2003年度　卒　業　論　文

コブ形状および分枝部分の平滑化を考慮した写実的な樹木形状の生成に関する研究

指導教員：渡辺　大地　講師

メディア学部 3DCGアプリケーション構築プロジェクト
学籍番号　00P186
斉藤　寛明

2003 年度　　卒　業　論　文　概　要

論文題目

コブ形状および分枝部分の平滑化を考慮した写実的な樹木形状の生成に関する研究

メディア学部 **学籍番号：00P186**	**氏名**	斉藤　寛明	**指導教員**	渡辺　大地　講師

キーワード	3DCG、　樹木、　モデリング、　コブ

これまで樹木形状のモデリングおよびレンダリングに関する研究は数多くされてきており、樹木を重要視するようなシミュレーションや大量の樹木形状が必要となる映像コンテンツなどの分野で利用されてきた。特にモデリングにおいては樹木形状を自動的に生成することを目的とし、写実的かつ効率的な手法の研究が行なわれてきた。

しかし従来の手法で生成される樹木形状は樹木の骨格モデルの生成に特化したものが主流であり、樹木特有の個性的な隆起形状はあまり考慮していなかった。これは不規則に発生する隆起の表現が困難であることや、モデリングの効率と精度の両立が問題となるためである。そこで本研究では樹木の随所に見られる特徴的な隆起であるコブを枝の奇形生長の一つとして実現する。一方、枝の根元である分枝部分が滑らかになっていなければ不自然なコブを生成することになる。そこで生成コストを極力抑えるために、ユーザが指定した精度で分枝部分のみを局所的に滑らかにする手法を実現する。本研究では以上の 2 点から生成コストを考慮した上でコブ形状を実現し、樹木形状をより自然な形に近づけることを目的とし、その有用性を実証する。その際に樹種ごとにコブの発生要素が異なることを踏まえ、樹種ごとの特徴を考慮した形状生成が可能な Weber らの手法を本研究の基盤とした。

目 次

第 1 章

はじめに

これまで樹木形状のモデリングおよびレンダリングに関する研究は数多くされてきた。特にモデリングにおいては樹木形状を自動的に生成することを目的とし、写実的かつ効率的な手法の研究が行なわれてきた。このような手法は景観を重要視するドライビングシミュレーションや街路樹などを考慮した建設計画における景観評価、大量の樹木形状が必要となる映像コンテンツなどの分野で利用されてきた。

一般的な樹木形状を生成する方法では、樹木の幹や枝、葉の生える位置や方向を決めて樹木の骨格モデルを生成する。その骨格モデルに樹皮としての太さを持たせ樹木形状を表現する。このような樹木形状の生成工程において、これまでにフラクタルと呼ばれる分岐法則を用いた手法 [1] や、環境から受ける影響や樹木の生長を定式化して形状生成を行う手法 [2]-[6]、2 次元画像を基に 3 次元の樹木形状を構成させる手法 [7]-[9] など数多くの手法が研究されてきた。

例えば千葉ら [10] は、架空の植物ホルモンを用いることで樹木の生長法則を実現している。これにより太陽光の向きに樹木が傾く向日性やホルモンバランスを考慮した分枝法則などを樹木形状の生成に適用している。一方、Weber ら [11] は樹木形状を生成するためのパラメータを提案している。分枝角度や枝の関節数などのパラメータを樹種ごとに設定することで個性的かつ写実的な形状生成を実現している。

従来の手法で生成する樹木形状は花や実などの例外を除いて、枝と葉のみで構成するのが一般的である。特に枝は図1.1のように直線的もしくは一定法則に従った凹凸などによって生成することが多い。この背景には樹木形状をシミュレーション等で利用する際に一度に大量のモデルが必要となることが多いため、計算コストの問題から必要最低限の精度で生成することが原因となっている。そのため例えば個性的な隆起を含んだ高品質な樹木のモデリングが求められる場合、従来の手法で生成した樹木形状を手作業で変形する必要があった。

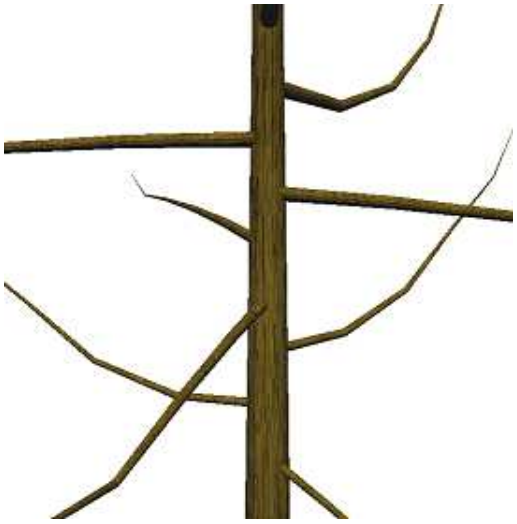

図1.1: 直線的な枝で構成された樹木形状

樹木形状の質を高める個性意的な隆起の一つに「コブ」があげられる。コブは自然界の樹木を見るとほぼ全てに存在し、樹木の個性を特徴付ける重要な要素となっている。しかし自然界におけるコブの発生要因が多種多様かつ不確定要素が多い点や、コブの形が複雑である点から従来の手法において排他されてきた要素であり、その形状を実現する手法は提案されていない。そこで本研究ではコブの発生要因と形状生成を樹種ごとに用意したパラメータを利用することによって、実物に近似したコブ形状を生成する手法を提案する。その際、本研究ではコブの発生要因の一つである休眠芽に着目し、コブを枝の奇形生長の結果として実現した。

精度の高いコブや枝を生成するためには枝の生える接続部分（以下「分枝部分」

とする）を見直す必要がある。例としてあげた研究をはじめ、従来の手法では樹木形状生成の重点を骨格モデルに置いているが、枝や幹としての太さを持たせる工程では円錐台などを利用して単純に生成することが多く、特に枝同士の接続は非常に不自然なものとなっている。これは樹木形状を景観シミュレーションなどで大量に利用することを前提にしていることが大きな理由であり、形状の細部のクオリティをあげることはメモリの消費や計算コストが増大するため不必要とされてきた。しかし本研究では枝を変形してコブを生成するため、分枝部分が滑らかに接合していなければコブの精度が低くなる。そこで分枝部分の精度を任意で制御でき、精度とコストを両立させる効率的な手法を提案する。

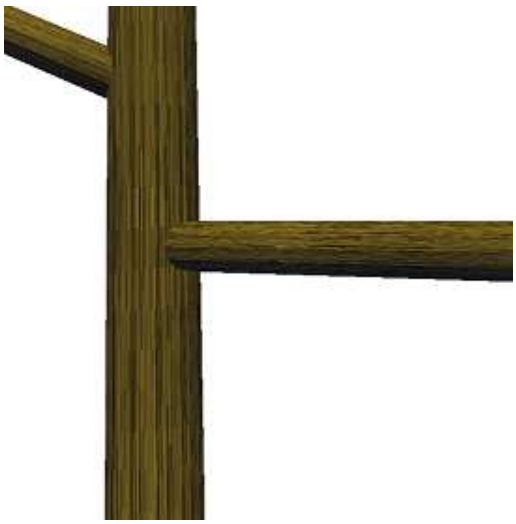

図 1.2: 不自然な分枝部分

本研究では以上の 2 点から生成コストを考慮した上でコブ形状を実現し、樹木形状をより自然な形に近づけることを目的とし、その有用性を実証する。その際に樹木形状を生成する手法として樹種ごとにコブの発生要素が異なることを踏まえて、樹種ごとの特徴を考慮した形状生成が可能な Weber ら [11] の手法を本研究の基盤とした。

