芸術科学会論文誌 Vol.3 No.1 pp.76－85

風による樹木の揺らぎの効果音の効率的な生成法

松山克胤† 藤本忠博† 村岡一信‡ 千葉則茂†
†岩手大学 ‡東北工業大学

アブストラクト

筆者らは，これまで風による樹木の揺らぎの効果音を自動的に付加するサウンドモデリング法を提案している．この提案手法は，樹木の枝と葉に対して，揺らぎの付加と効果音の生成をそれぞれ独立に行い，個々に作成された効果音を合成することで，樹木全体の効果音としている．本論文では，枝と葉の効果音の生成に対して，既提案手法とは異なる統計的なアプローチに基づくより効率的に動作する手法を提案する．

Efficient Generation of Sound Effects of Trees Swaying in the Wind

Katsutsugu MATSUYAMA† Tadahiro FUJIMOTO† Kazunobu MURAOKA‡ Norishige CHIBA†
†Iwate University ‡Tohoku Institute of Technology

Abstract

We proposed the sound modeling method that automatically produces sound effects of a tree swaying in the wind. In the method, we first apply the different sound effect generation algorithms to branches and leaves. The individual results are then compounded into the whole sound effect of the tree. In this paper, we will present the more efficient generation algorithms for branches and leaves developed by employing the statistical strategies different from the previous ones.

キーワード：アニメーション，自然現象，樹木の揺らぎ，サウンドモデリング，サウンドレンダリング
Keywords: Animation, Natural Phenomena, Swaying Trees, Sound Modeling, Sound Rendering

１．はじめに

効果音はアニメーション，映画，ゲーム等で広く使用されており，映像コンテンツには重要な要素として認知されている．効果音作成の一般的な工程は，サウンドデザイナーが映像を基に手作業で作成・付加する場合が多く，専門的なサウンドデザイナーを必要とする場合も多い．また，バーチャルリアリティ（VR）やミックスドリアリティ（MR）といったインタラクティブコンテンツでは，キューとなる映像が動的に決定されるため，効果音の自動生成は有用である．特に，インタラクティブに動作する環境では，効果音はリアルタイムに生成する必要がある．

筆者らは，樹木の枝葉の風による揺らぎのリアルタイムアニメーションを生成するためのノイズベースの効果的な手法を開発しており[1]，このアニメーション技術に対して，さらに風による揺らぎで発生する効果音を自動的に付加するサウンドモデリング法を提案した[2, 3]．これらは，枝の効果音はカルマン渦列の周波数に基づいた手法を，葉の効果音は葉同士の位置関係から波形表を作成する手法を基本としている．本論文では，既提案手法とは異なるアプローチによる，インタラクティブに動作するサウンドモデリング法を提案する．既提案手法と本提案手法の特徴は以下のようである．既提案手法では，本提案手法と比べ計算時間をより多く必要とするが, 葉の交差判定に基づいており，モーションと一致した接触音の生成が可能となるため，近接シーンの表現にも対応可能である．また，前計算を必要としないため使用メモリを増加させることなく，多種多様な樹木からなるシーンのウォークスルーなども可能となる．一方，今回の提案手法では，交差判定を統計的に行うことにより，インタラクティブな応用を可能としているため，特に遠景のシーンの表現に向く．

一般に，音楽ＣＤレベルの高品質な音はサンプリングレートが 44100Hz 以上で成立しうるが，本論文ではリアルタイムにこれを満足する効果音を生成する手法を提案する．フレームレートは可変で，計算資源と樹木データの大きさに応じて自動的に対応させている．

本論文の構成は以下のようである．２章にコンピュータグラフィックスにおける音の研究を，本提案手法のシステム概要を３章に，枝と葉に関するサウンドモデリング法をそれぞれ４章と５章に，サウンドレンダリングを６章に，実行結果を７章に，まとめと今後の課題を８章にそれぞれ記述する．

２．関連研究

コンピュータグラフィックスの分野における効果音の自動生成に関する研究の先駆的な試みは１０年前にさかのぼり[4]，その後の研究例はほとんどなく，ここ２，３年で特に活発になってきた新しい研究分野である[5, 6, 7, 8, 9, 10, 11, 12, 13].

３次元仮想環境での音生成の技術は，サウンドレンダリングとサウンドモデリングとに大別できる[4]．サウンドレンダリングは，音の伝播シミュレーションを用いた聴覚化のことで，仮想空間内の音源と受聴点との関係を計算するものである．サウンドレンダリングに関する研究は，近年ビームトレーシング法を用いる手法が発表されており，ユーザがインタラクティブに移動できる音環境[5]や，アバター間でのサウンドレンダリング法[6], さらに回折を考慮したもの[7]が提案された．また，実際の聴覚環境の部屋を用いて，室内音響に関するキャリブレーションと一連のビームトレーシング法の有効性が示された[8].

サウンドモデリングは，物体の形状や特性を用い，その物体を音源とし，生成される波形を計算するものである．サウンドモデリングに関する研究は，物理法則に基づいた，剛体の衝突，摩擦，回転に対して効果音を自動的に付加する手法[9]が提案されている．これはインタラクティブかつ自動的に効果音を付加できる．この研究に関連して，さらに，現実の３Ｄオブジェクトの音を実測すること[14]で，自然な音を作成することも可能である．また，非線形有限要素法[15]を用いて，短いタイムステップで幾何形状の微小変化を利用して音を作成する手法[10]により，効果音の自動生成を可能とした例もある．これは物理シミュレーションに基づくため計算コストが大きいという問題点を持つ．ウェーブレット解析を静的に行い，キャラクタやテクスチャに投影することで新しい音を再構築する手法[11]も提案されている．

