樹木シミュレーションに基づくリアルタイム環境音合成システムの開発

筑波大学
人間総合科学学術院人間総合科学研究群
情報学学位プログラム

2023年03月

松浦　一輝

樹木シミュレーションに基づくリアルタイム環境音合成システムの開発
A System for Interactive Sound Synthesis Based on Tree Simulation

氏名：松浦　一輝
Matsuura Kazuki

近年の映画やゲーム，インタラクティブなコンテンツなどエンターテインメント分野におけるコンピュータグラフィックス (CG) は映像表現のために欠かせない技術となっており，CG の中でも自然現象をコンピュータ内で再現する物理法則に基づくシミュレーションは多くのシーンで用いられている．これは現実により近い複雑な自然現象の映像をクリエイターの技量に依存せずコンピュータによる計算で自動生成することができるという利点があるためである．こうした映像生成においては，視覚的な CG だけでなく，聴覚的な音も現実感を高めるために重要な要素である．従来の方法では，CG 映像に音を追加するために，利用したい音をあらかじめ録音もしくは合成しておき，アニメーションと音を同期させるという作業が必要になるため，クリエイターの作業コストが高い．加えて用意した音しか鳴らすことができないため，シーンに合わせたインタラクティブな音の変化を実現することも難しい．そこで，物理シミュレーションに基づく計算処理によりシーンに合わせた音を自動生成できれば，映像に合わせた音の付随したリアルな CG アニメーションが作成でき，音を別で録音または合成する手間もなくなる．また，リアルタイム生成を可能にすることで，ユーザが風の強さや音の鳴る物体の形状のようなシーン内のパラメータを変更した際に再計算の時間を待たずに音を変化させることができるようになり，さらにはユーザが物体を揺らす，ぶつけるのような毎回揺れ動き方が異なるシーンにもインタラクティブに対応可能である．

本研究では樹木の揺らぎにより発生する環境音を樹木シミュレーションに基づき生成するシステムを提案する．樹木の揺れ動きを位置ベース法を用いて細かい枝の動きや葉の接触までリアルタイムにシミュレーションし，そこから木の葉の衝突と摩擦のタイミングを取得する．衝突音については葉形状にモード解析を行い固有振動数を求め音を生成し，摩擦音に関しては 1/f ノイズを用いて自然界の摩擦音に見られる再帰的な周波数特性を再現する．それに加えて 1/f ノイズに reson filter を適用することで風切り音も生成し，これら樹木シーンの環境音をシーンと連動しながら再生する．本システムでは任意に設定できる樹木や葉の形状における各種パラメータを用いて，シミュレーションに基づく音の生成ができるため，今まで CG 制作現場で行われていたような，録音または合成した音を後で合わせるという手間がなく，映像と音の同時生成が可能となる．

主研究指導教員：藤澤　誠
副研究指導教員：三河　正彦

目 次

図 目 次

表 目 次

第1章　序章

近年の映画やゲーム，インタラクティブなコンテンツにおけるコンピュータグラフィックス (以下，CG とする) は映像表現のために欠かせない技術となっており，CG の中でも自然現象をコンピュータ内で再現する物理法則に基づくシミュレーション（以下，物理シミュレーション）は多くのシーンで用いられている．これは高いクオリティの映像，つまり現実により近い複雑な自然現象の映像をクリエイターの技量に依存せずコンピュータによる計算で自動生成することができるためである．そして，このような映像生成において，視覚的な CG だけでなく，聴覚的な音も現実感を高めるために重要な要素である．例えば，ゲームなどのコンテンツにおいては物体同士が衝突した時の衝突音をはじめ，爆発音や流水音など，その映像に付随する音は現実感を増すために重要な役割を担っている．

従来の CG アニメーションでシーンに音を追加するためには，利用したい音をあらかじめ録音もしくは合成しておき，アニメーションと音を同期させるという作業が必要になっていた．この工程では映像とは別に音を作るための手間と音の種類とタイミングを適切に調整する手間がかかる．さらに用意した音しか鳴らすことができないため，シーンに合わせたインタラクティブな音の変化を実現することも難しい．一方で，物理シミュレーションに基づく計算処理によりシーンに合わせた音を自動生成できれば，映像に合わせた音の付随したリアルな CG アニメーションが作成でき，音を別で録音または合成する手間もなくなる．また，リアルタイム生成を可能にすることで，ユーザが風の強さや音の鳴る物体の形状のようなシーン内のパラメータを変更した際に再計算の時間を待たずに音を変化させることができるようになり，さらにはユーザが物体を揺らす，ぶつけるのような毎回揺れ動き方が異なるシーンにもインタラクティブに対応可能となる．

本論文では，CG アニメーションでも頻出する要素である樹木の揺れ動くシーンとそこで発生する環境音を物理シミュレーションに基づいてリアルタイムに生成するシステムを提案する．ここでいう樹木の揺れ動くシーンにおける環境音とは一般に木々のざわめきなどと呼ばれ，風などの外力により樹木の枝や葉が揺れ動くことで発生する音であり，その主な要素として，木の葉同士の衝突音および摩擦音，そして風が枝の間をすり抜ける際に発生する風切り音が考えられる．このような環境音をリアルタイムで生成するためには，音の発生源となる樹木の揺れ動きも高速にシミュレーションできる必要がある．本研究では弾性体を高速かつ安定に計算可能なシミュレーション手法である位置ベース法 (Position Based Dynamics) [1] をノードとエッジにより離散化した木構造に適用することで木の揺れ動きを計算する．その物理シミュレーションの結果から音の発生源となる木の葉や風の状態を取得し，衝突音，摩擦音及び風切り音を鳴らすことで，環境音を生成する．

固体の衝突音は物体が衝突による外力に起因して特定の共振周波数で振動することで発生する．共振周波数は物体の材質や形状に依存し，ヤング率と境界条件からモード解析を行うことで求めることができる．モード解析のために有限要素法を用いる手法 [2] があるが，計算コストが大きく低速である．本論文では有限要素法と比べて非常に高速な計算が可能であるバネ-質点系で近似した物体に対してモード解析を行い，その結果を衝突音合成に用いるという手法 [3] を使用することで計算量を減らし高速化する．摩擦音に関しては自然界

の摩擦音は1/fノイズと呼ばれる自己相似性を持つ音であることに注目し，生成した1/fノイズを接触や風の状態に応じて再生することで再現する．風切り音は風が枝の間をすり抜けていく際に発生する音であり，これはカルマン渦の周期的な渦放出が原因で発生し摩擦音と同様に自己相似性を持つ．この音の周波数はカルマン渦放出周波数と同じであるため，カルマン渦列の周波数を基底周波数とし，1/fノイズに特定の周波数帯域を強調しつつ自己相似性を再現するreson filter[4]をかけることで波形を作成し，風の状態に応じて音の大きさを変化させることで風切り音を再現する．

本論文で提案する環境音合成システムは，様々な形状の樹木が揺れ動くシーンに対応可能であり，ユーザが任意に設定し自由に変更することが可能な樹木の各種パラメータを用いて音と映像を生成するため，パラメータ設定だけで任意の映像と音を生成可能かつインタラクティブな変化にも対応できる．

第2章　関連研究

2.1　樹木の発する音のリアルタイム生成

CG 分野においてシーンに自動で効果音を付与する研究は盛んに行われており，その種類は剛体の衝突音や摩擦音に加え，水の流れる音や空気の流れによる風の音など多岐にわたる．このような音合成を行う場合大きく分けて 2 つの方法があり，1 つはあらかじめ録音・生成しておいた音源をシーンに合わせて再合成し再生する手法，もう 1 つは物理演算などにより，発生する音を直接計算し生成する手法である．本論文で扱う樹木の揺らぎによる音のような多数の構成要素からなる物体の音を生成する場合はシーン毎に物理演算をすることは難しく，前者の方法がとられる場合が多い．

例えば，松山ら [5] の樹木の発する音のリアルタイム生成の研究では，あらかじめいくつかの波形表を用意しておき，モデル化した葉の大きさと接触状態に応じてその波形表の音を再生するという手法を提案している．しかし，このような手法はあらかじめ波形表を用意しておかなければならないことに加え，その用意した種類の音しか再生できない．本論文では，用意した音源をシーンに合わせて再生する方法と，シーン毎の物理シミュレーションを音の種類によって使い分けながら組み合わせており，音源の用意もユーザの木の葉形状の入力，変更に応じて自動で行うため，あらかじめ音源を用意する必要がない上に，木の葉の形状に応じた音の変化を再現できる．