本論文の構成は次のとおりである。第 2 章では本研究の基盤となった Weber らの手法について説明する。第 3 章ではコブの必要性および形状生成のためのパラ

メータの定義を述べ、コブ形状の生成について説明する。第 4 章では分枝部分の平滑化について考慮すべき点を述べ、その実現方法について説明する。第 5 章では結論と課題について述べる。

第 2 章

Weberらの手法による樹木形状の生成

2.1 生成に用いるパラメータ

本研究では樹木形状の生成をするベースとしてWeberら[11]の手法を用いた。この手法では樹木形状を生成する枝の数・長さ・分枝角度などをパラメータとして定義し、これらを変化させることで様々な樹種を生成することができる。本研究ではこの手法を採用した。コブ形状の生成および分枝部分の平滑化を行うための幾つかのパラメータを新たに定義することにより、個性的な形状生成ができる。樹木形状生成のためのパラメータはWeberらによって樹種ごとにいくつか提案されており、Webサイト[12]でも公開されている。表2.1はパラメータの一部である。

表2.1: パラメータの一部

パラメータ	用途
Level	分枝の最大深度
Scale	樹木の大きさの指標となる値
NLength	*LevelN* の枝の長さ
NCurveRes	*LevelN* の枝のノード数
NRotate	*LevelN* の枝の分枝角度（Y軸）
NDownAngle	*LevelN* の枝の分枝角度（X軸）
NBranches	*LevelN* の枝の総数

この手法では樹木形状を *Level* という階層的な概念に基づいて生成する。通常の場合、幹を *Level*0 として定義し、幹から生える枝を *Level*1 、*Level*1 の枝から

生える枝を $Level2$ として分枝が進むごとに階層が深くなる。枝や葉は各 Level ごとに定義したパラメータを元に生成するが、$Level3$ 以降は全て $Level3$ のパラメータを用いて生成する。

パラメータの中には樹木の個性を実現するために補正値が用意されているものもある。通常のパラメータが樹種ごとの特徴を実現することに対して、補正値は樹木一本一本の個性を実現することに役立つ。例えば $Level1$ の枝の長さを示す $1Length$ には $1LengthV$ 、樹木の大きさを示す $Scale$ には $ScaleV$ というように、補正値を表すパラメータは末尾に V がつく。実際には $1Length \pm 1LengthV$ のように用いられ、同じ樹種であっても一本一本の個性を表現できる方法となっている。

2.2 幹・枝の生成

幹や枝の大きさは基本的に表2.1のような $LevelN$ の長さや半径のパラメータから生成し、曲がり方はノード（Node）と呼ばれる関節ごとに向きを変えることで実現している。このノードの数は $NCurveRes$ というパラメータから決定する。図2.1はその大まかな設計図である。まず $Level0$ である幹の全長（$length0$）は $Scale$ と $0Length$ によって算出する。$Level1$ 以降の枝に関してはパラメータに加えて、それぞれの親枝の長さや半径などを元にして生成し、分枝における親子関係のバランスを保っている。ここで親枝とは生成対象の枝が生えてくる枝や幹のことを指す。一方、$Level1$ の枝の長さ（$length1$）は樹木の大まかなシルエットを決定付けるとして、$offset$ と $Shape$ いう値を用いて決まる。$offset$ は親枝の全長を 1 として、枝の発生する位置から親枝の先端までの長さを示す。また $Shape$ はパラメータの 1 つで 0〜8 までの整数の番号を取る。この $offset$ と $Shape$ を ShapeRatio という関数の引数として用いることで、$Shape$ の示す番号ごとに用意した特徴的な枝ぶりを実現する。

分枝角度には $DownAngle$ で定義する X 軸方向への回転角度と、$Rotate$ で定義する Y 軸方向への回転角度がある。これらは枝が発生する親枝のノードのロー

カル座標系上で適用する。図 2.2は樹木形状を上方から見た図である。親枝から複数の枝が発生する場合、$Rotate$ は相対的に適用する。例えば $LevelN$ の $NRotate$ が 30 度とした場合、親枝から生える 2 本目の枝は 60 度回転した位置から発生することになる。