風による樹木の揺らぎで発生する効果音は，枝と風による音，枝同士の衝突，枝が軋む音，葉と風による音，葉同士の衝突・摩擦，枝と葉の衝突・摩擦，葉が風で飛ぶ時に鳴る音と多くあるが，本論文では，これらのうち基本的な音として，枝の効果音は枝と風による音を，葉の効果音は，葉同士の衝突・摩擦による音をとりあげ，その生成手法を提案する．筆者らは枝と風で発生する音の自動生成[2]，そして，葉の衝突・摩擦による効果音の自動生成法[3]をそれぞれ提案してきている．また，土橋等により，サウンドテクスチャーを用いた風きり音のリアルタイムレンダリング法が最近提案されている[12, 13]．この手法では，数値流体解析に Curle のモデルを適用することで，様々な形状の風きり音を表現できるが，Curle のモデルは，物体が音波長と比較して，十分小さい領域でなければ適用できないため，物体を小領域に分割し，互いに無相関な音源として扱っている．これは樹木のような規模の大きな対象に適用すると，音源の数が膨大になり，効率性の観点から問題となる．本提案手法は，枝や葉の多くの要素からなる樹木の効果音の生成に適用可能な効率的なものとするため，土橋らの手法とは異なるアプローチを採用している．

３．システム構成

音は媒質中を伝わる波であり，ここでは媒質は空気である．音には，周波数，音色，振幅という特徴があり，これらは音の３要素と呼ばれる．したがって，効果音の付加はこれらの３要素を決定することである．

本論文では，簡単のため，樹木一本を一つの音源として考える．樹木の一部または複数の樹木を一つの音源とみなすことも可能であるが，説明の簡単のため，樹木一本を仮定する．その樹木を *TREE* とし，*TREE* は，枝 $B(t)$と葉 $L(t)$で構成されていると考える．

$$TREE(t) = \{B(t), L(t)\} \quad (1)$$

$$B(t) = \{B_i(t)\}, i = 1, \ldots, NB \quad (2)$$

$$L(t) = \{L_j(t)\}, j = 1, \ldots, NL \quad (3)$$

NB: 枝の本数

NL: 葉の枚数

ここで，$B_i(t)$，$L_j(t)$は，それぞれ，個々の枝，葉の時刻 t での状態を表し，それら全体からなる集合を $B(t)$, $L(t)$ としている．時刻は $t \geq 0$ とし，t=0 の時は無風状態を示すものとする．樹木の形状モデルには，これまで提案されている樹木の形状モデルの生成に関する手法が

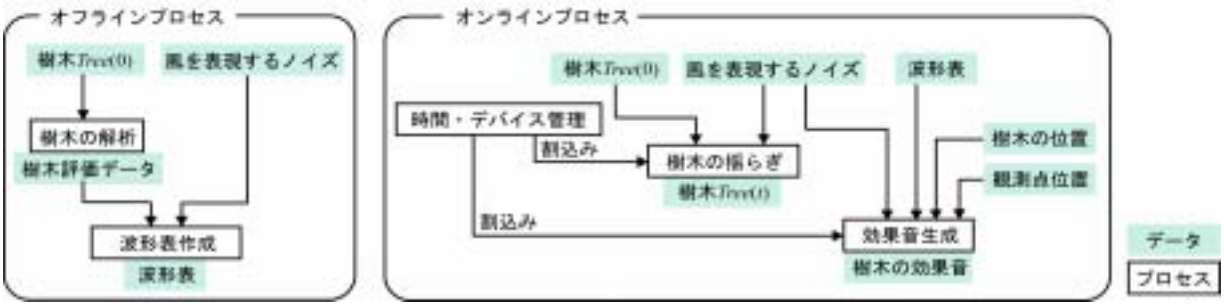

図１：システム構成図

適用できる[16, 17, 18].

本提案手法はオフラインプロセスとオンラインプロセスとに分かれている（図１）．オフラインプロセスは樹木の形状データを入力とし，解析を行うブロックと，解析されたデータを基に，波形表を作成するブロックとで構成される．オフラインプロセスで作成された波形表は，オンラインプロセスの効果音生成の入力に使用され，時間・デバイス管理ブロックからの割り込み信号で作動する効果音生成ブロックで再生される．オフラインプロセスの解析と波形表生成は，同じブロックでも枝と葉とで異なる手法をとる．なお，以下の説明文中では，波形表 *WaveTable* を，*WaveTable*()と括弧付きで記述した場合，括弧内の要素に応じた波形表を参照するものとする．

４．枝に関するサウンドモデリング

４．１　オフラインプロセス

枝の効果音は，カルマン渦の周期的な渦放出が原因で発生する音で，音の周波数は渦放出周波数と同じである[19]．流速 U の一様流中に直径 D の円柱が存在する場合に発生するカルマン渦列の周波数 f_0 は実験的に求められており，以下の式で記述される[19].

$$f_0 = StU / D \tag{4}$$

St：ストローハル数

また，f_0 の振幅 I_0 は，

$$I_0 \propto U^6 / r^2 \tag{5}$$

r：枝と観測点との距離

となる．式(4)のストローハル数 St はレイノルズ数 Re の関数であり，$4\times10^2<Re<2\times10^5$ であれば St はほぼ一定値で 0.2 であるとされているので，ストローハル数 St を 0.2 という定数として扱う[19]．また，ここでの流速 U は枝に垂直な風の流れであり，周波数 f_0 を求めるためには，各枝 $B_i(t)$と風の速度との角度を考慮し，枝 $B_i(t)$ に対する垂直成分を計算する必要がある．

枝のデータ構造は木構造であり，形状データは円柱または円錐台とする．ここで，枝 $B_i(t)$が子を持つ場合は円錐台とし，自分の直径と子の直径を使用する．先端の枝のような，子を持たない場合は円柱とする．これは式(4)，(5)を適用できる基本的な形状である．ところで，サウンドモデリングは樹木から発生する音そのものの計算を行う．最終的に受聴点で観測される音はサウンドレンダリングで計算するため，式(5)の枝と観測点との距離 r に関する計算はサウンドレンダリングで行う．したがって，サウンドモデリングの段階では，r は考慮する必要がなく，式(5)の I_0 を U の関数とみなすことができる．