2.2　物理シミュレーションによる音生成

樹木の揺らぎによる音とそれに付随する環境音には主に木の葉同士の衝突音と摩擦音，風が枝の間を通り抜けることによる風切り音の 3 つが挙げられる．

2.2.1　衝突音

物体の衝突音を生成するためにはその物体の共振周波数をモード解析により求める必要がある．O'Brien ら [2] は有限要素法を用い物体形状のモード解析を行い，それにより複素周波数として導かれる共振周波数の虚部と衝突時のインパルス応答から計算される振幅，実部から計算される減衰項，虚部から計算される角速度を用い波の式を生成した．Eston ら [6] はバネのような細長く弾性変形するロッド形状の物体の発する音のシミュレーションに O'Brien らの音の波の式を利用した．この方法は他にも薄い剛体の衝突音 [7] などにも用いられている．しかし，有限要素法を用いたモード解析は計算量が多く低速である．Rausch ら [8] は GPU を用いて高速化する手法，Raghuvanshi ら [3] はバネ-質点系を用いて高速化する手法，Jin ら [9] は機械学習を用いて計算を高速化する手法をそれぞれ提案した．

本論文では，モード解析には Raghuvanshi らの物体を複数のバネ-質点系で近似し，そのそれぞれについてモード解析を行い，計算結果を後から合成するという手法を利用し，これにより高速計算を可能にする．

2.2.2 摩擦音

摩擦音は発生機構が複雑であることから，その音の特性を再現するという方法がとられることが多い．例えばDoelら [4] はランダムに生成した音の波にreason filterというフラクタルフィルターをかけることで摩擦音の性質を再現し，さらに接触の速度に応じて再生速度を変えた．このフィルターは特定の共振周波数を指定した帯域幅で強調させるもので，この共振周波数と帯域幅は物体の材質と表面形状によって決定される．中塚ら [10] は癒着説に基づく摩擦の物理シミュレーションにより摩擦音を生成した．具体的には，摩擦力によって生じる実接触点の振動を計算した後に，各点で生じる音を観測点にて合成し摩擦音を生成している．

しかし，これらはいずれもある程度強い力で擦り合わせた際に単一の物体から発生する摩擦音を想定しており，今回のような小さな摩擦音が重なり合い大局的なフラクタルノイズとなるシーンでは扱いづらい．そのため本論文ではランダムな信号を元に生成した音全体で自己相似性を持つ1/fノイズを，ユーザが入力したパラメータを元に変化させ，シーンに応じて再生するという手法をとる．

2.2.3 風切り音

風切り音に関しては，Dobashiら [11] がサウンドテクスチャを用いた風きり音や棒状の物体を振った時の空力音のリアルタイムレンダリング法を提案している．この手法は数値流体力学に基づいた物体の各部分における空気の流れの事前計算を大量に行う必要がある．本手法ではユーザ入力により木の形状などをリアルタイムに変更されることから，多くの事前計算を必要とする方法は適していない．松山ら [5] はカルマン渦列の周波数を規定周波数とし，それに音色を付加する手法を提案した．本研究では，このカルマン渦の周波数を基底周波数とする方法を参考に，1/fノイズにその特徴を持たせるようにreson filterをかけるという手法で風切り音を生成する．

2.3 木の揺れ動きのシミュレーション

樹木の揺らぎにより発生する環境音をリアルタイムで生成するためには，音の発生源となる樹木の揺れ動きも高速にシミュレーションできる必要がある．Wangら [12] は有限要素法を用いて，高計算コストであるが，材料特性を指定した単一の樹木の非常にリアルなアニメーション生成する手法を提案した．Otaら [13] は枝葉の揺れに1/fノイズを応用する確率的手法とシミュレーション手法を組み合わせたハイブリッド手法を提案した．Liら [14] は有限要素法を用いて事前計算したテーブルを用意し1000本の木がある森林のリアルタイムシミュレーションを可能にした．これらの方法は木ごとに全体的な動きを生成するため，細かい枝の動きや葉の衝突タイミングが取得できず，音生成への応用は難しい．

本論文では，木をノードとエッジに離散化し，位置ベース法を用いることでリアルタイムに細かい枝や葉の動きまでシミュレーションする方法を提案する．

2.3.1 位置ベース法

Mullerら [15] の提案した位置ベース法 (Position based dynamics) は従来の力学ベースの手法とは異なり幾何学的な制約条件に従って物体の位置を動かす手法であり，弾性体を高

速かつ安定にシミュレーションすることが可能である．しかし，この手法はノード位置の計算のみを行うため，回転情報を保持出来ず，例えば毛髪などのロッド形状における曲げ・ねじれを表現することはできない．そこで Umetani ら [16] は，新たに曲率と捩率を保持するためのゴーストポイントを追加することで，位置ベース法での曲げとねじれに対する弾性変形のシミュレーションを実現した．さらに，Kugelstadt ら [17] は Umetani らの手法のゴーストポイントと同じ役割に四元数 (Quaternion) を用いることでより高速にシミュレーションを行うことを可能にした．

これらの手法では細長い弾性体をノードとエッジで離散化し，隣り合うノード間に対する距離制約と隣り合うエッジ間の曲げ・ねじれ制約を用いている．本手法では，Kugelstadt らの手法を元に離散化した木構造に親ノード，親エッジの情報を持たせ，親子ノード，親子エッジ間に制約をかけることで枝分かれする木構造に応用し，リアルタイムに細かい枝や葉の動きまでシミュレーションすることを可能にする．

第3章　提案手法の概要

本論文で提案するシステムは木の揺れ動きのシミュレーションと音合成の 2 つで構成される．木の揺れ動きのシミュレーションはノードとエッジで離散化した木構造に位置ベース法を適用し，ノード間に距離制約，エッジ間に曲げ・ねじれ制約をかけることで高速かつ安定にシミュレーションを行う．音合成は衝突音生成，摩擦音生成，風切り音生成の 3 つに分けられ，それぞれについて，木の葉をバネ-質点系で近似したボクセルモデルにモード解析を行い，その結果を衝突音合成に用いる手法，1/f ノイズを接触や風の状態に応じて再生し摩擦音を生成する手法，音発生の原因であるカルマン渦の基底周波数を計算し，その 1/f ノイズにおける周波数帯域を reson filter を用いて強調し風切り音を生成する手法を提案する．

本論文では 3.1 節でシステム全体の構成とシミュレーションの流れを説明した後，3.2 節でユーザが指定できる各種パラメータとその入力方法について説明する．その後 4 章で詳しい揺れ動きのシミュレーション手法について，5 章で音の生成と再生手法について説明する．

3.1　システム全体の構成

図 3.1 に提案手法のシステム全体構成を示す．まず前処理としてユーザの入力パラメータから初期木構造の構築を行い（4.2 節），それらの値から衝突音に用いる固有周波数を計算する弾性行列の作成（5.1.2 項，5.1.3 項）とモード解析（5.1.1 項），摩擦音に用いる 1/f ノイズの生成（5.2 節），風切り音に用いる波形表の作成（5.3 節）を行う．

木の揺れ動きのシミュレーションの 1 ステップ内の流れとしては，まず重力や風などの外力を加え，計算点の速度・位置をそれにより更新する．次に移動後の計算点の位置を，位置ベース法（4.1 節）の制約条件を木構造に適用したもの（4.3 節）に従って修正する．これは制約を満たすまでステップ内で反復計算される．その後移動後の各葉の位置から近傍ノード探索を用いた衝突判定（4.4 節）を行い，最後に速度と衝突状態を更新する．

音は揺れ動きのシミュレーションと連動しながら再生される．衝突音は衝突判定を行い接触状態が変化した際に，前処理で行ったモード解析で計算された固有周波数を用いて生成し再生され（5.1.4 項），摩擦音と風切り音は用意した 1/f ノイズと波形表に風による音の変化を加えながら再生される（5.2.1 項，5.3.2 項）．