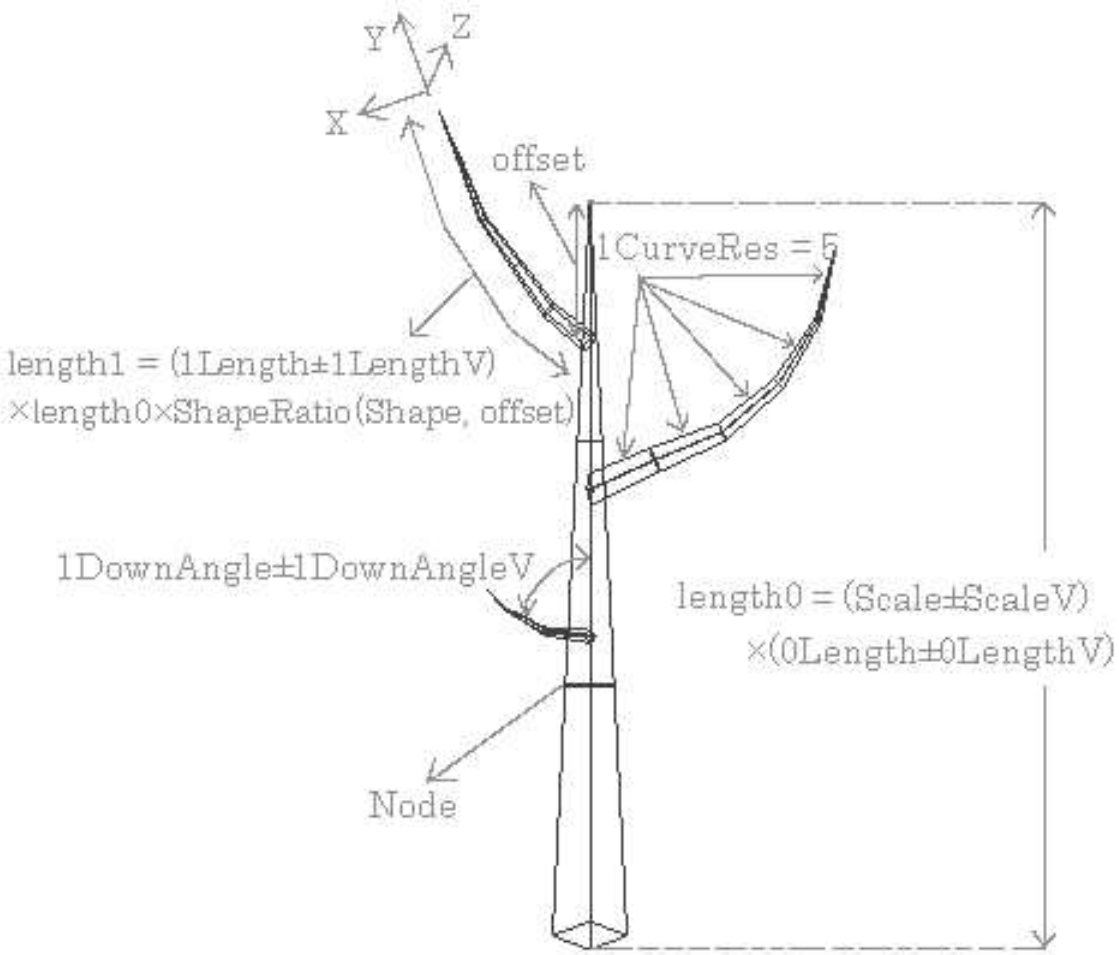

図 2.1: 幹と枝の構成の概要

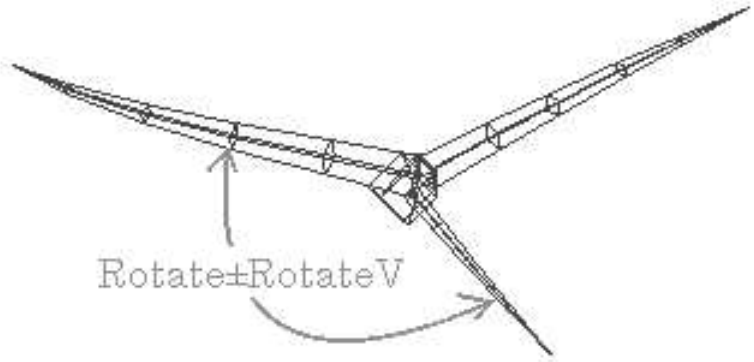

図 2.2: 樹木形状の上面図

第 3 章

コブ形状の生成

3.1 コブとは

自然界における樹木の多くには図 3.1のようにコブと呼ばれる塊や隆起を枝や幹に見ることができる。このコブは大小さまざまであり、その形状も単純なものから複雑なものまで存在し、樹木の形を特徴付ける重要な要素となっている。しかしその反面、コブの発生要因が様々であることや複雑な形状の実現が困難であることから、最小限の精度で生成される際には省かれてきた。そこで本研究ではコブの発生を柔軟な形状生成が行える樹種ごとにパラメータを設ける手法を採用し、個性的なコブ形状を近似的に実現する手法を提案する。

Peter Thomas[13] によるとコブの発生には大きく分けて二つの原因がある。一つは昆虫やバクテリア、ウィルスなどの外的要因による奇形生長である。これは一度コブの生成が始まるとその上に次々と樹皮が覆っていくことや、ウィルスが転移することが原因で年々その大きさを増してゆく癌に似た現象である。もう一つのタイプは幹や枝から生える芽が樹皮の中に埋もれてしまうことで形成される内的要因による奇形生長である。樹木は生長過程の中で休眠芽と呼ばれる芽を生成する。この芽は正常な形で形成されるが、他の芽に比べてほとんど生長をせず樹皮の表面かその直下にとどまっており、分枝をすることで複雑に肥大していき、コブを形成していく原因となる。

本研究では環境から受ける不確定要素が大きい外的要因における奇形生長ではなく、樹木の生長過程から比較的に予想が可能である内的要因、つまり休眠芽による奇形生長をターゲットとしてコブ形状の生成手法を提案する。これにより従来の樹木形状の生成手法においてもコブの発生処理を追加するだけでコブ形状を実現できると同時に、発生要因を最小限に用いることで計算コストを抑えることができる。

図 3.1: 樹木の所々で見られる特徴的なコブ

3.2　休眠芽と休眠打破

休眠芽は樹木の保険のようなものである。例え健康で正常な生長を続けている樹木であっても休眠芽は存在し、コブは発生する。そして休眠芽の存在する幹から生えている枝が損傷すると生長機能の低下が起きるため、休眠芽は急激に生長して新しい枝となる。これは休眠打破と呼ばれ、枝の機能が低下することで起こる場合と、機能低下が起きなかった古い幹や枝がある程度生長することで発芽する場合がある。図 3.2の樹木は後者の原因によって休眠打破して発芽し、生長した

枝である。本研究では休眠芽の生長をコブ発生の要素として定義し、休眠打破などの休眠芽の機能を考慮した形状生成を行う。またコブは複数の休眠芽が集まって形成される場合もあるが、本研究では一つのコブから休眠打破して生長する休眠芽は一つだけとする。

図 3.2: 休眠打破されたコブの中にある休眠芽

3.3 コブ形状の実装

3.3.1 パラメータの定義

コブを生成する幾つかのパラメータを定義する。本手法ではコブを内的要因である休眠芽として生成するため、その発生要因と変化に着目し、以下の 5 つのパラメータをコブ形状生成の要素と定義する。

- 発生率

  発生率はコブの親枝が固有に持つ値で発生頻度を決定付け、コブの発生とともに減少していく。本研究ではコブの発生率はそのまま休眠芽の発生率を指し、樹木がどれだけ保険をかけるかを示すことになる。この発生率と乱数を用いることで枝がコブになるかどうかを判定するが、樹木に生える枝数はあらかじめ決めて生成する事が多く、その枝数にはコブの発生を考慮していない。そこで枝数を算出する際にコブの発生率をあらかじめ用いて計算することで不自然にコブが目立つ問題を避けることができる。またこの発生率は一本一本の親枝に対して同じ値が初期値として適用されるもので、樹木全体のコブの発生率を示すものではない。

- 発生位置

  発生位置はコブが発生を始める位置を示し、0 から 1 の値をとる。0 が最も親枝の根元に近く、1 が先端に近い位置にコブが発生する傾向を示す。コブの発生位置は樹種によって様々であり、幹の上方に拡散するものや根の付近に集団で発生するものもある。そのため発生位置を樹種ごとに定義することでコブの発生を制御し、樹種の特徴を考慮した形状生成を実現する。特に幹から生成される枝の位置は樹木のシルエットを決定付ける重要な要素のため、Weber らは $BaseSize$ という値で枝の生え始める位置を樹種ごとに定義している。本研究では $BaseSize$ とコブの発生位置を考慮し、コブの発生位置が $BaseSize$ より小さい、つまりコブの方が枝より下に生えるような場合は枝を $BaseSize$ の位置まで全てコブに変形させる、という拡張を行った。これにより幹の下方に分布するコブの集団を実現することが可能となった。

- 生長抑制率

  生長抑制率はコブの長さを決める値で、本来生える予定であった枝の長さ

を縮小させる値であり、値が大きいほどコブ形状が短くなる。コブの中にある休眠芽は完全に活動を休止させているわけではなく、生長の速度が他の芽に比べて遅いだけである。そのため休眠芽は少しづつ伸びていき、やがてコブや隆起となって樹皮の表面に現れる。そこで本研究では生長抑制率という生長を抑える値を用いることで、コブの長さを枝の生長が抑制された結果として表現する。

- 密度

　密度はコブの太さを決定付ける値で、本来生える予定であった枝の太さの補正値であり、値が大きいほどコブ形状が太くなる。この値の本質はコブを形成する休眠芽の量を擬似的に決定付けるものである。コブを形成する休眠芽は 1 つとは限らず複数の休眠芽が集まることで形成されることもあり、その場合隆起はより大きく現れる。そこで擬似的に休眠芽の量を決定することによって、コブの太さを近似的に表現する。

- 休眠打破値

　休眠打破値はコブの中にある休眠芽が再び生長を始めるまでの時間である。樹木形状は樹木を生長させた、もしくはその生長過程の形として生成するものであるため、幹や枝、葉と同様にコブもある程度生長させた形を実現することが必然となる。そこでコブの中にある休眠芽の休眠打破を樹木形状の生成の過程で擬似的に起こすことで、自然のコブに似た形状生成が可能となる。その際休眠打破を起こすきっかけの 1 つとしてコブを生成してからの時間経過がある。そこで本研究ではこれを分枝深度が休眠打破値を超える時として定義し、親枝に属するコブ全てから一斉に発芽する。

前述したように本研究ではコブを枝の奇形生長の一つとして生成するため、これらコブのパラメータは枝の生成に必要なパラメータの補正値として定義している。このように定義したパラメータを樹種によって異なる値を持たせることで、特徴的なコブの形成を実現できる。また同じ樹種であっても一本一本の樹木に個性を持たせるために、それぞれのパラメータに補正値を定義して用いる。例えばコブの発生率を $KnotRate$、発生率の補正値を $KnotRateV$、0〜1 までの乱数を $Random$ とした場合、コブの発生率は $KnotRate \pm (KnotRateV \times Random)$ となり、樹種の特徴を保持しつつ樹木ごとの個性を実現する。

3.3.2 コブ形状の生成手法

ここではコブ形状の生成手法を説明する。コブ形状の生成は Weber らの手法に沿って枝形状を生成し、その際に前途したパラメータを用いて枝形状をコブ形状に変形する。図 3.3はコブ形状生成の工程である。