式(4)，(5)で周波数，振幅が決定できる，そしてさらに音色を付加することで枝の効果音とする．音色は高周波の要素にノイズを加える．すなわち，式(4)，(5)で得られるカルマン渦列の周波数 f_0 を基底周波数とし，振幅 I_0 を用いて音色を付加する．音色には周波数スペクトルが $1/f^{\beta}$ ノイズになるように付加する．

$$I = I_0 /(f / f_0)^{\beta} \qquad \left(f_0 < f < \frac{f_s}{2}\right) \tag{6}$$

f_s：サンプリング周波数

β：音色を決定する定数

枝の効果音生成に，枝 $B_i(t)$に式(4)，(5)および式(6)を直接適用することを考える．この場合，まず，枝 $B_i(t)$ の両端について式(4)，(5)を適用して f_0 及び I_0 を求め，その後，それらを補間することで枝 $B_i(t)$上の各点での f_0 及び I_0 を計算する．そして，それらに式(6)を適用することで音色を付加する．この計算をフレーム毎に計算するというのが既提案手法[2]である．この手法は，円柱や円錐台の形状に対し，ある程度の正確さで効果音を付加することができる．しかし，フレーム毎の計算量が多いため，一般的な PC（例えば，CPU 2.53GHz

Pentium®4 程度）を用いた場合は，枝の本数が小さい場合はある程度の速さで動作するが，樹木を対象にした場合にリアルタイムで行うことは現状では困難である．そこで，以下，式(4)，(5)，(6)に基づく高速化手法を提案する．

オンラインプロセスの効果音生成に必要な時間計算量を削減するため，あらかじめオフラインプロセスで式(4)，(5)，(6)を用い，波形表 *BranchWaveTable* を作成するアプローチをとる．そのため，式(4)，(5)，(6)をオフラインプロセスに適用することを考える．式(4)は流速 U と枝の直径 D に，式(5)は流速 U に依存する数式である．ここで，前もって枝の形状を知ることができれば，式(4)の D を前もって決定でき，式(4)，(5)ともに U のみに依存する数式となる．したがって，オフラインプロセスで，流速 U を量子化し，流速 U に関する波形表 *BranchWaveTable*(U)を前処理で作成する．音源は複数の枝 B_1，B_2，…，B_{NB} で構成されているので，各枝 B_i の流速，すなわち，枝に垂直な風の成分を求めるための枝の角度が関係しているが，本提案手法は角度の情報を無視し，流速を風速で代用する．また，厳密には枝 B_i の各部位で式(4)，(5)，(6)を用い，各部位で発生する周波数，振幅，音色を求めるが，本提案手法では，部位ごとの周波数，振幅，音色の差を無視し，枝 B_i の全ての場所で同じ周波数，振幅，音色であるとする．これらの近似を行うことで，枝の長さを直接用いて枝の評価ができる（図２）．また，枝の直径 D を量子化することで前処理に必要な計算コストを減少できる．

以上の手法をまとめ，以下のように波形表 *BranchWaveTable*(U)を作成する．

(a) まず，入力された樹木 *Tree* から，枝の集合 $B(0)$ を評価し，樹木 *Tree* に対する枝の評価値を求める（図２）．

(b) 各流速 U_i について，枝の直径 D_j に式(4)，(5)，(6)を適用して波形を作成する．作成された各波形に対し，D_j の評価値で重み付けを行い，足し合わせをして *BranchWaveTable*(U_i)を作成する．

(b)をすべての U_i に対して行うことで，*BranchWaveTable*(U)を作成できる．

４．２　オンラインプロセス

オンラインプロセスは，入力された風速 U を量子化し，対応する波形表 *BranchWaveTable*(U)を再生するだけである．波形表の再生には，時刻 t にどの波形表が用いられているかを示す ID と，波形表の再生終了点を保持する．時刻$t-1$の再生終了点は時刻 t の再生開始点となり，風速が変化しない場合は，ID と再生開始点を用いることで，滑らかな波形を再生できる．風速が変化する場合は，数ステップ前に使用された波形表 ID と再生開始点を用い，複数の波形表をクロスフェードさせて再生することで，また，フェードインする波形表の再生開始点をゼロクロス点とすることで，U の変化に伴う波形表 ID の変化で混入されるノイズを減少させる．

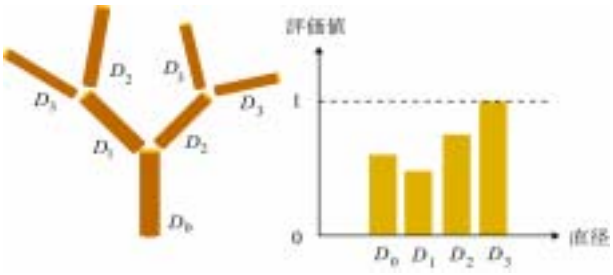

図２：枝の評価

５．葉に関するサウンドモデリング

５．１　オフラインプロセス

葉の効果音は葉身同士の衝突および摩擦によって発生するものとする．既提案手法[3]では，以下のようにして，葉の衝突・摩擦をもとに，葉の効果音生成を行っている．まず，異なる葉身の大きさを持つ葉ごとに，周囲の葉と当たることで生成される音を表すための周波数の異なる複数の波形表を作成しておく．そして，個々の葉に対して，その葉身の大きさに対応した波形表の一つを割り当てる．次に，各時刻に，個々の葉について，周囲の葉との位置関係から，５つの状態（初期状態，接触状態，非接触状態，Key on，Key off）のいずれにあるかを決定し，その状態に応じた波形表の再生を行う．なお，各状態は２枚の葉の距離をあらかじめ決めた閾値と比較することで決定される（図３）．各状態は以下のようである．

