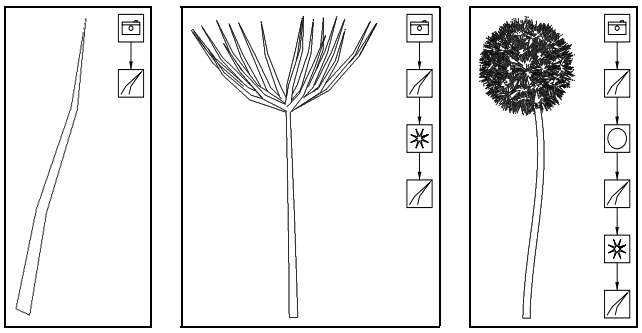


Abbildung 6: Herstellung eines Löwenzahns: a) Haar, (b) Schirmchen, (c) Kugel aus Schirmchen mit Stiel.

viele Haare benötigt werden, ist die geometrische Komplexität gering gewählt. Der Kegel wird ferner durch die Angabe zweier Rotationsparameter gebogen. Der eine Parameter beschreibt die relative Rotation benachbarter Kegelstücke an dessen Anfang, der andere die Rotation am Ende. Zwischen diesen Werten wird für die inneren Kegelstücke interpoliert.

Nun wird der p-Graph so modifiziert, daß ein Multiplikator-Baustein, die Hydra, vor den Horn-Baustein gehängt wird. In diesem Baustein wird die Anzahl zu erzeugender Nachfolger auf 18 gesetzt. Aus dem jetzt entstandenen Graphen wird ein i-Baum mit einer Hydra-Instanz und 18 Horn-Instanzen gebildet. Parameter der Hydra sorgen für eine Rotation der lokal definierten Koordinatensysteme aller nachfolgenden Horn-Bausteine. Nun wird ein weiterer Horn-Baustein vor den Rest gehängt, er definiert den Stiel des Schirmchens (siehe Teilbild (b)).

Durch Addition eines Phiball-Bausteins werden einige hundert dieser Schirmchen auf einer Kugeloberfläche erzeugt. Die Anzahl der Multiplikationen ist wiederum ein Parameter des Phiball-Bausteins. Weitere Parameter sind der Radius der Kugel sowie Transformationen der Koordinatensysteme der Nachfolger. Der Algorithmus zur Plazierung nach dem goldenen Schnitt auf der Kugelfläche ist in [11] beschrieben. Es ist eine Erweiterung früher vorgestellter Verfahren für ebene Flächen.

Vor den Phiball-Baustein wird wieder ein Horn gesetzt, welches den Stiel des Löwenzahns bildet (Teilbild (c)). Die Werte der Rotationsparameter werden ähnlich denen der Schirmchen-Stiele gewählt, um die gewünschte Krümmung hervorzurufen.

Schließlich wird ein Blatt modelliert. Hierzu wird dessen Außenkante über einen Spline definiert, die Blattoberfläche entsteht durch Triangulation. Parameter bestimmen die Krümmung des Blattes sowie dessen Welligkeit. Des weiteren kann eine Textur angegeben werden, die die innere Struktur des Blattes wiedergibt.

Ein Hydra-Baustein wird verwendet, um das Blatt zu vervielfältigen; hierbei können durch einen Parameter Größe und Winkel der einzelnen Blätter variiert werden.

Die Modifikation der geometrischen und strukturellen Parameter geschieht jeweils über graphische Benutzerschnittstellen, die zu jedem Bausteintyp gehören.