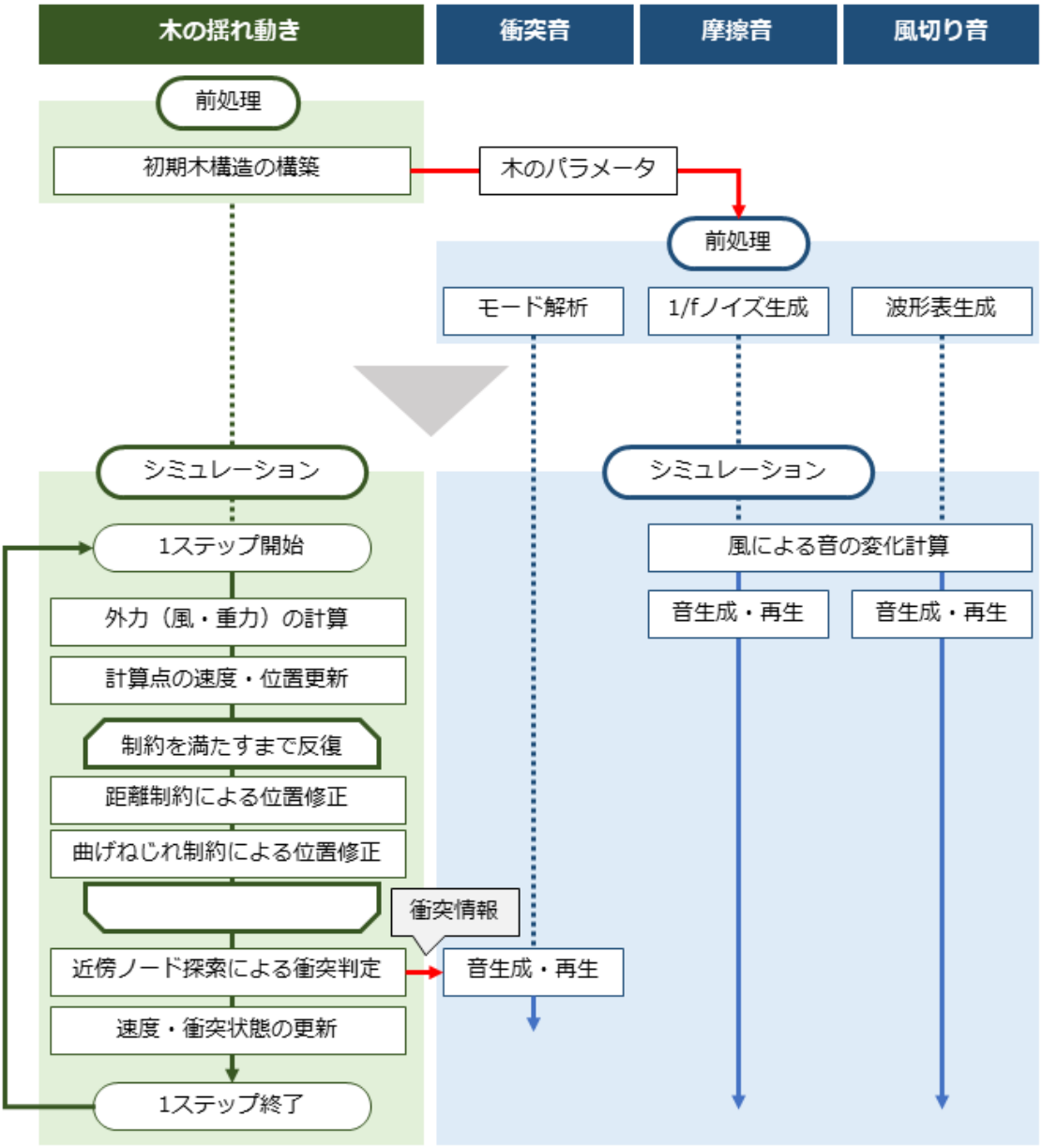

図 3.1: システム全体の構成

3.2 ユーザインターフェース

本論文で提案する樹木シミュレーションに基づく環境音合成システムでは樹木の各種パラメータや葉の形などを指定可能であり，その値に応じて樹木の形状の構築や音の合成を行う．本節では指定できるパラメータと入力画面のインターフェース（図 3.2）について説明する．

まず，樹木の形状に関わるパラメータは樹木の高さ $tree\_h$，幅 $tree\_w$，葉の枚数 $leaf\ num$ であり，これらは図 3.2 で示すようにスライドバーで任意の範囲内で値を指定できる．また，これらの値から木構造を構築する方法は 4.2 節で述べる．

次に葉に関わるパラメータは葉の大きさ $leaf\_rad$ と形状である．葉の大きさも樹木パラメータと同様にスライドバーで指定でき衝突判定（4.4 節）に利用する．葉の形状は画面内に 20 × 30 ピクセルの入力エリアがあり（図 3.2 右下付近），ユーザは葉の外形形状をマウスドラッグ操作により指定し，システムはそこから内部を埋めたピクセル形状を生成する．入力された木の葉のピクセル形状から木の葉ボクセルモデルを作成し，衝突音生成で用いる弾性行列を作成する（5.1.3 項）．

樹木を揺らす風に関するパラメータとしては，風の強さ $wind\ pow$ を数字で入力でき，3 次元の方向矢印で向き $wind\ direction$ を指定できる．風の強さの値は揺れ動きの外力となる他，摩擦音の再生（5.2.1 項）や風切り音の再生（5.3.2 項）に反映される．

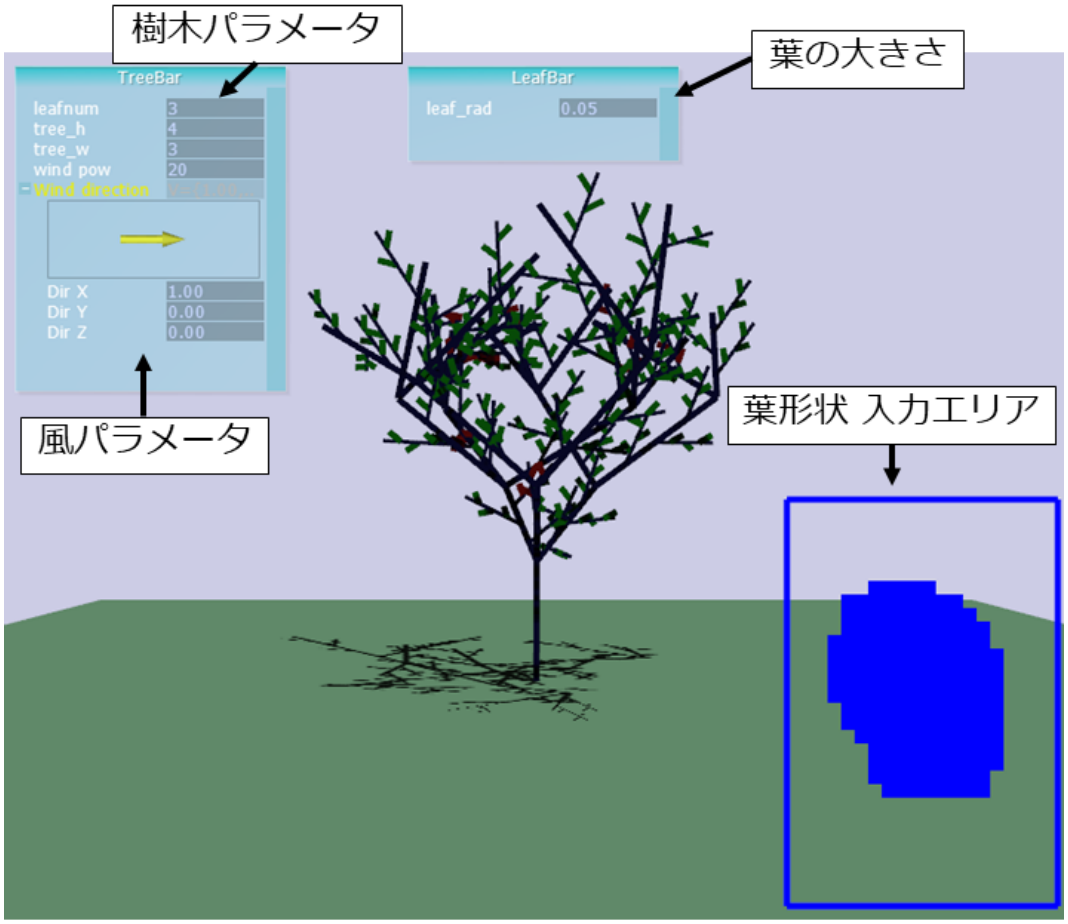

図 3.2: 入力インタフェース

第4章　木の揺れ動きのシミュレーション

4.1　位置ベース法

樹木の揺れ動きのシーンと音を同時に生成するには，揺れ動きのシミュレーションにおいて細かい枝の揺れ動きや葉の衝突まで高速に計算する必要がある．この揺れ動きの再現のために本研究では安定かつ高速な物理シミュレーション手法である位置ベース法 (Position Based Dynamics) [1] を用いる．従来の力学ベース法では，ニュートンの運動方程式に基づいて計算された加速度から時間積分により速度・位置を求めているため，時間経過と共に誤差が蓄積してしまい，シミュレーションが不安定になる場合があるが，位置ベース法は従来の力学ベース法と異なり制約条件を用いて物体の位置を直接変更する方法であるため誤差の蓄積は発生せず，安定かつ制御性の高いシミュレーションを行うことができる．