(a) 接触状態: 葉身同士が摩擦している状態．波形表を，その摩擦速度に応じて再生する．２枚の葉の距離が閾値以下なら状態は遷移しない．それ以外では Key off に遷移する．

(b) 非接触状態: 葉身同士が離れているために，何の

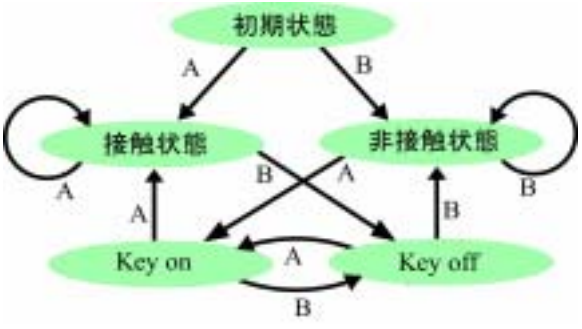

図３：葉の状態遷移図
Ａ：２枚の葉の距離≦閾値
Ｂ：２枚の葉の距離＞閾値

作用もない状態．２枚の葉の距離が閾値以下なら Key on に遷移する．それ以外では状態は遷移しない．

(c) Key on: 葉身同士が衝突している状態．衝突速度を用いて波形表を再生する．２枚の葉の距離が閾値以下なら接触状態に遷移する．それ以外では Key off に遷移する．

(d) Key off: 衝突または摩擦の状態が終了した状態．ノイズの混入を防ぐために存在する．２枚の葉の距離が閾値以下なら Key on に遷移する．それ以外では非接触状態に遷移する．

(e) 初期状態: 時刻 t=0 の時の全ての葉 $L(0)$は初期状態とする．２枚の葉の距離が閾値以下なら接触状態に遷移する．それ以外では非接触状態に遷移する．

この手法では，個々の葉の相互作用というレベルで状態の決定を行っているため，前処理で葉の隣接関係をある程度限定しているが，計算量は膨大である．

そこで，本論文では，個々の葉の相互作用を求めるのではなく，葉の状態を大局的に決定することで葉の効果音を効率的に生成する手法を提案する．本提案手法では，葉群の動きを統計的に示す「揺らぎの程度 *flicker_rate*」というパラメータを用いる．このパラメータは，葉群の揺れの度合いを示すもので，$0 \le flicker_rate \le 1$の値を持ち，その値が大きいほど揺れが激しいことを示す．*flicker_rate* をどのように決定するかについては後述する．*flicker_rate* は主に葉同士の単位時間内の衝突数と，衝突時の強さに影響する．*flicker_rate* を量子化することで前処理が可能となり，高速化を行うことができる．

前処理は大きく２つのブロックに分かれる．最初のブロックは，入力された樹木の葉を解析し，基となる波形表 *LeafBaseTable* (*leaf_size*)を作成する．２つめは，揺らぎの程度に応じて再生する波形表 *LeafWaveTable* (*flicker_rate*)を，波形表 *LeafBaseTable*(*leaf_size*)を用いて作成するブロックである．

初めのブロック，すなわち波形表 *LeafBaseTable* (*leaf_size*)の作成は，まず，入力された葉を解析し，葉身の面積の最大・最小値を求め，これを基に葉身の面積ごとのクラス分けを行い，葉の面積階級別出現頻度分布を葉の評価値として保持する．簡単のため，葉の形状データは葉柄を線分，葉身を長方形で近似したものを採用する．一般に，物体の大きさが大きいほど，基底周波数は低くなるという定性的な関係がある．しかしながら，葉の大きさと基底周波数との関係に関するこれまでの研究例が見当たらなかったため，本論文では以下のような関係を仮定する．

$$f\_basis = \frac{\alpha}{leaf\_size} \tag{7}$$

ここで，*f_basis* は基底周波数，*leaf_size* は葉身の葉柄方向の長さ，α は樹種に応じて調整する定数である．本研究では，α は試行錯誤的に求めている．基底周波数から，周波数スペクトルが $1/f^{\beta}$ ノイズとなるような音色を作成し，逆フーリエ変換することで時間領域の波形表 *LeafBaseTable* (*leaf_size*)を得る．葉の音色は枝の音色の決定と同様に，

$$\begin{aligned} &I = (f / f\_basis)^{\beta} \\ &I : \text{音の強さ} \\ &f : \text{周波数} \\ &f\_basis : \text{基底周波数} \\ &\beta : \text{音色パラメータ} \end{aligned} \tag{8}$$

で決定する．

次のブロックでは，上記で求めた波形表 *LeafBase Table*(*leaf_size*)と評価値を用い，揺らぎの程度に応じて再生される波形表 *LeafWaveTable*(*flicker_rate*)を作成する．単位時間内の葉の衝突数 *numKeyOn* は揺らぎの程度 *flicker_rate* と葉群の規模 *scale* により決定する．

$$numKeyOn \propto flicker\_rate \cdot scale \tag{9}$$

葉群の規模 *scale* は，樹木 *Tree* で一定の値とする．この値は，任意に選択した複数の葉 L_i について，それぞれ，時刻 t=0 における $L_i(0)$を中心としたある範囲内に存在する周囲の葉の面積の総和を求め，それらの平均値に樹木 *Tree* の体積を掛けることで決定する．衝突時