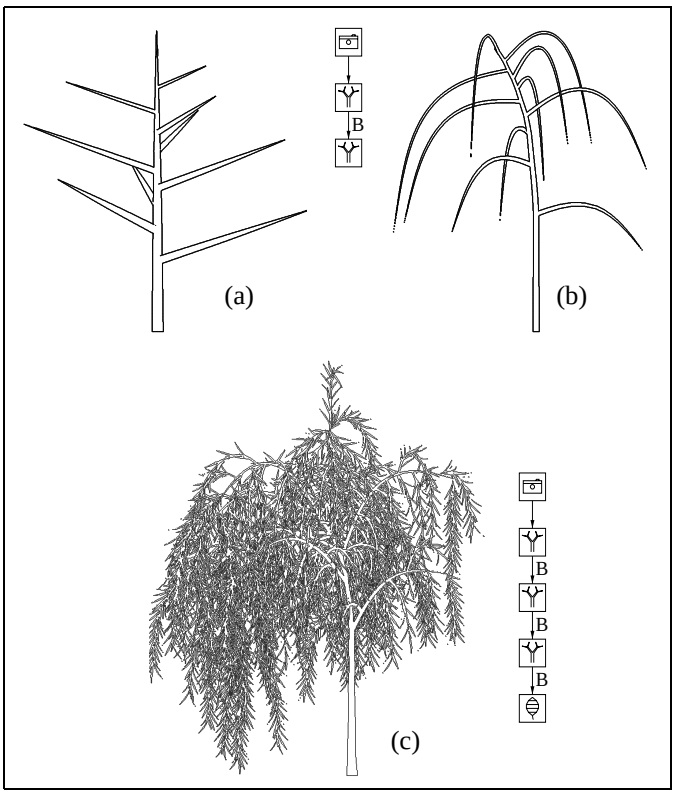


Abbildung 7: Modellieren einer Trauerweide: a) Initialer p-Graph und Resultat. b) Aussehen nach Parameteränderungen; c) fertiger Baum.

5.2 Trauerweide

Ein Baum wird meist durch eine Sequenz von Baum-Bausteinen beschrieben. Zuerst werden zwei Baum-Bausteine kombiniert, wobei ein “Branch-Link” verwendet wird. Das initiale Aussehen sowie der p-Graph sind in Bild 7(a) zu sehen. Nun wird die Gestalt und die Verzweigungscharakteristik des Stammes geändert, außerdem wird ein Gravitropismus definiert, um die Zweige nach unten zu biegen (Teilbild (b)).

Nun wird eine Kopie des zweiten Baum-Bausteines über einen “Branch-Link” an den zweiten angehängt und wieder Gravitropismus, Verzweigungsstruktur, Größe sowie Abzweigungswinkel angepaßt. Schließlich wird ein Blatt-Baustein über einen “Branch-Link” angehängt, um die Blätter zu erzeugen. In Bild 7(c) ist das Ergebnis zu sehen, Bild 9 zeigt ein verfeinertes Modell.

5.3 Zurück zum Busch

Als letztes Beispiel dient der Busch, der in Abschnitt 2 über ein L-System hergestellt wurde. Das L-System arbeitet über eine rekursive Verzweigung, die an einen Zweig jeweils drei kleinere Zweige sowie Blätter ansetzte.

Dieses Verfahren bildet auch hier die Grundlage der Bausteine: Ein PhiBall-Baustein multipliziert einen Baum-Baustein dreimal (man hätte auch einen Hydra-Baustein zur Multiplikation verwenden können). An dem Baum-Baustein werden Blätter mittels “Branch-Links” angebracht, das Ganze wird fünfmal rekursiv ausgeführt. Der p-Graph sowie eine Skizze des Resultats sind in Bild 8 dargestellt.

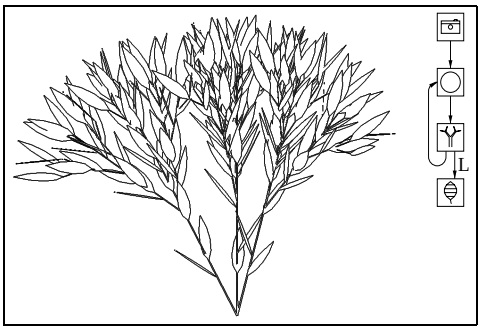


Abbildung 8: Busch, diesmal mit Bausteinen hergestellt

6 Vorteile und Grenzen des Verfahrens

Ein mit L-Systemen generiertes Objekt nachzumodellieren heißt freilich noch nicht, die Mächtigkeit von L-Systemen zu besitzen. Wie bereits in Abschnitt 3.1 angedeutet, entspricht der Aufbau des p-Graphen in etwa einer kontextfreien Grammatik. Daher sind L-Systeme generell mächtiger, indem auch kontextsensitive Grammatiken angegeben werden können. Dieser Unterschied fällt jedoch bei der reinen Geometriemodellierung nicht ins Gewicht, wenn zusätzlich globale Modelliermethoden wie die oben erwähnten Tropismen und Freiformdeformationen verwendet werden ([15]). Viel wichtiger sind auch die praktischen Unterschiede:

- Generierungsalgorithmen sind in eine regelbasierte Beschreibung integrierbar, wie das etwa beim PhiBall-Baustein geschieht.
- Geometrische und strukturelle Parameter können auf dieselbe Weise editiert werden. Jedem Baustein kann eine graphische Benutzerschnittstelle zugeordnet werden, mit der die Parameter optimal eingestellt werden können.
- Bausteine können komplexe Geometrien repräsentieren. Auf diese Weise werden Modellbeschreibungen übersichtlicher.
- Animation kann auf der Grundlage der Parameter leicht integriert werden. Dieser Aspekt kam bisher nicht zum Tragen (siehe auch [11] sowie Bild 11).