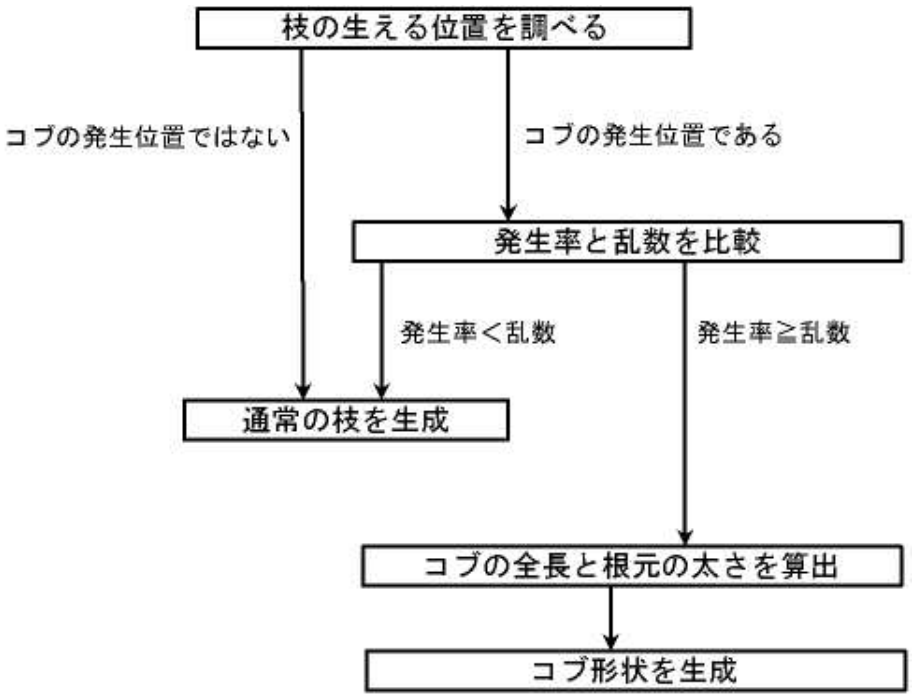

図 3.3: コブ形状生成の工程

以上の工程を全ての枝形状の生成時に行うことで、コブ形状を実現した。具体的な各工程の内容は次の通りである。

1. 枝の生える位置を求め、そこがコブの発生位置の範囲内かどうかを調べる

   枝の生える位置を求める前に、3.3.1 節のパラメータの定義の発生位置の項で説明した通り、枝の生え始める位置とコブの生え始める位置を比較し、どちらが親枝の根元に近いかを決定する。その上で枝の生える位置を算出し、その位置がコブの生え始める位置よりも親枝の先端に近い位置、つまりコブの発生範囲内であるかどうかを調べる。

2. コブの発生範囲内であれば、乱数と発生率を用いコブがて発生するかどうかを判定する

枝の発生位置がコブの発生範囲内であれば、乱数と発生値を用いてコブが発生するかどうかを判定する。これによりコブの発生範囲内であっても枝の全てをコブに変形させることなく、樹木特有のコブの発生量を保った形状生成が行える。

3. コブが発生する場合には、コブの全長と根元の太さを算出する

まず Weber らの手法を元に本来生える枝の長さと太さを求める。コブの長さ（$length_{\mathrm{knot}}$）を生長抑制率（$Contraction$）、生長抑制率の補正値（$ContractionV$）と 0〜1 までの乱数（$Random$）、本来生える枝の長さ（$length_{\mathrm{branche}}$）とコブの生える位置（$offset$）を用いて、(3.1) 式から求める。

$$length_{\mathrm{knot}} = \frac{length_{\mathrm{branche}}}{Contraction \pm (ContractionV \times Random)}(1 - offset) \quad (3.1)$$

ここで $offset$ は 0 に近いほど親枝の根元付近にコブが生えることを意味する。よって (3.1) 式はコブが親枝の先端付近に発生するよりも根元付近で発生した方が長くなることを示す。

また、コブの根元の太さ（$radius_{\mathrm{knot}}$）を密度（$Density$）、密度の補正値（$DensityV$）、0〜1 までの乱数（$Random$）と本来生えるべき枝の太さ（$radius_{\mathrm{branche}}$）を用いて、(3.2) 式 から求める。

$$radius_{\mathrm{knot}} = radius_{\mathrm{branche}} \times Density \pm (DensityV \times Random) \quad (3.2)$$

4. コブ形状を生成する

コブのノード数と全長と根元の太さを元に、先端にいくにつれて縮小するような形状生成をする。生成方法は Weber らの枝生成の手法を適用する。そし

てこのコブには固有の休眠打破値を与える。

3.3.3 休眠打破の実現

生成したコブが休眠打破するかどうかを判定し、実現する。図3.4はその工程である。

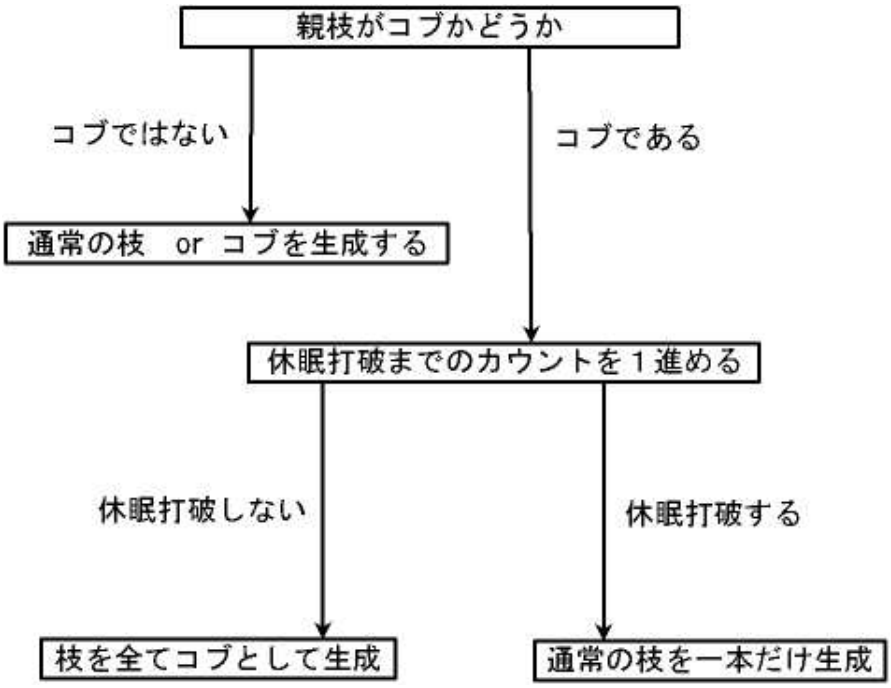

図 3.4: 休眠打破の工程

具体的な各工程の内容は次の通りである。

1. 処理対象の枝の親枝がコブかどうか判定する

   処理対象の枝の親枝が通常の枝かコブかを判定する。これによって生成する枝が休眠芽かどうかを調べる。

2. 親枝がコブの場合、枝のレベルと休眠打破値を用いて休眠打破をするかどうかの判定をする

親枝がコブの場合、すなわち生成する枝が分枝した休眠芽の場合は休眠打破が起きるかどうかを判定しなければならない。本手法では休眠打破のタイミングは時間を基準とし、レベル（分枝深度）が進むたびに休眠打破値が減少する。そして休眠打破値が 0 となった時にコブは休眠打破し、急激に芽が生長する。もし休眠打破をしなかった場合は、生成した枝も新たなコブとなり、休眠打破値を引き継ぐことになる。図 3.5は休眠打破をした場合と、しなかった場合による枝の生成の違いを示している。