の振幅 *envelope* は *flicker_rate* に比例させる．

$$envelope \propto flicker\_rate \tag{10}$$

葉の衝突が発生した場合，衝突した葉の大きさを，葉の評価値（葉の面積階級別出現頻度分布）に応じた確率で選択する．波形表 *LeafBaseTable*(*leaf_size*)から，選択された葉の大きさ $leaf\_size_i$ の波形表 *LeafBaseTable*(*leaf_size*$_i$)を使用する．また，衝突が摩擦に移行する場合の減衰や，摩擦状態の減衰も揺らぎの程度 *flicker_rate* に依存させ，

$$envelope(t) = envelope(t-1) \cdot flicker\_rate \tag{11}$$

とする．衝突状態および摩擦状態が終了する数 *numKeyOff* は衝突・摩擦状態の継続条件に関連するものとする．継続条件は，乱数 *rand* を用い，

$$flicker\_rate \le \lambda \cdot rand \tag{12}$$

が真で継続，偽で終了する．これは，揺れが激しいほど継続せず，*numKeyOff* が増加する．ここで，λ は継続の条件を調整する定数で，本研究では試行錯誤的に求めた．筆者らは，λ に 20 を用い，1/30[s]ごとに継続判定を行っている．一定の *flicker_rate* に対して，λ を小さくすると音が途切れやすく，衝突音が強調される．一方，大きくすると摩擦音が継続し，衝突音が認識されなくなる．

以上の手法をまとめ，以下のように波形表 *LeafWaveTable*(*flicker_rate*)を作成する．

(a) まず，入力された樹木 *Tree* から，葉の集合 *L*(0)を評価し，樹木 *Tree* に対する葉の評価値と葉群の規模 *scale* を求める．

(b) 式(7)，(8)を用いて波形表 *LeafBaseTable*(*leaf_size*)を作成する．

(c) 各 $flicker\_rate_i$ に対し，式(9), (10)を用いて *numKeyOn*，*envelope* をそれぞれ計算する．

(d) 波形表 *LeafWaveTable*($flicker\_rate_i$)の作成のため，葉の評価値により，波形表 *LeafBaseTable*(*leaf_size*)から一つの波形表 *LeafBaseTable*(*j*)を選択する．選択された *LeafBaseTable*(*j*)に，*envelope* の値を掛けて，あらかじめ指定した単位サンプル数だけ波形表 *LeafWaveTable*($flicker\_rate_i$)に加える．式(12)の継続条件で真偽の判定を行う．真であれば式(11)を用い *envelope* の減衰を行い，単位サンプル数だけ足し合わせをおこなう．この作業を継続条件が偽になるまで繰り返される．

(d)は *numKeyOn* だけ行う．(c)と(d)は入れ子構造であり，(c)をすべての $flicker\_rate_i$ に対して行うことで，*LeafWaveTable*(*flicker_rate*)を作成できる．

揺らぎの程度 *flicker_rate* の値は，葉の揺らぎの動きに関連付けて決定する．本研究では枝葉の揺らぎに文献[1]で提案された手法を使用している．この手法では，$1/f^{\beta}$ ノイズを使用して枝葉の揺らぎを効率的に表現する．まず，個々の葉 L_i の葉柄座標系として，時刻 t=0 の $L_i(0)$を考え，その葉柄方向を z 軸，z 軸と鉛直上向きベクトルとの外積を x 軸，z 軸と x 軸との外積を y 軸とする．葉の揺らぎは，葉柄座標系に対し，２つの $1/f^{\beta}$ ノイズ，すなわち，ローカルなノイズ *Local*(*t*)とグローバルなノイズ *Global*(*t*)を使用して与える．枝の根元から先端へ，ある程度整列された葉 $L_i(t)$の x, y 軸方向の揺れ幅をそれぞれ $\theta_i(t)_x$，$\theta_i(t)_y$，z 軸を中心とする回転方向の揺れ幅を $\theta_i(t)_r$ として，

$$\begin{aligned}
\theta_i(t)_x &= Local_x(t - i \cdot p) \cdot Global(t) \\
\theta_i(t)_y &= Local_y(t - i \cdot p) \cdot Global(t) \\
\theta_i(t)_r &= Local_r(t - i \cdot p) \cdot Global(t) + \alpha\theta_i(t)_x
\end{aligned} \tag{13}$$

となる．ここで，$Local_x(t)$，$Local_y(t)$，$Local_r(t)$は各方向に適用するノイズ値を，p は葉の位相差の増加率を表す値で，ユーザがインタラクティブに設定できる[1]．この２種類のノイズによる揺らぎの表現において，ローカルなノイズは個々の葉に個別の位相を与え，樹木全体の葉が同一に揺らぐのを防ぐ役割をもつ．一方，グローバルなノイズは，時間の経過に伴う樹木全体の揺らぎの程度を表現していると考えられる．よって，揺らぎの程度を表すパラメータ値 *flicker_rate* は，グローバルなノイズ値のみを用い，次式で決定する．

$$flicker\_rate = |Global(t)| \,/\, MaxGlobal \tag{14}$$

MaxGlobal はグローバルなノイズ *Global*(*t*)がとり得る値の絶対値の最大値を表す．

５．２　オンラインプロセス

オンラインプロセスは，枝の効果音のオンラインプロセスと同様に，入力された風の強さから揺らぎの程度 *flicker_rate* を求め，量子化を行い，*flicker_rate* に対応した波形表 *LeafWaveTable*(*flicker_rate*)を再生するだけである．また，枝の効果音同様，波形表の ID 及び再生開始点を用いることで連続性を確保し，複数の波形表のクロスフェードを行うことでノイズの混入を防ぐ．

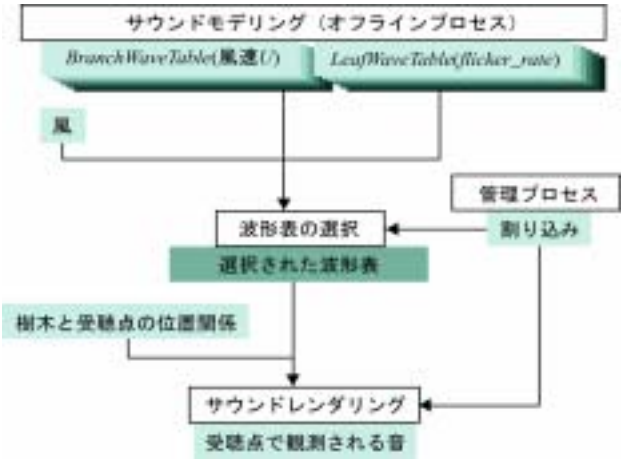

図４：オンラインプロセス

６．サウンドレンダリング

４章と５章で樹木のサウンドモデリングについて提案した．サウンドレンダリングは，オンラインプロセス上の波形表の選択プロセスの結果出力されたデータを樹木 *Tree* の位置で発生する音として，最終的に受聴点で観測される音について計算を行う（図４）．