4.1.1　位置ベース法の理論

位置ベース法における制約条件とは，シミュレーションする物体を計算点の集合として離散化した時の計算点の位置が満たすべき条件のことである．移動・変形する物体内の計算点 p の位置 $\boldsymbol{p}$ に対する制約条件 $C(\boldsymbol{p})$ を考え，式 (4.1) が常に満たされているとする．

$$C(\boldsymbol{p}) = 0 \tag{4.1}$$

1 ステップ後の計算点 p の移動量を $\Delta\boldsymbol{p}$ とすると，制約条件は移動後も満たしていると考えられるため，式 (4.2) が得られる．

$$C(\boldsymbol{p} + \Delta\boldsymbol{p}) = 0 \tag{4.2}$$

この式 (4.2) をテイラー展開し，二次以降の項を無視することで式 (4.3) に近似することができる．

$$C(\boldsymbol{p} + \Delta\boldsymbol{p}) \approx C(\boldsymbol{p}) + \nabla\boldsymbol{p}C(\boldsymbol{p}) \cdot \Delta\boldsymbol{p} = 0 \tag{4.3}$$

ここから $\Delta\boldsymbol{p}$ に関する方程式を得ることを考える．制約条件 $C(\boldsymbol{p})$ は運動物体内部の状態に対する制約条件なので，物体全体の平行及び回転運動とは独立した条件であり，移動量 $\Delta\boldsymbol{p}$ は元の運動の平行移動量と回転に影響を及ぼさないと考えられる．これは，ベクトルの傾き ∇C が運動方向に対して垂直な方向の時に成り立つため，$\Delta\boldsymbol{p}$ の方向を傾き $\nabla_{\boldsymbol{p}}C(\boldsymbol{p})$ とするとラグランジュ乗数 λ を用いて式 (4.4) で表される．

$$\Delta\boldsymbol{p} = \lambda\nabla_{\boldsymbol{p}}C(\boldsymbol{p}) \tag{4.4}$$

さらに，これを式 (4.3) に代入すると λ の値が得られる．

$$\lambda = -\frac{C(\boldsymbol{p})}{||\nabla_{\boldsymbol{p}}C(\boldsymbol{p})||^2} \tag{4.5}$$

これらの制約条件から $C(\boldsymbol{p})$ と $\nabla_{\boldsymbol{p}}C(\boldsymbol{p})$ の値が計算できるため，求めた λ を式 (4.5) に代入することで，制約を満たすための位置修正量 $\Delta\boldsymbol{p}$ を以下の式 (4.6) より求めることができる．

$$\Delta\boldsymbol{p} = -\frac{C(\boldsymbol{p})}{||\nabla_{\boldsymbol{p}}C(\boldsymbol{p})||^2}\nabla_{\boldsymbol{p}}C(\boldsymbol{p}) \tag{4.6}$$

この $\Delta\boldsymbol{p}$ を用いて計算点位置を修正すれば良い. 注意として $\Delta\boldsymbol{p}$ は制約条件を満たすための位置修正量であり，式 (4.5), (4.6) は $C(\boldsymbol{p}) \neq 0$ の時に適用する．

ここまでの考え方は 1 つの計算点のみについてのものであるが，実際のシミュレーションでは物体は複数の計算点の集合として離散化されており，それら計算点の集合に対して位置ベース法を適用するため，ここまでの考え方を拡張し，複数の計算点に用いる．$\boldsymbol{p}_1, ..., \boldsymbol{p}_n$ の n 個の計算点に対して制約条件 $C(\boldsymbol{p}_1, ..., \boldsymbol{p}_n)$ が与えられたとすると，計算点の位置 $\boldsymbol{p}_i$ の位置修正量は以下の式 (4.7) のように表され，この時の λ_i は式 (4.8) になる．

$$\Delta\boldsymbol{p}_i = \lambda_i\nabla\boldsymbol{p}_iC_i(\boldsymbol{p}_1, ..., \boldsymbol{p}_n) \tag{4.7}$$

$$\lambda_i = -\frac{C_i(\boldsymbol{p}_1, ..., \boldsymbol{p}_n)}{\Sigma_j||\nabla_{\boldsymbol{p}_j}C_i(\boldsymbol{p}_1, ..., \boldsymbol{p}_n)||^2} \tag{4.8}$$

ただし，これらの式は計算点の重さがすべて等しいことを仮定している．各計算点 $\boldsymbol{p}_1, ..., \boldsymbol{p}_n$ の質量が一定ではない場合はその重みを式に導入する必要がある．計算点 p_i の質量を m_i として，質量の逆数の重み $w_i = \frac{1}{m_i}$ を位置修正量にかけることで重み付きの修正式は以下の式 (4.9) のように求められる．この時の λ_i は式 (4.10) になる．

$$\Delta\boldsymbol{p}_i = \lambda_i w_i\nabla\boldsymbol{p}_iC_i(\boldsymbol{p}_1, ..., \boldsymbol{p}_n) \tag{4.9}$$

$$\lambda_i = -\frac{C_i(\boldsymbol{p}_1, ..., \boldsymbol{p}_n)}{\Sigma_j w_j||\nabla_{\boldsymbol{p}_j}C_i(\boldsymbol{p}_1, ..., \boldsymbol{p}_n)||^2} \tag{4.10}$$

なお，計算点が互いに影響する制約条件の場合，複数の計算点すべてが 1 回の修正で制約を満たすことは難しいので，制約を条件を満たすまで反復処理することが必要になる．

位置修正後は，1 ステップ前の計算点の位置 $\boldsymbol{p}^{prev}$ からの移動量により，以下の式 (4.11) で速度 $\boldsymbol{v}$ を更新する．ここでの Δt はタイムステップ幅である．

$$\boldsymbol{v} \leftarrow \frac{\boldsymbol{p} - \boldsymbol{p}^{prev}}{\Delta t} \tag{4.11}$$

位置ベース法では図 4.1 に示すように，速度 $\boldsymbol{v}$ による位置推定，制約に基づく位置修正，そして最後に速度更新を繰り返すことで処理が進む.

4.1.2 細長い弾性体への位置ベース法の適用

本論文では，位置ベース法を弾性体ロッドに適用した Kugelstadt ら [17] の手法を応用し，樹木を弾性体として扱い，木構造に制約条件を掛けることで弾性変形のシミュレーションを行う．そのため先に Kugelstadt らの手法での弾性体の扱いと，用いる制約条件である距離制約と曲げ・ねじれ制約について説明する．

Kugelstadt らの手法では細長いロッド形状の弾性体を図 4.2 のようにエッジとノードで離散化する．各ノードは座標 $\boldsymbol{p}$ と質量 m を持ち，さらにノード間の各エッジは姿勢を表す四元数 q を保持している．四元数はベクトル部分とスカラー部分からなる虚数表現を拡張したものであり，三次元空間中での回転を記述することができる．