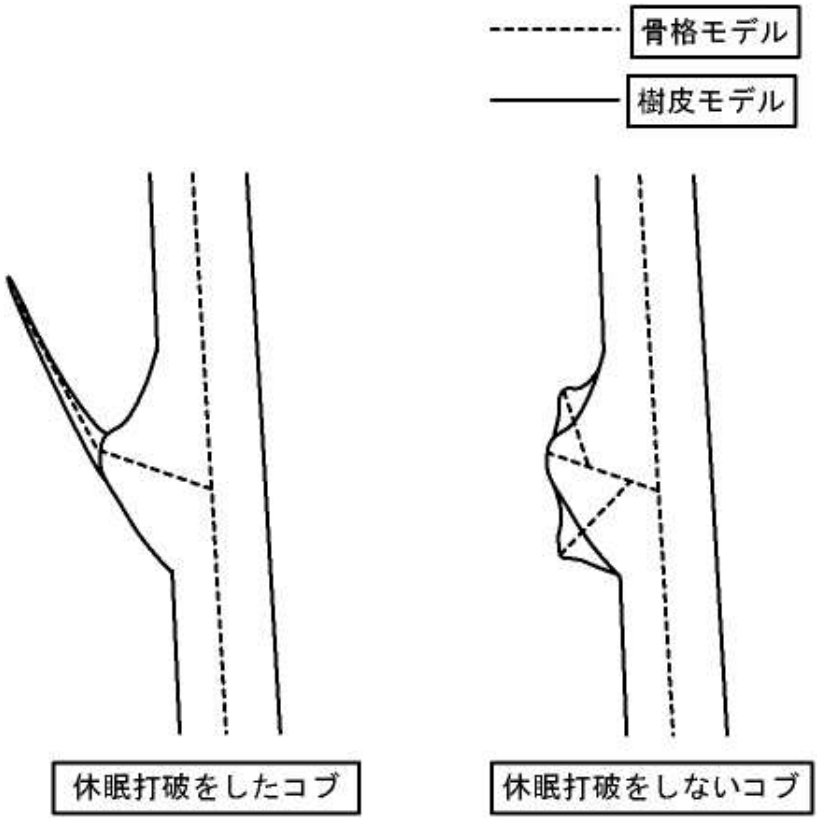

図 3.5: 休眠打破の有無による違い

3. 休眠打破する場合、枝の形状を補正する

Weber らの手法で説明したように、枝の生成にはその親枝の長さや太さといった値が反映される。生成する枝の親枝が通常の枝であれば問題ないが、コブであった場合には生長抑制値や密度によって補正していることを考慮しなければならない。これはコブから発生している枝は全てコブになってしまうことを意味する。そのため生成する枝が休眠打破したものであった場合、急激な生長を表現するためには親枝の大きさを補正する必要がある。(3.3) 式は休眠打破した枝の長さ（$length_{\mathrm{c}}$）を示す式で、親枝の長さ（$length_{\mathrm{p}}$）に生長抑制値（$Contraction$）で生長抑制値の補正値（$ContractionV$）と 0〜1 までの乱数（$Random$）補正をしていることを示す。また、$Length_{\mathrm{N}}$ は生成する枝の基準となるパラメータを示し、N は分枝深度を示す。

$$length_{\mathrm{c}} = length_{\mathrm{p}} \times (Contraction \pm (DensityV \times Random)) \times Length_{\mathrm{N}} \quad (3.3)$$

同様に枝の太さも補正を行なう。(3.4) 式は休眠打破した枝の太さ（$radius_{\mathrm{c}}$）を求めるもので、親枝の太さ（$radius_{\mathrm{p}}$）、長さ（$length_{\mathrm{p}}$）、生成する枝の長さ（$length_{\mathrm{c}}$）、密度（$Density$）、密度の補正値（$DensityV$）、0〜1 までの乱数（$Random$）と樹木全体の太さの補正値（$RatioPower$）を用いて実現する。

$$radius_{\mathrm{c}} = \left(\frac{radius_{\mathrm{p}}}{Density \pm (DensityV \times Random)} \right) \times \left(\frac{length_{\mathrm{c}}}{length_{\mathrm{p}}} \right)^{RatioPower} \quad (3.4)$$

以上のようにコブ形状の生成を実現した。次章ではコブ形状の品質を向上させるための分枝部分の平滑化について述べる。

第 4 章

分枝部分の平滑化

4.1　分枝部分の平滑化の必要性と問題点

これまでの樹木形状の生成手法では骨格モデルの生成に重点を置くものが多く、枝としての太さを持たす工程では骨格モデルに沿って円柱などを配置する手法が一般的であった。枝の関節部については骨格モデル自体が滑らかな曲線を描くように生成することが多いが、枝と茎の接合部分についてはほとんど考慮していないことが一般的である。これは樹木形状が一本一本の独立した枝を継ぎ足していった結果として生成するためであり、その結果連続性に欠ける形状を生成する。しかし現実世界における樹木は図 4.1のように幹から生える枝は滑らかに接続しており、写実的なモデリングをする上では考慮をしなければならない。

図 4.1: 滑らかに接合された分枝部分

分枝部分の不自然さを解決する手法として大志田ら [14] の樹木形状の肥大生長に関する研究がある。これは樹木形状を分枝部分などの部位にかかるストレスをもとに膨張させ、自然な肥大生長を実現する手法である。具体的な方法は樹木の骨格モデルを生成し、それに太さを持たせた仮の樹木モデルを生成する。そしてこのモデルを 3 次元のボクセル空間に配置して各ボックスにおけるモデルの存在量を調べ、ボックス間の質点－ばねモデルを生成する。ばねモデルを用いて各質点での圧縮もしくは引張を調べ、これをもとにボクセル空間に配置してある仮の樹木モデルを肥大化する手法である。この手法は分枝部分の平滑化には適しており高品質な樹木形状を生成できるが、生成工程が長い点から非常に処理コストが高く、特に複雑な樹木形状の場合にはとても効率が悪い。また樹木形状全体を再構築するという目的に対しては本来不要な処理を行なっている。

本研究ではこれらの問題点を踏まえ、高品質の樹木形状を生成するためにクオリティと、処理工程を重視した最小限の生成という面を両立させるため、分枝部分のみに対して局所的に平滑化を行う手法を提案する。

4.2　分枝部分を生成する際に考慮すべき点

4.2.1　接続位置と平滑化の適用範囲

分枝部分の平滑化を適用する個所は、図 4.2のように生成する枝の根元から根元に最も近い位置にある第 1 ノードまでとする。これによって親枝を複雑に変形させる必要がなく滑らかに接合でき、生成する枝のみを意識するだけで高品質の形状生成が実現できる。

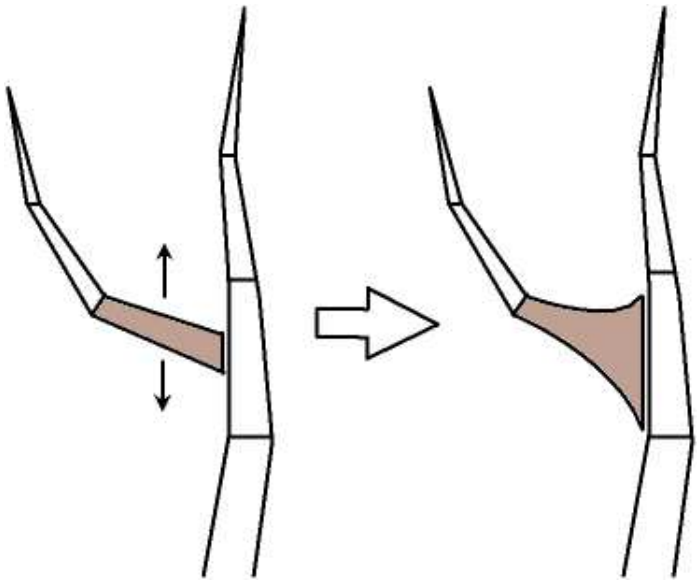

図 4.2: 平滑化の適用範囲

分枝部分の平滑化を行なう際に、滑らかに接続するように生成する枝の根元の位置を決定する必要がある。これについて本手法では生成する枝の親枝の先端を上、根元を下として、上下に位置する他の枝の根元や親枝のノードとの滑らかな接続を目的とし、それらの座標値を用いた。それにより単純に枝の根元を拡大させた位置に接続点を設けることによって生じる、接続点が親枝上に存在しないといった不自然さなどを避けることができる。