波形表の選択プロセスでは，管理プロセスからの割込みが受信された時点での風の情報を使用し，４．２節と５．２節に示すように，枝の効果音には風の強さ U，葉の効果音には揺らぎの程度 *flicker_rate* として解釈し，どの波形表を再生するかの選択を行う．

管理プロセスの割込みには生成できるオーディオサンプルの数の情報を付加させ，サウンドレンダリングで受信したオーディオサンプルの数だけオーディオサンプルを作成し，オーディオデバイスに送信する．樹木と受聴点との位置関係による音響効果はサウンドレンダリングで考慮する．サウンドレンダリングには文献[4]で提案された音の伝達を表現する手法を用いる．これは，受聴点に仮想マイクロホンを設置し，仮想マイクロホンの指向性は単一指向性を表すカーディオイド関数を使用する手法である．カーディオイド関数を次式で示す．

$$S(\mu)=C+((1+\cos\mu)/2)^k \qquad (15)$$

本論文では係数として k=1，C=0 を用いる．μ は樹木から受聴点へ向かうベクトルと仮想マイクロホンの向きとの角度である．空気中での音速は約 340m/s であり，音は樹木と受聴点との距離に応じて遅延･減衰を行う．音は距離の２乗に反比例して減衰することを考慮すると，最終的に得られる音の大きさは，基準となる距離

図５：スクリーンショット
上：樹木 A，下：樹木 B

d_0 での音の大きさ A_0 を使用して，次式で表される．

$$A(d)=A_0(d_0/d)^2S(\mu) \qquad (16)$$

ここで，d は音源と受聴点との距離である．

７．実行例

上記のアルゴリズムを，Windows XP で Visual C++ を用いて実装した．CPU は 2.53GHz Pentium®4，1.5Gbyte RDRAM メモリを使用した. 描画には OpenGL を用いている．図５は実行時のスクリーンショットを示す．１本の樹木を１つの音源として実行した．表１に, 計算量の比較のために用いた 2 種類の樹木データ，および，参考までに，効果音生成を行わずに，樹木の揺らぎのみを適用して実行したときのフレームレートと平均 CPU 利用率を示す. 実行例のアニメーションは付属のメディアに添付する．実行例のアニメーションは, (a)樹木 A の枝, (b)樹木 B の枝, (c)樹木 A の枝葉, (d)樹木 B の枝葉の４種である．(a)，(b)，(c)のアニメーションのフレームレートは十分であるが，(d)については十分ではない．しかしながら，(d)についても効果音の生成には問題はなく，音質への影響はない．添付のアニメーションは，実行画面を直接撮影したもので

画質は良くない．効果音は，ステレオで録音してあるので，樹木による音の違いだけでなく，受聴点の動きにあわせた音の移動が確認できる．

枝の効果音については，量子化における直径 D の分割数と流速 U の分割数が音質に関するパラメータとなる，また，葉の効果音については，葉柄の長さ L の分割数と揺らぎの程度 *flicker_rate* の分割数が音質に関するパラメータとなる．これらのパラメータは大きいほど良い結果が期待されるが，あまり大きすぎると，処理に時間がかかり，使用するメモリ量が増大する．分割数は大きすぎてもその効果が知覚できないので，計算環境に依存した分割数を決定する必要がある．本章ではこれらの分割数を，8，16，32，64，128，256 に設定し，各々の組み合わせに対する計測を行った結果を示す．なお，筆者らと研究室構成員数名が感覚的に自然であると感じた各パラメータの最小値は，直径 D の分割数が 16，流速 U の分割数が 128，葉柄の長さ L の分割数 8，揺らぎの程度 E の分割数 128 である．流速の分割数や *flicker_rate* の分割数は，風の量子化の度合いを示しており，これらの分割数が小さいと，段階的な音の変化が認識されてしまう．直径の分割数や葉身の分割数は，樹木 *Tree* を構成する枝と葉の大きさの量子化の度合いを示しており，これらの分割数が小さいと，基となる音の種類が少なくなり，効果音の印象が単調になる．本研究での実験によれば，上述した最小分割数以上の分割であれば，枝数や葉数が異なった場合でも，既提案手法と比較して，効果音の劣化は認められなかった．以下，上記のパラメータの分割数に関して，効果音生成に必要とした時間を計測した結果を示す．

７．１　枝のオフラインプロセスの時間計算量

枝の効果音生成のオフラインプロセスは，枝データの解析と波形表生成とに分かれている．枝データ解析に要した時間は樹木 A，B 共に 1ms 弱であり，波形表生成に要した時間に比べて無視できる程度であった．表２は波形表作成に関する時間計算量である．樹木の大きさに関わらず, 直径 D の分割数と流速 U の分割数に比例しているといえる．

７．２　葉のオフラインプロセスの時間計算量

葉のオフラインプロセスは，葉データの解析，葉の

表１：樹木データと揺らぎのフレームレート

	樹木A	**樹木**B
枝の本数	123	1160
葉の枚数	4860	16380
平均フレームレート[fps]	21.3	6.4
平均CPU**利用率**[%]	71	91

表２：分割数と計算時間（S）

上：樹木 A，下：樹木 B

枝: 123		**流速の分割数**					
		8	16	32	64	128	256
直径の分割数	8	6.77	13.22	25.99	51.50	102.67	204.98
	16	13.58	26.00	51.05	101.17	201.67	402.72
	32	26.70	51.31	100.77	199.61	397.91	794.59
	64	52.47	101.20	198.77	394.08	785.44	1569.39
	128	103.34	199.64	393.09	779.91	1555.28	3107.38
	256	202.94	395.03	779.86	1550.80	3092.95	6181.91