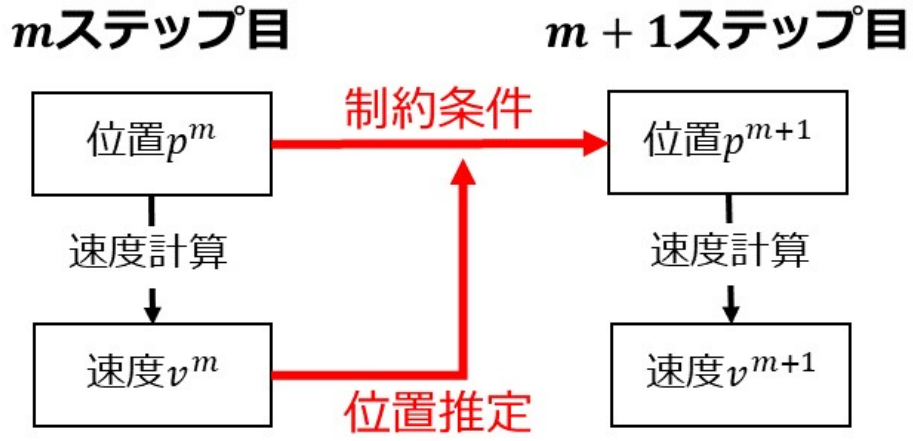

図 4.1: 位置ベース法の計算の流れ

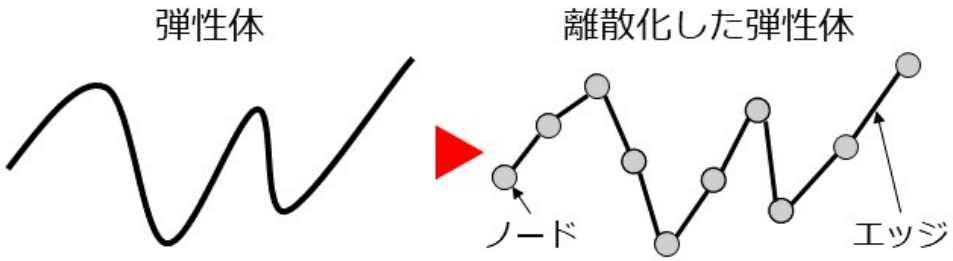

図 4.2: 弾性体の離散化

距離制約

距離制約は図 4.3 に示すように，計算点 p_1 の位置 $\boldsymbol{p}_1$ と計算点 p_2 の位置 $\boldsymbol{p}_2$ の距離を一定に保つ制約条件である．初期状態の 2 点間の距離を l とすると $\boldsymbol{p}_1$，$\boldsymbol{p}_2$ 間の距離は $||\boldsymbol{p}_2 - \boldsymbol{p}_1||$ であるため，制約条件 $C^{stretch}$ は式 (4.12) のように定義できる．

$$C^{stretch}(\boldsymbol{p}_1, \boldsymbol{p}_2) = ||\boldsymbol{p}_2 - \boldsymbol{p}_1|| - l = 0 \tag{4.12}$$

それぞれの計算点 $\boldsymbol{p}_1$，$\boldsymbol{p}_2$ で制約条件を微分することで，制約条件の空間勾配は以下の 2 つの式 (4.13)，式 (4.16) のように計算できる．

$$\nabla \boldsymbol{p}_1 C^{stretch}(\boldsymbol{p}_1, \boldsymbol{p}_2) = -\frac{\boldsymbol{p}_2 - \boldsymbol{p}_1}{||\boldsymbol{p}_2 - \boldsymbol{p}_1||} \tag{4.13}$$

$$\nabla \boldsymbol{p}_2 C^{stretch}(\boldsymbol{p}_1, \boldsymbol{p}_2) = +\frac{\boldsymbol{p}_2 - \boldsymbol{p}_1}{||\boldsymbol{p}_2 - \boldsymbol{p}_1||} \tag{4.14}$$

そして，式 (4.10) より，係数 $\lambda^{stretch}$ は以下の式 (4.17) のように決まる．

$$\lambda^{stretch} = -\frac{1}{w_1 + w_2}(||\boldsymbol{p}_2 - \boldsymbol{p}_1|| - l) \tag{4.15}$$

これらを用いて最終的な制約条件 $C^{stretch}$ の位置修正式が以下のように計算できる．なお，w_1，w_2 はそれぞれ $\boldsymbol{p}_1$，$\boldsymbol{p}_2$ の質量の逆数である．

$$\Delta \boldsymbol{p}_1 = +\frac{w_1}{w_1 + w_2}(||\boldsymbol{p}_2 - \boldsymbol{p}_1|| - l)\frac{\boldsymbol{p}_2 - \boldsymbol{p}_1}{||\boldsymbol{p}_2 - \boldsymbol{p}_1||} \tag{4.16}$$

$$\Delta \boldsymbol{p}_1 = -\frac{w_1}{w_1 + w_2}(||\boldsymbol{p}_2 - \boldsymbol{p}_1|| - l)\frac{\boldsymbol{p}_2 - \boldsymbol{p}_1}{||\boldsymbol{p}_2 - \boldsymbol{p}_1||} \tag{4.17}$$

曲げ・ねじれ制約

曲げ・ねじれ制約は，弾性体の持つ外力により曲げやねじれの生じる変形が起こった際に元に戻ろうとする性質を位置ベース法の制約として定式化したものである．樹木においても風などにより曲がった木は風に反発して元に戻ろうとする．曲げ・ねじれ制約はつまり初期形状と現在の形状が一致するという制約であるが，4.1 項で述べたように位置ベース法の制約条件は運動物体内部の状態に対する制約条件なので各エッジの姿勢 q は物体全体における平行移動や回転運動からは独立しており，そのまま利用することはできない．そこでエッジの姿勢 q に直接制約を適用するのではなく，ノードを介して接続する 2 つのエッジ間の姿勢変化を表す Darboux ベクトルを使用して制約条件を設定する．Darboux ベクトルの詳しい原理については [17] を参照してほしい．Darboux ベクトル $\boldsymbol{\Omega}$ は図 4.3 に示すような，2 つのエッジの長さ l_{q1q2} と姿勢 q_1, q_2 から次のように計算する．

$$\begin{aligned} \boldsymbol{\Omega} &= \Im(\Omega^*) \\ \Omega^* &= \frac{1}{l_{q1q2}}(\bar{q_1} + \bar{q_2})(q_1 - q_2) \\ &= \frac{2}{l_{q1q2}}\bar{q_1}q_2 \end{aligned} \tag{4.18}$$

ここでの $\Im(\Omega^*)$ は四元数 Ω^* のベクトル部分を表す．2 つのエッジそれぞれの姿勢 q_1, q_2 の初期状態での曲がり量・ねじれ量，つまり，維持する形状を Ω^{rest} とし，現在の形状を Ω とすると，弾性体の変化量は Darboux ベクトルの差 $\Omega - \Omega^{rest}$ で表せ，以下の式 (4.19) に示すように，この値を 0 にする制約が曲げ・ねじれ制約となる．

$$C_{bending\_twist}(q_1, q_2) = \Im(\Omega^* - s\Omega^{*rest}) = \boldsymbol{\Omega} - s\boldsymbol{\Omega}^{rest} = 0 \tag{4.19}$$

$$s = \begin{cases} +1 \text{ if } ||\Omega^* - \Omega^{rest}||^2 < ||\Omega^* + \Omega^{rest}||^2 \\ -1 \text{ if } ||\Omega^* - \Omega^{rest}||^2 > ||\Omega^* + \Omega^{rest}||^2 \end{cases} \tag{4.20}$$

式 (4.20) に示した s は符号係数であり，同じ回転を表す四元数 q と $-q$ によって実際の制約条件で修正の方向が逆方向の回転にならないようにしている．なお，式 (4.18) では Darboux ベクトルを求める際に $\frac{2}{l_{q_1 q_2}}$ を掛けているが，今回のように変形量（Darboux ベクトルの差）を 0 にする場合においてはこの係数を無視できる．そのため Ω^* は式 (4.21) のように計算できる．

$$\Omega^* = \bar{q_1}q_2 \tag{4.21}$$

ここまでの条件式より，曲げ・ねじれ制約の修正式は以下のように表される．

$$\Delta q_1 = +\frac{Kw_{q1}}{w_{q1} + w_{q2}}q_2(\Omega - s\Omega^{rest}) \tag{4.22}$$

$$\Delta q_1 = -\frac{K w_{q1}}{w_{q1} + w_{q2}} q_2 (\Omega - s\Omega^{rest}) \tag{4.23}$$

$$s = \begin{cases} +1 \text{ if } ||\Omega^* - \Omega^{rest}||^2 < ||\Omega^* + \Omega^{rest}||^2 \\ -1 \text{ if } ||\Omega^* - \Omega^{rest}||^2 > ||\Omega^* + \Omega^{rest}||^2 \end{cases} \tag{4.24}$$

なお，w_{q_1}，w_{q_2} はそれぞれ q_1，q_2 の重み（エッジの質量の逆数）であり，K は弾性を表す係数（$K \in [0, 1]$）である．