具体的な接続点の決定方法は図 4.3に示す通りである。これは生成する枝の上側と下側のそれぞれに最も近い他の枝や親ノードの位置を求め、親枝との接続点と

して用いる方法である。例えば枝 A の上側で最も近いのが枝 B である場合、枝 A の上側の接続点は枝 B の骨格モデルの根元と同じ高さにあたる点となる。ここで高さとは親枝における上下の位置だけを指す。また、枝 A の下側で最も近いのが親枝のノードである場合は、枝 A の下側の接続点は親枝のノードと同じ高さにあたる点となる。また生成する枝の根元が親枝のノードと重なる場合は、その親枝のノードを無視して生成する枝の上下に位置するノードを適用する。

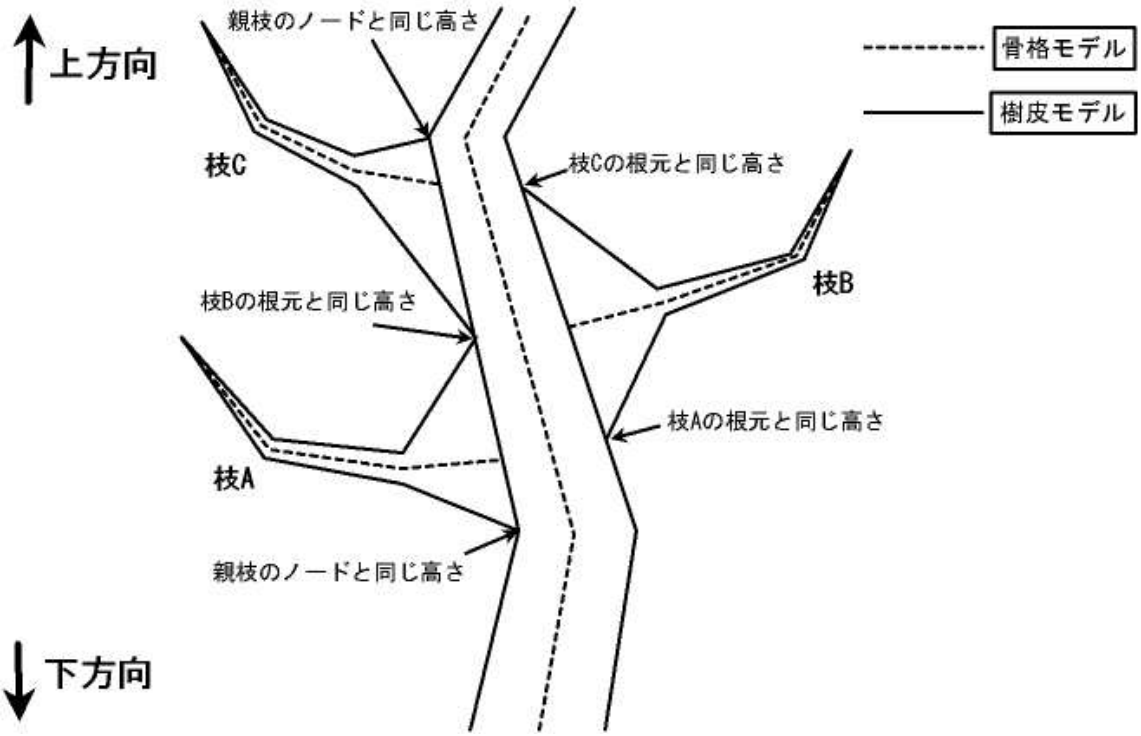

図 4.3: 本研究で定義する枝の接続図

4.2.2 引張あて材と圧縮あて材

Thomas によると樹木は地面の傾斜や風などの外力によって幹が傾くと、あて材と呼ばれる補強および姿勢の補正のための肥大生長が起こりまっすぐに立て直そうとする。これはほんのわずかな例外があるものの針葉樹と広葉樹に大別した場合、それぞれ反対の方法をとっている。

傾いている針葉樹の場合、あて材は幹を支えるように発生する。これは幹の下側部分に対して圧縮するようなあて材が施されることを意味する。このようなあて材を圧縮あて材と呼び、幹を押しあげるように膨らむことからこのように呼ばれる。一方、傾いている広葉樹の場合は、あて材は幹を引っ張るように発生する。傾いた幹の上側部分に対して幹を引張するようなあて材が施される。このようなあて材を引張あて材と呼び、発生すると徐々に縮み、幹は傾く方向と逆に引っ張られて姿勢を正される。

注目すべき点はこのようなあて材は幹の根元のみならず、分枝部分にも発生することである。枝の生長に伴って枝が重くなったり、その枝から枝や葉が発生することで全体の重量が増すと、分枝角度を正しく保つためにあて材が発生する。しかしこのようなあて材がどのような機構でコントロールされているのかは判明していない。そこで本手法では圧縮あて材と引張あて材の存在のみに着目し、樹種ごとにそれらの量をパラメータとして定義することで分枝部分を近似的に精巧に表現する。

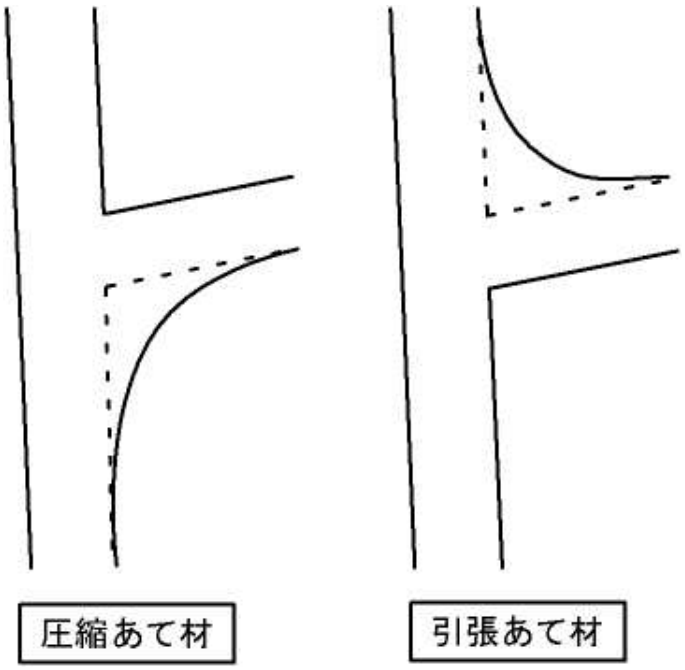

図 4.4: 引張あて材と圧縮あて材による肥大の違い

4.2.3　分割数と品質

分枝部分の平滑化は樹木形状の品質を向上させる反面、生成コストの増大を招く。この問題に本研究では 2 通りの解決策を提案する。1 つめはユーザによる分枝部分の品質決定である。この方法はユーザが樹木形状を生成する際に図 4.5のように分枝部分の分割数を決定することで品質とコストを調整することが可能となる。2 つめは分枝深度によって品質を下げる方法である。これはユーザの決定した分割数を分枝深度が進むごとに減少させる方法である。ほとんど全ての樹木は分枝深度が進むほど細い枝を大量に生成するが、その分枝部分を支えるあて材の量は減少する。これらを考慮して分枝深度が進むごとに分枝部分の分割数を減少させることで、枝の生成量の増加に伴って膨大に増える計算コストを削減できるとともに、必要最小限のクオリティを保つことも実現できる。