枝: 1160		**流速の分割数**					
		8	16	32	64	128	256
直径の分割数	8	6.05	12.59	25.44	51.13	102.47	205.22
	16	11.98	24.95	50.28	100.92	202.30	405.09
	32	24.42	49.45	99.44	199.39	399.44	799.49
	64	48.73	98.17	197.16	394.98	790.91	1583.14
	128	96.78	194.66	390.36	781.75	1564.42	3130.06
	256	192.63	386.23	773.41	1548.14	3097.08	6197.50

表３：葉の大きさに依存した波形表作成

葉: 4860		*flicker_rate* **の分割数**					
		8	16	32	64	128	256
葉身の分割数	8	0.80	0.78	0.80	0.80	0.78	0.80
	16	1.59	1.59	1.58	1.59	1.58	1.59
	32	3.19	3.19	3.19	3.17	3.17	3.17
	64	6.36	6.36	6.36	6.38	6.36	6.38
	128	12.70	12.72	12.73	12.69	12.70	12.73
	256	25.44	25.45	25.42	25.47	25.45	25.45

葉: 16380		*flicker_rate* **の分割数**					
		8	16	32	64	128	256
葉身の分割数	8	0.80	0.78	0.78	0.80	0.80	0.78
	16	1.58	1.59	1.58	1.59	1.59	1.59
	32	3.19	3.17	3.17	3.17	3.19	3.19
	64	6.36	6.36	6.34	6.38	6.38	6.38
	128	12.67	12.67	12.70	12.66	12.67	12.70
	256	25.31	25.33	25.31	25.34	25.31	25.34

表４：実際に再生される波形表作成

葉: 4860		*flicker_rate* **の分割数**					
		8	16	32	64	128	256
葉身の分割数	8	53.97	120.59	252.98	517.59	1048.95	2102.78
	16	54.30	120.08	252.27	518.67	1045.69	2102.95
	32	53.84	120.61	252.42	514.23	1047.47	2105.24
	64	54.27	120.75	252.08	517.41	1047.22	2107.73
	128	54.20	120.55	252.70	517.92	1047.25	2106.17
	256	54.00	121.16	252.45	517.39	1048.34	2108.95

葉: 16380		*flicker_rate* **の分割数**					
		8	16	32	64	128	256
葉身の分割数	8	245.36	546.70	1148.64	2351.06	4760.64	9569.45
	16	246.73	548.45	1147.22	2348.88	4756.00	9578.38
	32	246.30	548.11	1147.83	2351.58	4761.45	9582.14
	64	246.33	547.83	1148.88	2353.91	4763.05	9588.17
	128	246.50	548.53	1148.17	2353.42	4762.81	9587.72
	256	247.30	547.19	1148.05	2353.14	4762.14	9590.80

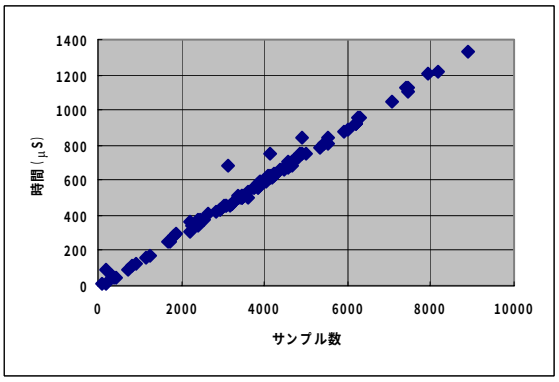

図６：効果音生成時間計測

大きさ *leaf_size* に依存した波形表 *LeafBaseTable* の作成（LP1），そして，*numKeyOn* や継続条件等を用いた実際に再生する波形表 *LeafWaveTable* の作成（LP2）の３つに分けられる．枝と同様に，解析に要した計算時間は樹木 A，B 共に 1ms 弱であった．表３は LP1 の時間計算量，表４は LP2 に要した時間計算量である．

LP1 は葉柄の分割数に，LP2 は揺らぎの程度の分割数にそれぞれ比例しており，LP1 より LP2 の方が常に大きな時間計算量となっている．LP1 の葉の枚数による計算時間の増大は，葉の大きさの違いから来ていると考えられる．樹木 A より樹木 B の葉の方が大きいので，作成される波形表の基底の周波数は低くなる．したがって，樹木 A より樹木 B の方が，波形表 *LeafBaseTable*(*leaf_size*) 作成の際により多くの計算を必要とするため，それが時間計算量に影響していることが考えられるが，それほど大きい影響ではない．LP2 に関する時間計算量は，葉の規模（*scale*）の違いに関連している．揺らぎの程度が同じなら，*scale* が大きいほど葉の衝突数 *numKeyOn* が多くなり，結果的に多くの計算を必要とする．

７．３　効果音生成の時間計算量

効果音生成プロセスは，時間・デバイス管理プロセスの割込み時に作成できるオーディオサンプルの数を受け取り，その数だけ波形を作成する．入力の樹木が複数である場合，また，一つの樹木に複数の音源を配置した場合，リアルタイム性を維持するために，このプロセスがどの程度の音源に対してリアルタイムに処理できるかがポイントとなる．図６は，作成すべきオーディオサンプルの数に対し，処理に要した時間をプロットしたものである．このグラフはほぼ直線であり，1 秒（44100Hz）の音生成に対し，平均で約 7ms 要している．したがって，上記の計算機に効果音生成のみを処理させた場合，140 個程度の音源を配置できる．