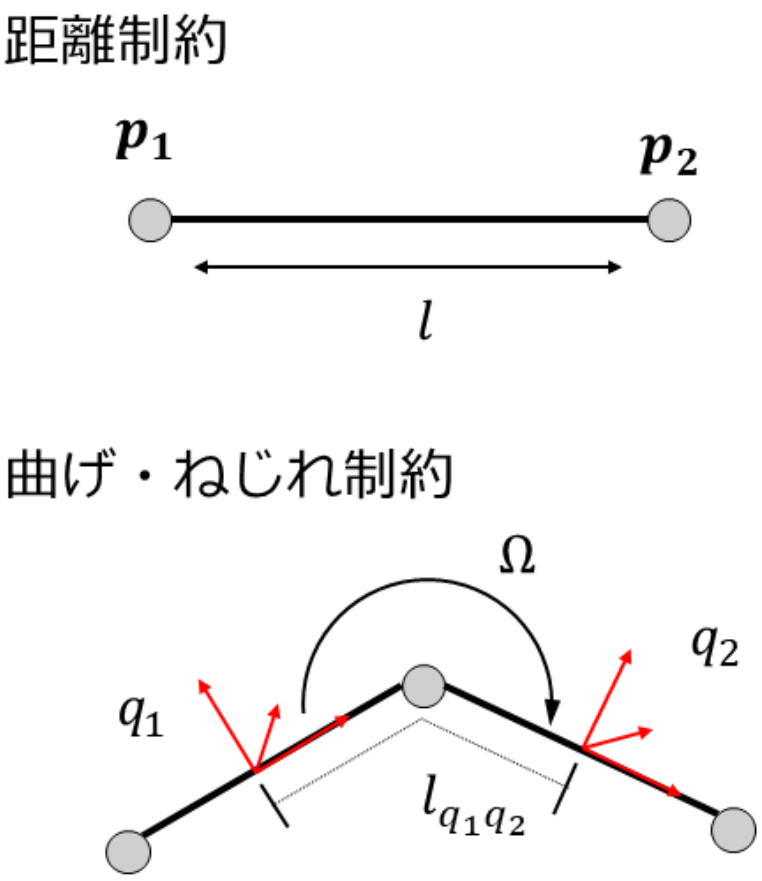

図 4.3: 位置ベース法の制約条件

4.2　離散化した木構造の作成

本論文では位置ベース法を用いて樹木の揺れ動きをシミュレーションするが，4.1.2 節で説明した Kugelstadt ら [17] らの手法を応用するためには，木構造の弾性体である樹木をノードとエッジに離散化する必要がある．そのため本節では樹木の離散化の方法と，シミュレータ内部で樹木を構成する方法を説明する．

本手法で扱う樹木は大きく分けて低木と高木の 2 種類あり，低木は図 4.4 のような植え込みなどに用いられる椿などの木を，高木は図 4.5 のような街路樹や公園の木でみられる樫などの木を表す．本論文では，図 4.6 に示すように低木を木構造の幹から直接葉が生えている木，高木を木構造の幹から枝が生え，そこから葉が生えている木と考える．

図 4.4: 低木の例（著者撮影）

図 4.5: 高木の例（著者撮影）

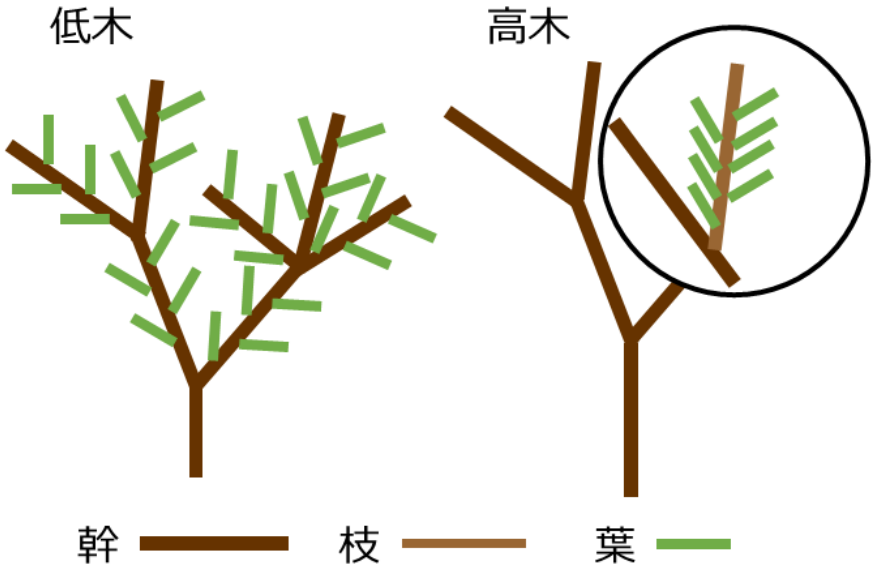

図 4.6: 低木と高木の構造

まず，幹については木全体の形状を決定するものであり，根元から木構造を形成する．こ

の木構造において，先端と枝分かれする点をノード，それらをつなぐ線分をエッジとする．次に枝については両端点をノード，それらをつなぐ線分をエッジとし，枝は幹から生えているため，片方のノードは幹の線分上にあるようにする．そして葉については生えている幹もしくは枝上の端点と葉の中心をノード，それらをつなぐ線分をエッジとする．ここで葉の根元ではない方のノードが葉先ではなく中心に置かれている理由は，葉の揺れ動きは葉の根元から中心までのエッジで充分制御できることと，4.4 項で後述する葉の衝突判定において，葉の中心ノードの座標を葉の位置として利用することである．

シミュレータ内で樹木は枝分かれ本数や方向にランダムな要素を含みながら，毎回異なる初期形状が構築される．ここでは枝を含む高木の初期形状の構築プロセスを説明する．なお，低木の初期形状の構築はこのプロセスから枝を生成するステップを省略したものである．

高木の初期形状の構築は以下のステップから構成され，図 4.7 はこの流れを図で表したものである．

(1) 根・地面・1 階層目にあたる 3 点のノードを定義する．

(2) 1 階層目のノードを始点とし，枝分かれを繰り返しながら指定された階層数まで幹を構築する．

(3) 構築された幹のそれぞれから枝を生やす．

(4) すべての枝から葉を生やす．

(1) では図 4.7 に示すように根，地面，1 層目のノードの 3 点の位置を決定する．この時これらのノードは軸となるため傾かないように固定する．この 1 階層目のノードの高さは実際の樹木の形に倣って，低木の時はやや低めの $0.3m$，高木の時はやや高めの $0.8m$ に設定している．本シミュレータ内の樹木は階層構造を構成することを想定しており，(2) では指定の階層数まで 1 階層目のノードを始点とし，枝分かれを繰り返しながら，ノードとエッジを追加することで階層的な木構造を構築する．このときの階層数は 3.2 節で説明したユーザの指定した値 $tree\_h$ を用いる．枝分かれの本数もユーザの指定した値 $tree\_w$ に基づいており，2 本から $tree\_w$ 本の間でランダムな本数 1 つのノードから枝分かれしていく．新しい幹の先端ノードの位置は x,y 方向に $-0.8m$ から $0.8m$，y 方向に $0.4m$ から $0.6m$ の範囲内で決定される．(3) では実際の樹木と同様に根・地面・1 層目にあたる 3 点のノードからなる最初の 2 辺の幹エッジからは枝が生えないと考え，幹として構築された木構造の根本から 4 ノード目以降のノードが追加される度に生まれるエッジ，つまり 3 辺目以降の幹エッジすべてに枝を生やしていく．1 辺の幹から生える枝の本数がユーザの指定した $leaf\ num$ 本になるように，幹エッジを $m+1$ 等分する両端を除く m 箇所の点に根元ノードを追加し，そこから図 4.8 に示すように，幹方向の単位ベクトルとその法線ベクトルを求め，その 2 ベクトルの間，つまり幹に対して 45° の角度で長さ $0.3m$ の枝先端ノードを追加する．なお，法線ベクトルはランダムな方向ベクトルと幹方向ベクトルの外積から計算する．(4) での葉の追加は枝と同様の方法で，枝に対して 45° の角度で，ユーザのした葉の大きさ $leaf\_rad$ の大きさの葉を追加する．