  Die Grundlage hierfür bildet die sog. Keyframing-Animation ([6]), bei der man Parameterwerte zu bestimmten Zeiten explizit definiert und sie anschließend für dazwischenliegende Zeitpunkte interpoliert.

Insgesamt, so zeigt eine Benutzerstudie in [5], können mit der Methode Pflanzenmodelle sehr schnell und komfortabel hergestellt werden. Ungeübte Benutzer haben allerdings manchmal Schwierigkeiten im Umgang mit der großen Anzahl von Parametern.

7 Fazit

Der vorliegende Aufsatz stellt eine Modelliermethode für komplexe botanische Objekte vor. Im Gegensatz zu traditionellen regelbasierten Ansätzen wird durch regelbasierte Erzeugung von Bausteinen strukturelle Information auf zweierlei Weise repräsentiert. Der Aufbau des zugrundeliegenden Graphen von Baustein-Prototyen liefert die generelle Beschreibung des Modells, während seine Gestalt über Parameter der Bausteine editiert wird. Der Ansatz erlaubt es, Generierungsalgorithmen zu verwenden sowie Geometrie über Spline-Modellierung, Freiformdeformationen und Beschränkungen zu modellieren.

Offene Probleme liegen in geeigneten Benutzerschnittstellen für eine intuitive Repräsentation der Baustein-Parameter sowie in Verfahren, die es erlauben, die immense geometrische Komplexität nahtlos zu reduzieren. Das Programm kann als Shareware unter [7] bezogen werden.

8 Danksagung

Die Autoren danken Herrn Prof. Dr. Alfred Schmitt von der Universität Karlsruhe sowie Herrn Jeffrey Shaw vom ZKM Karlsruhe. An beiden Stellen wurde der größte Teil der Arbeiten durchgeführt. Inspirationen und Verfahrensansätze zur Skizzierung der Pflanzenbilder kamen von Prof. Dr. Thomas Strothotte und seinen Mitarbeitern an der Otto-von-Guericke-Universität Magdeburg. Frau Sylvia Zabel und Herrn Dipl.-Inf. Andreas Raab sei Dank für die Durchsicht des Aufsatzes.

Literatur

[1] A. Beutelspacher and B. Petri. *Der Goldene Schnitt*. BI Wissenschaftsverlag, Zürich, 1988.

[2] CIRAD/GERDAT. Amap presentation. http://www. cirad.fr/amap/amap.html.

[3] D. Cohen. Computer simulation of biological pattern generation processes. *Nature*, (216):246–248, October 1967.

[4] P. de Reffye, C. Edelin, J. Francon, M. Jaeger, and C. Puech. Plant models faithful to botanical structure and development. In J. Dill, editor, *Computer Graphics (SIGGRAPH '88 Proceedings)*, volume 22, pages 151–158, August 1988.

[5] O. Deussen and B. Lintermann. A modelling method and user interface for creating plants. In *Proc. Graphics Interface '97*. Morgan Kaufmann Publishers, Mai 1997.

[6] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practices (2nd Edition)*. Addison Wesley, 1990.

[7] Greenworks. Home page of the xfrog modelling software. http://www.greenworks.de.

[8] M. Holton. Strands, gravity and botanical tree imagery. *Computer Graphics Forum*, 13(1):57–67, 1994.

[9] A. Lindenmayer. Mathematical models for cellular interactions in development. *Journal of Theoretical Biology*, I&II:280–315, 1968.

[10] A. Lindenmayer. Developmental systems without cellular interaction, their languages and grammers. *Journal of Theoretical Biology*, (30):455–484, 1971.

[11] B. Lintermann and O. Deussen. Interactive modelling and animation of natural branching structures. In Ronan Boulic and Gerad Hegron, editors, *Computer Animation and Simulation ´96*, pages 139–151. Springer-Verlag, Berlin, 1996.

[12] C. Machover. Springing into the fifth decade of computer graphics - where we've been and where we're going (Podiumsdiskussion). In *Computer Graphics (SIGGRAPH '96 Proceedings)*. ACM SIGGRAPH, ACM Press, 1996.