７．４　領域計算量

本提案手法では，枝葉ともに波形表を用いることで効率的な音生成を可能としている．波形表の保持に必要なメモリの大きさは，枝の波形表は流速 U の分割数，葉の波形表は揺らぎの程度 *flicker_rate* の分割数に比例する．本提案手法では，波形表のサンプル数を 44100，一つのサンプルを倍精度浮動小数点型（64bit）で表現しており，分割数を n とした場合，枝葉共に使用するメモリの大きさは，

$$44100[sample]\cdot 64[bit]\cdot n[\text{分割数}]/8[bit/byte] = 0.3528n[Mbyte] \quad (17)$$

となる．実装する場合には利用可能なメモリ領域内に収まるように分割数を設定する必要がある．また，波形表のサンプル数は波形表を繰り返し使用することで抑えることができるが，波形表を繰り返し使用すると音の単調さが知覚されてしまうことがあるので，サンプル数を大きめにして繰り返しの使用を防いでいる．

８．まとめと今後の課題

本論文では，樹木に対し，風による揺らぎによる効果音を自動的にかつ効率的に付加するサウンドモデリング法を提案した．

本提案手法は，枝の風による音と葉同士が接触し合う音を生成するものである．枝同士の衝突，枝と葉との衝突，葉と風による音，そして，葉が飛ぶときに鳴る音などの生成法の開発が今後の課題としてある．また，本手法では広葉樹を想定しており，針葉樹への拡張も残されている．さらに，風の強さについては日常的なレベルを想定しており，嵐のような強風への拡張も興味深い．

また，本提案手法では効率性を重視しているため，複数の枝と葉を一つの音源としてまとめ，その場所での風により効果音を生成している．近接樹木のように，受聴点に対して広がりが存在する場合には，樹木を分割し，音源数を増加させる必要がある．これは音源数に関する LOD の問題であり，インタラクティブなウォークスルーなどへ適用するためには，樹木の規模，受聴点，計算資源などから動的に決定する手法の開発

も大切である．

謝辞

本研究の一部は，「夢県土いわて戦略的研究推進事業」の支援による．

参考文献

[1] Ota, Tamura, Fujita, Muraoka, Fujimoto, and Chiba, “1/f$^{\beta}$ Noise-Based Real-Time Animation of Trees Swaying in Wind Fields,” *Proceedings of the Computer Graphics International 2003 international conference*, 2003.

[2] 松山，藤本，村岡，千葉，風による樹木の揺らぎの効果音の生成法，第 18 回 NICOGRAPH 論文コンテスト，芸術科学会，pp. 41-46，2002．

[3] 松山，藤本，村岡，千葉，風による樹木の揺らぎの効果音の生成－葉の揺らぎの効果音－，2003 年 NICOGRAPH 春季大会, 芸術科学会, pp. 67-68，2003．

[4] Takala, T. and Hahn, J. Sound Rendering, In *Proceedings of ACM SIGGRAPH92*, ACM, 1992; 26: 2: 211-220.

[5] Funkhouser, T., Carlbom, I., Elko, G., Pingali, G., Sondhi, M. and West, J. A beam tracing approach to acoustic modeling for interactive virtual environment, In *Proceedings of ACM SIGGRAPH98*, ACM, 1998; 21-32.

[6] Funkhouser, T., Min, P. and Carlbom, I. Real-time acoustic modeling for distributed virtual environments, In *Proceedings of ACM SIGGRAPH99*, ACM, 1999; 365-374.

[7] Tsingos, N., Funkhouser, T., Ngan, A. and Carlbom, I. Modeling acoustics in virtual environments using the uniform theory of diffraction, In *Proceedings of ACM SIGGRAPH2001*, ACM, 2001; 545-552.

[8] Tsingos, N., Carlbom, I., Elko, G., Kubli, R. and Funkhouser, T. Validating acoustical simulations in the bell labs box, *IEEE Computer Graphics and Applications*, 22, 4, 2002; 28-37.

[9] Doel, K., Kry, P. and Pai, D. Foley automatic: Physically-based sound effects for interactive simulation and animation, In *Proceedings of ACM SIGGRAPH*, ACM, 2001; 537-544.

[10] O’Brien, J. Cook, P. and Essl, G. Synthesizing sounds from physically based motion, In *Proceedings of ACM SIGGRAPH2001*, ACM, 2001; 529-536.

[11] Dubnov, S., Joseph, Z., Yaniv, R., Lischinski, D. and Werman, M. Synthesizing sound textures through wavelet tree learning, *IEEE Computer Graphics and Applications 22*, 4, 2002; 38-48.

[12] Dobashi, Y., Yamamoto, T., Nishita, T., Real-Time rendering of aerodynamic sound using sound textures based on computational fluid dynamics, *SIGGRAPH2003* , ACM, 2003

[13]土橋，山本，西田，流体力学に基づくサウンドテクスチャーを用いた風きり音のリアルタイムレンダリング，Visual Computing / グラフィクスと CAD 合同シンポジウム 2003

[14] Pai, D., Doel, K., James, D., Lang, J., Lloyd, J, Richmond, J. and Yau, S. Scanning physical interaction behavior of 3D objects, In *Proceedings of ACM SIGGRAPH2001*, ACM, 2001; 87-96.

[15] O’Brien, J. and Hodgins, J. Graphical modeling and animation of brittle fracture, In *Proceedings of ACM SIGGRAPH99*, ACM, 1999; 137-146.

[16] Reffye, P., Edelin, C., Francon, J., Jaeger, M. and Puech, C., Plant Models Faithful to Botanical Structure and development, Computer Graphics, *Proc SIGGRAPH 88*, Vol.22, No.4, pp.151-158, 1988

[17] Chiba, N., Ohshida, K., Muraoka, K., Miura, M., Saito, N., A growth model having the abilities of growth-regulations for simulating visual nature of botanical trees, *Computers & Graphics* Vol.18, No.4, pp.469-479, 1994

[18] Mech, R., Prusinkiewicz, P., Visual models of plants interacting with their environment, Computer Graphics, *Proc SIGGRAPH 96*, pp. 397-410, 1996

[19] 望月修，丸田芳幸，流体音工学入門，朝倉書店，1996