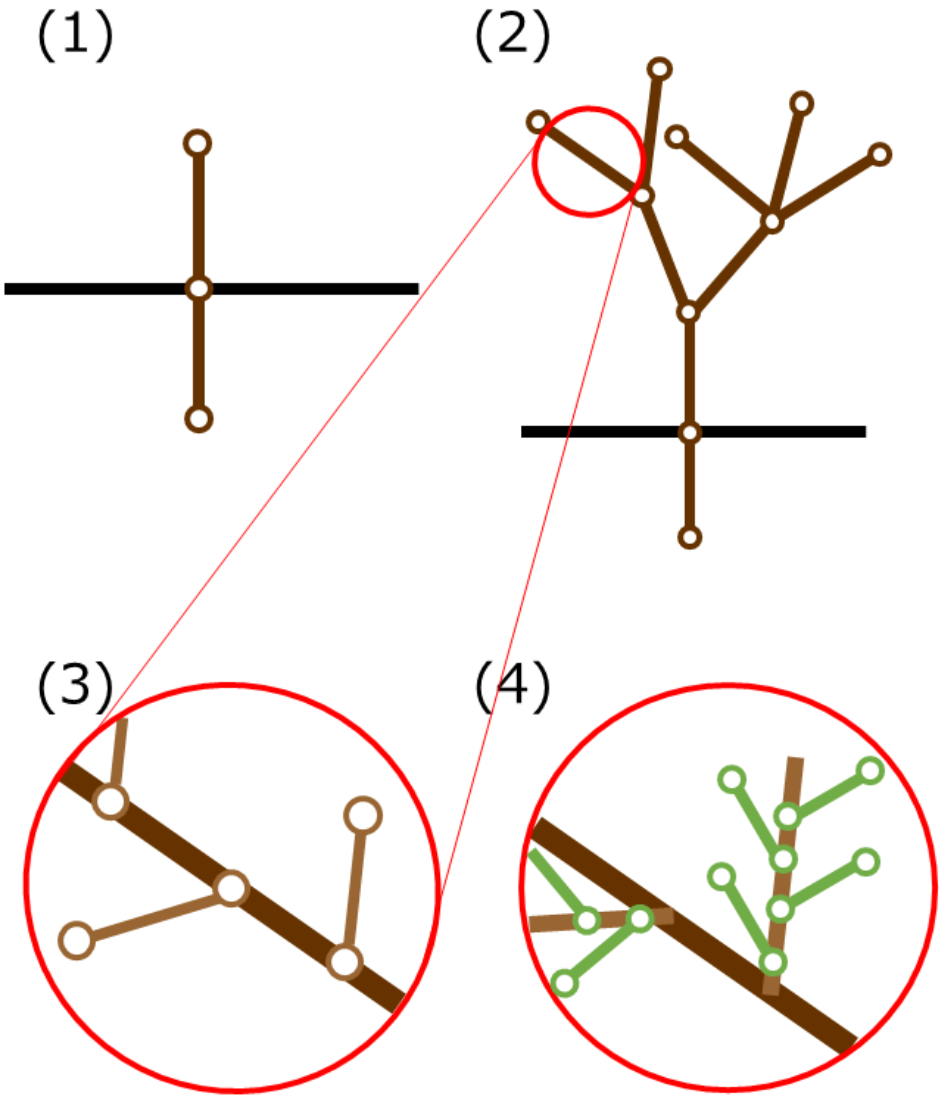

図 4.7: 初期形状構築の流れ

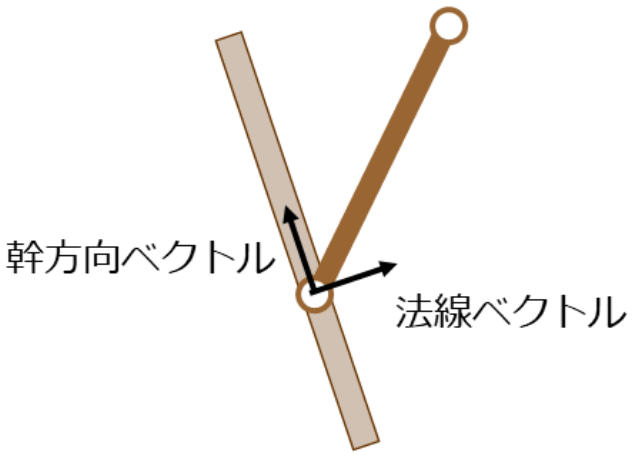

図 4.8: 枝の方向

4.3 木構造への位置ベース法の適用

作成した木構造に位置ベース法を適用することで，木の揺れ動きをシミュレーションする．まず，幹に関しては隣り合う親子関係のノード間に距離制約を，親子関係にある 2 エッジ間に曲げ・ねじれ制約を適用する．これにより木全体が概形を保った状態で揺れ動く様子を再現する．枝に関しては幹上の根元ノードとそこから伸びた先の先端ノードに距離制約を，葉に関しては幹もしくは枝上の根元ノードと葉の中心ノードに距離制約を適用し，これら 2 点からなるエッジと根元ノードのあるエッジとの間に曲げ・ねじれ制約を適用する．

Kugelstadt らの手法で扱われているようなロッド形状の 1 本の細い弾性体であればノードが一直線に並ぶため，n 番目のノードと $n+1$ 番目のノード間の距離制約と，m 番目のエッジと $m+1$ 番目のエッジ間の曲げ・ねじれ制約を考えればよいが，本手法のように枝分かれのある木構造かつ，毎回違った形の木構造をランダムで形成する場合は，制約条件をかけるノードやエッジの組み合わせを木構造を構築する段階で記録しておく必要がある．そのため，既存手法でノードは位置座標や質量という情報を保持していたが，本手法ではそれに加えて，親ノードの番号を保持させる．木構造では親子間に制約条件を適用するため，これにより制約条件を適用する適切なノード，エッジの組み合わせを参照することができる．

また，式 (4.12)，(4.19) の制約で用いる初期状態での距離，曲げ・ねじれの情報については，子ノードの側に親ノードとの距離，曲げ・ねじれを保持させる．これは１つのノードに子ノードは複数ある可能性が考えられるが，親ノードは必ず１つである木構造を用いるためである．枝や葉に関する曲げ・ねじれ制約も，1 つの幹・枝から複数の枝・葉が伸びていくため，子ノードである枝・葉側に初期状態での情報を保持させる．既存手法ではエッジ同士が端点で接続し角を成す隣り合うエッジ間にのみ曲げ・ねじれ制約を適用していたが，このようにエッジが付いている親エッジの番号を記録しておけば，２つのエッジの姿勢とその成す角から位置修正式は計算できるため，隣接していないエッジ間にも曲げ・ねじれ制約が適用できる．

ここまでに説明した木構造における制約条件と適用対象をまとめると図 4.9 に示すように，幹の場合は隣り合うノード間に，枝と葉の場合は根元ノードと枝先端ノードもしくは葉中心ノード間に距離制約を適用し，隣り合う幹同士，枝とその枝が生えている幹，葉とその葉が生えている幹もしくは枝に曲げ・ねじれ制約を適用する．

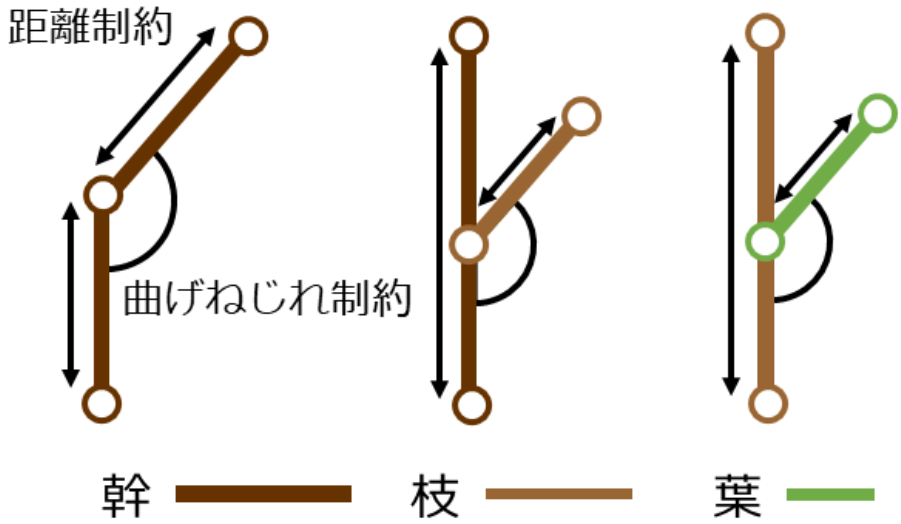

図 4.9: 木構造における制約条件

4.4 葉の衝突判定

樹木の揺れ動きのシミュレーションから衝突音を鳴らすタイミングを取得するために，葉の衝突判定を行い，葉の接触状態を取得する必要があるが，葉の詳細形状をポリゴンで表し，ポリゴン同士の衝突判定を行うことは大量の木の葉を扱う樹木のような場合は計算コストが高く難しい．そのため，今回扱うような樹木においては葉の揺れや衝突が他の葉や枝の動きに与える影響は極めて少ないと考え，図 4.10 に示すように葉を，中心ノードを中心とした球で近似し，粒子法で用いられる近傍ノード探索 [18] を用いて，一定の範囲内に他の葉中心ノードが存在した場合衝突していると扱うことで，大量の葉の衝突判定を高速に実行する．