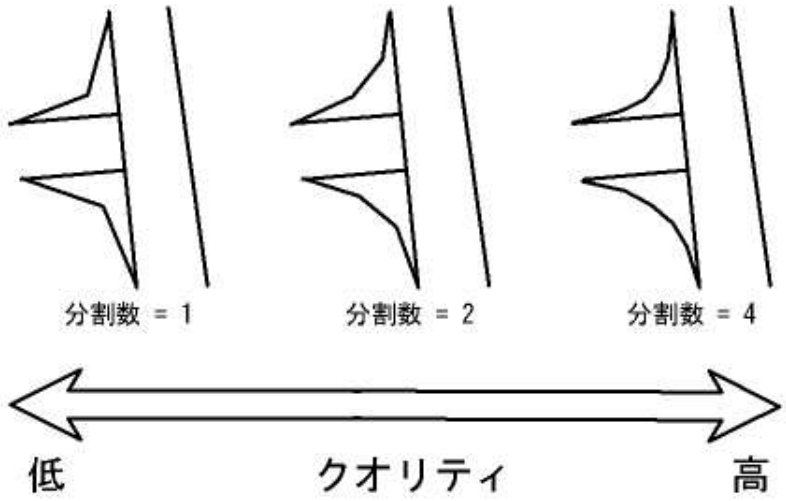

図 4.5: 分割数とクオリティ

4.3　分枝部分の平滑化の実装

分枝部分の平滑化は生成する枝の根元から第 1 ノードまでの間に適用するが、この適用部分はさらに親枝の先端方向と根元方向で生成処理を分ける。これは分枝

部分に発生する圧縮あて材と引張あて材をそれぞれのパラメータを用いて実現することや、分枝部分の上下に位置するそれぞれの接続点を考慮するためである。この点を踏まえ本手法では図 4.6のように分枝部分の平滑化のための制御点を 3 個所用いて、平滑化した分枝部分を構成する頂点を算出する。1 つめの制御点 **P** は親枝との接続点を示す。2 つめは生成する枝の第 1 ノードを示す **C** という制御点である。3 つめは生成する枝と親枝との本来の接点を示す **B** という制御点である。この 3 つの制御点を用いて図 4.6のように **P** と **C** を結ぶ曲線上に分枝部分の分割数だけ頂点を生成する。また、これら 3 つの制御点は枝を表現する多角柱の角数分だけ必要となり、角数分だけ新しい頂点を算出する。

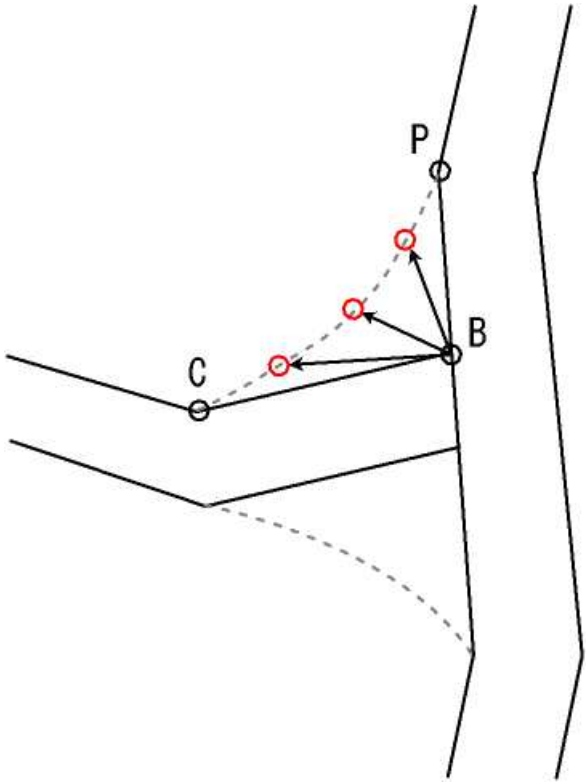

図 4.6: 分割数が 3 の場合

4.2.3 項で前述したように本手法は分枝部分の生成コストを抑えるために、分枝深度が進むにつれて分割数を減少させる。これを本手法では (4.1) 式で実現し、分枝部分の分割数（D）を分枝深度（$level$）が進むにつれてユーザの定義した分枝

部分の分割数（D_u）を減少させた。

$$D = \frac{D_u}{level} \tag{4.1}$$

分枝部分に発生するあて材の量（F）は頂点を生成する個所によって用いるパラメータが異なる。これは生成する枝の骨格モデルを中心に、上側では引張あて材（$Stretch$）を施すため $F = Stretch$ として計算する。一方、下側では圧縮あて材（$Compress$）を施すため $F = Compress$ として計算する。

以上の点を踏まえ、(4.2) 式を用いて頂点 $\mathbf{N}_{d,m}$ を求めることにする。

$$\mathbf{N}_{d,m} = \left(\left(1 - \frac{d}{D}\right)^F (\mathbf{P}_{d,m} - \mathbf{B}_{d,m}) + \left(\frac{d}{D}\right)^F (\mathbf{C}_{d,m} - \mathbf{B}_{d,m}) \right) \left(\frac{length_c \times radius_c}{length_p \times radius_p} \right) + \mathbf{B}_{d,m} \tag{4.2}$$

$$d = 1, 2, 3, ..., D$$

$$m = 1, 2, 3, ..., M$$

(4.2) 式は制御点 $\mathbf{P}$、$\mathbf{C}$、$\mathbf{B}$ と親枝の太さ（$radius_p$）と長さ（$length_p$）、生成する枝の長さ（$length_c$）と太さ（$radius_c$）を用いて、枝を表現する多角柱の角数（M）だけ算出する。そしてこれらを分枝部分の分割数（D）だけ求めることで分枝部分を構成する全ての新しい頂点を算出する。

以上のように求めた $\mathbf{N}_{d,m}$ を用いて分枝部分の形状を再構築することで平滑化を実現した。

第 5 章

結果と考察

5.1 結果

3 章、4 章でそれぞれ述べたコブ形状と分枝部分の平滑化の実現手法を、3D グラフィックツールキットである FK System[15] を用いて実装を行なった。

図 5.1は 3 章で述べたパラメータと手法を用いて生成したコブ形状である。分枝部分の平滑化を調整することで、樹皮から突出した塊状のコブや滑らかな隆起状のコブを実現することができた。

図 5.1: コブ形状の実現

図 5.2は休眠打破を実現した図である。休眠打破したコブから細い枝が急激に伸びて、その枝がさらに枝分かれをして葉を生やしている様子が確認できた。

図 5.2: 休眠打破の実現

4 章で提案した分枝部分の平滑化手法について実装した。図 5.3はあて材について検証した図である。左右の画像は同じ樹種の樹木形状で、圧縮あて材と引張あて材を施すパラメータの圧縮と引張の値を反対にした実行結果である。図 5.3から枝の上下に施す圧縮あて材と引張あて材を別々の値を用いて実現でき、樹種ごとの特徴を考慮してパラメータを自由に設定できることが確認できた。