[13] Benôit B. Mandelbrot. *Die fraktale Geometrie der Natur*. Birkhäuser Verlag, 1991.

[14] R. Měch and P. Prusinkiewicz. Visual models of plants interacting with their environment. In H. Rushmeier, editor, *Computer Graphics (SIGGRAPH '96 Proceedings)*, pages 397–410. ACM SIGGRAPH, ACM Press, 1996.

[15] P. Prusinkiewicz. Persönliches Gespräch, Mai 1997.

[16] P. Prusinkiewicz, P. James, and R. Měch. Synthetic topiary. In *Computer Graphics (SIGGRAPH '95 Proceedings)*, pages 351–358, 1995.

[17] P. Prusinkiewicz and A. Lindenmayer. *The Algorithmic Beauty of Plants*. Springer-Verlag, New York, 1990.

[18] S. Ulam. On some mathematical properties connected with patterns of growth of figures. In *Proceedings of Symposia on Applied Mathematics*, volume 14, pages 215–224. American Mathematical Society, 1962.

Bernd Lintermann studiert Informatik an der Universität Karlsruhe mit Schwerpunkt Computergraphik. Sein Interesse gilt neben algorithmischen Verfahren zur Herstellung biologischer Objekte auch deren Umsetzung in der Kunst. Er arbeitet am ZKM Karlsruhe mit Medienkünstlern an Projekten für interaktive Medienkunstwerke.

Dr. Oliver Deussen studierte und promovierte an der Universität Karlsruhe. Seit 1996 ist er wissenschaftlicher Mitarbeiter am Institut für Simulation und Graphik der Otto-von-Guericke-Universität Magdeburg. Er ist Lenkungskreismitglied der GI-Fachgruppe 4.1.4 “Graphische Simulation und Animation”. Seine wissenschaftlichen Interessen liegen u.A. bei der Bewegungssimulation, Darstellung und Animation komplexer natürlicher Objekte.

Abbildung 9: Beispielbilder von Löwenzahn und Trauerweide (nackt und mit Blättern)

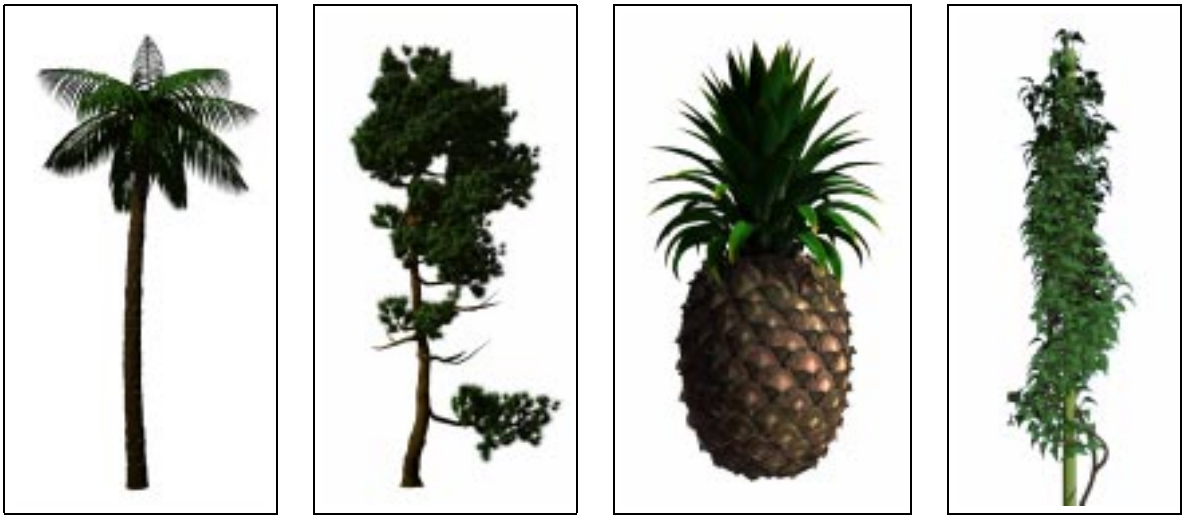


Abbildung 10: Weitere Modelle: a) Palme, b) durch Wind geformte Bergkiefer, c) Ananas, d) Efeu, um Stange wachsend.

Abbildung 11: Bilder aus einer Animation eines wachsenden und schließlich sterbenden Baumes. Die Animation wurde für den Medienkünstler Bill Viola am ZKM Karlsruhe hergestellt.