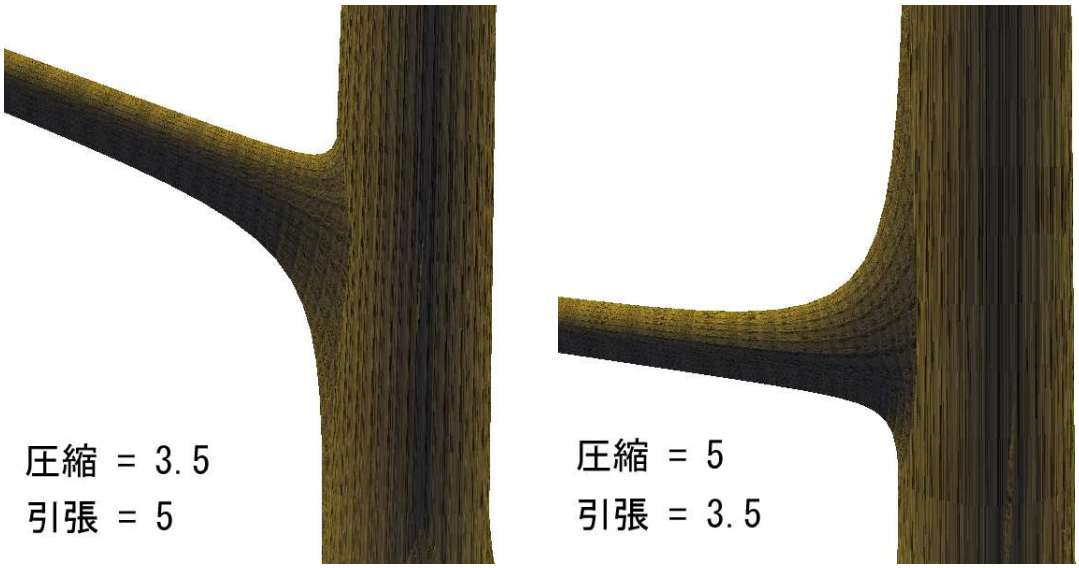

図 5.3: 分枝部分の平滑化の実現

図 5.4は分枝部分の分割数を変更して実行した図である。図 5.4で確認できるようにユーザが指定した分割数で形状生成が可能で、要求される精度と生成コストを考慮した分枝部分の平滑化が実現できた。

図 5.4: 分割数の指定による精度の違い

5.2　考察

本研究ではコブ形状生成のためのパラメータとその生成手法を提案した結果、樹木形状を特徴付けるコブ形状が表現できた。またコブ形状の精度を向上させるため、不自然さが顕著に表れる分枝部分に対して任意の品質で平滑化を行なう手法を実現することができた。これらの点から精度と生成コストを考慮した上で、個性的な隆起形状や図 5.5のような個性的なコブ形状を含む写実的な樹木形状の生成を実現することができ、目的を達することができたと言える。

図 5.5: コブ形状を含む写実的な樹木形状コブ形状を含む写実的な樹木

今後さらに写実的なコブ形状を生成するには課題がある。本研究ではコブの発生要因を休眠芽のみと仮定して実現したが、3 章で述べたように自然界においてコブの発生要因は他にも存在する。これは昆虫やバクテリア、ウィルスなど樹木を取り巻く環境が要因となって発生するもので、これらが原因でコブが発生すると休眠芽で発生したコブとはまた違った形となる。そのためコブ形状の精度を向上させると同時に、自然界に則したコブの発生を実現させるためには害虫やウィルスによる影響などの環境要因を考慮したコブ形状の生成を行うことが必要となる。

また、分枝部分の平滑化においても一考の余地がある。本研究では分枝部分の

平滑化をユーザが指定した分割数を基に分枝部分を分割して実現した。しかし分割によって生成された新しい頂点間の角度が均等ではないため、分割数が少ないと滑らかさにバラツキが生じる。そのため精度を上げるためには分割数を増やして、分枝部分全体をさらに分割する必要がある。この点を改善するには角度が一定基準以上の箇所のみに対して分割処理を局所的に行う手法を実現する必要がある。

5.3 まとめ

本手法ではコブ形状生成のためのパラメータを定義することで枝形状の生成手法を利用し、その結果個性的なコブ形状の生成を実現できた。これは樹木形状の生成のためにパラメータを用いる全ての手法において適用できる手法であり、非常に汎用的である。また精度と処理コストを念頭にパラメータ設定を行うことで、分枝部分を任意の品質で平滑化できた。

本研究で提案する手法はある程度の精度の樹木形状を自動的に生成でき、樹木形状を重視する図5.6のようなCGシーンにおいて非常に有効な手段となる。また高精度の樹木形状をモデリングする際には、本手法で生成した樹木形状を用いることでモデリング作業の手間を軽減できる。

今後の課題として本手法で採用しなかった環境から受けるコブの発生要因を用いた手法の実現や、分枝部分の平滑化処理において新しく生成する頂点間の角度を考慮し、分割数を抑えつつ精度を上げる手法を実現したい。

図 5.6: 本研究を用いて生成した CG

謝辞

本研究を進めるにあたり、温かいご支援、ご指導いただきました東京工科大学メディア学部の渡辺大地講師および電気通信大学の和田篤氏に心より感謝いたします。

また日ごろから本研究のサポートをしていただいた、研究室のメンバーに厚く御礼申し上げます。

本研究にご協力していただいたすべての皆様に心から感謝いたします。

参考文献

[1] 桑原教彰, 鉄谷信二, 志和新一, 岸野文郎, “ フラクタルを用いた階層的な樹木形状表現による 3 次元樹木画像の高速生成方法 ”, 電子情報通信学会論文誌, D-II, Vol.J78-DII, No.7, pp.1091-1104, 1995.

[2] R.Mech and P.Prusinkiewicz, “ Visual Models of Plants Interacting with Their Environment ”, SIGGRAPH 96 Computer Graphics Proceedings, pp.397-410, 1996.

[3] 金山知俊, 阪田省二郎, 増山繁, ”分枝規則を再現し，光，ホルモンの影響を考慮した 樹木の生長モデル”, 電子情報通信学会論文誌, D-II, vol.J79-D-II, no.8, pp.1362-1373, 1997 .

[4] 金山 知俊, ”コンピュータグラフィックスによる樹木画像生成に関する研究” http://www.smlab.tutkie.tut.ac.jp/research/paper/DT/kanayama/.

[5] 岡部誠, 五十嵐健夫, ”しだれ柳の効率的な幾何モデリング”, 情報処理学会研究報告, 「グラフィクスと CAD」, No.110-4, pp.19-24, 2002.

[6] 金丸 直義, 千葉則茂, 高橋 清明，斎藤 伸自, ”向日性による樹木の自然な枝振りのＣＧシミュレーション”, 電子情報通信学会論文誌, D-II, Vol.J75-D-II, No.1, pp.76-85, 1992.

[7] 岡部誠, 五十嵐健夫, ”手書きスケッチに基づく樹木の３次元モデリング”, 情報処理学会研究報告, 「グラフィクスと CAD」, No.112-8, pp.41-46, 2003.

[8] 桑原教彰, 志和新一, 岸野文郎, 新井民夫, “ 樹木画像を入力とする 3 次元樹木形状のフラクタルモデルの自動推定方法についての評価 ”, 画像電子学会誌, Vol.25, No.1, pp.45-53, 1996.

[9] 坂口竜己, 大谷淳, 中津良平, ”実写映像に基づいた三次元樹木モデルの生成”, 電子情報通信学会論文誌, D-II, vol.J82-D-II, no.9, pp.1469-1477, 1999.

[10] 千葉則茂, 大川俊一, 村岡一信, 三浦守, ”CG のための樹木の生長モデル―架空の「植物ホルモン」による自然な樹形の生成―”, 電子情報通信学会論文誌, D- , vol.J76-D- , No.8, pp.1722-1732, 1993.

[11] J.WEBER, and J.PENN, ”Creation and rendering of realistic trees”, SIGGRAPH 95 Computer Graphics Proceedings, pp.119-128, 1995.

[12] Jason Weber, http://www.imonk.com/jason/.

[13] Peter Thomas, ”樹木学”, 築地書館, 2001.

[14] 大志田憲, 村上一信, 千葉則茂, ”樹木 CG のための肥大生長シミュレーション”, 情報処理学会研究報告, 「グラフィクスと CAD」, No.098-4, pp.19-24, 1999.

[15] FK Toolkit System, http://www.teu.ac.jp/media/˜earth/FK/.