4.4.1 近傍ノード探索

木の葉をノードで近似したとしても，シミュレーション空間内の葉ノードすべてを全探索していると計算量はノード数の 2 乗となるため計算コストが高い．そのため，より効率的に周囲のノードを探索するためにグリッド構造に基づく近傍ノード探索を用いる．

探索の流れとしては，まず図 4.11 のようにシミュレーション空間を格子幅が影響半径 h である格子（グリッドセル）で分割し，次に各グリッドセル内に存在するノードを登録する．

そして各ノードに対して近傍探索を行うが，その際そのノードが存在しているグリッドと，隣接する（図 4.11 の灰色のグリッド）だけを探索すれば良い．これは，格子幅が h なので上記範囲のさらに外側のセルにいる粒子との距離は必ず h より大きくなるためである．なお，本手法では葉を近似した球の直径を影響半径としており，この値はユーザが指定できる．

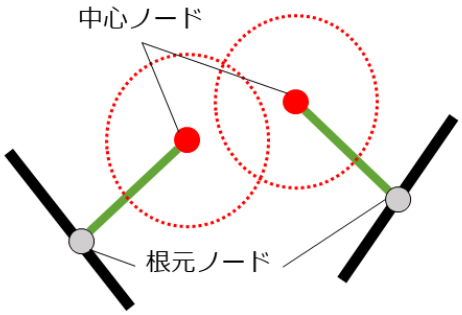

図 4.10: 葉の構成

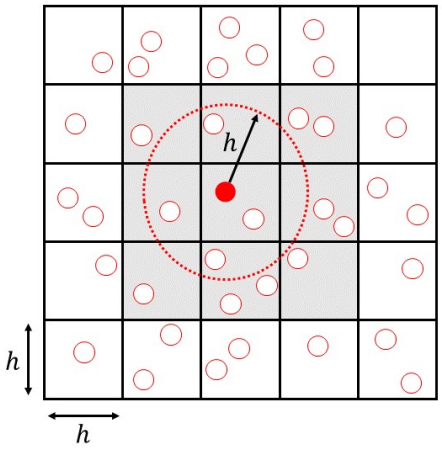

図 4.11: 等間隔格子による近傍探索手法

第5章　音生成

5.1　衝突音生成

物体の衝突音は，物体が衝突の衝撃によって振動することによって発生する．音は振動の仕方によって決まり，例えば木琴や鉄琴のような音板打楽器においては叩き方に関わらず材質と鍵盤の大きさで音程が定まるが，これは物体ごとに材質や形状によって決定される固有の共振周波数があるためである．この共振周波数を形状と材質，特に境界条件とヤング率から求めるものがモード解析である. 本節では本研究で用いるモード解析の考え方からそれに基づく衝突音生成・再生についてまで述べる．なお，衝突音，摩擦音については著者の卒業論文 [19] の手法と同様のものを用いている．

5.1.1　モード解析

本論文では Raghuvanshi ら [3] の手法に従い物体をバネ-質点系で近似することで高速にモード解析を行う．

まず，ある物体に外力が加わったときの物体全体の次の力の釣り合いの式としての運動方程式から考える．

$$M\frac{d^2\boldsymbol{r}}{dt^2} + (\alpha_1 M + \alpha_2 K)\frac{d\boldsymbol{r}}{dt} + K\boldsymbol{r} = \boldsymbol{f} \tag{5.1}$$

ここで M は物体の質量行列, K は物体の弾性を表す行列, $\boldsymbol{f}$ は外力, $\boldsymbol{r}$ は物体を構成する質点の移動量をまとめたベクトル,α_1, α_2は減衰定数である. 本論文ではバネ-質点系を用いて葉をモデル化するため，式 (5.1) は図 5.1 に示すように物体を弾性体とし，質点とそれを結ぶバネの集合で近似したときの，各質点における元の位置からの変位量に関する式となっている．質点 i の変位量を r_i，質点数を n とすると，式 (5.1) 中の $\boldsymbol{r}$ は $\boldsymbol{r} = (r_1, r_2, \ldots, r_n)^T$ と表され，M は各質点の質量を対角成分に持つ対角行列となる．K は物体全体の剛性を表す対称行列で，詳しくは次項で述べる．ここで K の固有ベクトルを並べた行列を G, K の固有値を対角要素とする対角行列を D とすると, K は次のようにおける．

$$K = GDG^{-1} \tag{5.2}$$

これを式 (5.1) に代入して両辺に G_{-1} を掛けると，

$$G^{-1}M\frac{d^2\boldsymbol{r}}{dt^2} + (\alpha_1 G^{-1}M + \alpha_2 DG^{-1})\frac{d\boldsymbol{r}}{dt} + DG^{-1}\boldsymbol{r} = G_{-1}\boldsymbol{f} \tag{5.3}$$

となり，$\boldsymbol{z} = G^{-1}r$ とおき，整理すると

$$M\frac{d^2\boldsymbol{z}}{dt^2} + (\alpha_1 M + \alpha_2 D)\frac{d\boldsymbol{z}}{dt} + D\boldsymbol{z} = G^{-1}\boldsymbol{f} \tag{5.4}$$

となる．D は対角行列なので式 (5.4) を各質点の変位 z_iに関する n 個の微分方程式に分解し，各式を解析的に解くことで各モードの音圧 $z_i(t)$ を次のように求めることができ，微分方程

式を解くときに設定した$\omega_i^{\pm}$が固有振動数となる．

$$z_i(t) = c_i e^{\omega_i^+ t} + \overline{c_i} e^{\omega_i^- t} \tag{5.5}$$

$$\omega_i^{\pm} = \frac{-(\alpha_1\lambda_i + \alpha_2) \pm \sqrt{(\alpha_1\lambda_i + \alpha_2)^2 - 4\lambda_i}}{2} \tag{5.6}$$

ここでのc_iは定数であり，振動の式として表した場合の衝撃の大きさ，つまり各振動モードの振幅に相当すると考えられる.λ_iは行列Kのi番目の固有値である．そして式(5.5)をオイラーの公式$e^{i\theta} = \cos\theta + i\sin\theta$を用いて三角関数に展開すると，

$$z_i = c_i e^{tRe(\omega_i)} \sin(t|Im(\omega_i)|) \tag{5.7}$$

となり，これが衝突による音の波の式である．音波の振幅c_iについてはO'Brienら[2]らの方法と同様に時間ステップ幅Δtの間に$\boldsymbol{g} = G_{-1}\boldsymbol{f}$の大きさのインパルスが加えられたとして，

$$c_i = \frac{2\Delta t g_i}{|Im(\omega_i)|} \tag{5.8}$$

とする．よって最終的に衝突音を表す波の式は，

$$z_i = \frac{2\Delta t g_i}{|Im(\omega_i)|} e^{tRe(\omega_i)} \sin(t|Im(\omega_i)|) \tag{5.9}$$

となる．ここでの$\Delta t g_i$が衝突の瞬間にかかる力で，$Re(\omega_i)$が式(5.6)の実部，$Im(\omega_i)$が虚部を表す．式(5.6)ではλ_iの大きさによっては虚部がなく波の式とならない場合もある．その場合はそのモードについては音は鳴っていないと考え，そのモードの計算をスキップする．

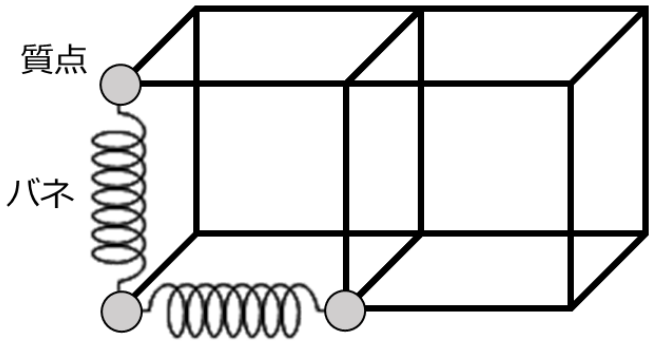

図 5.1: バネ-質点系で近似した物体

5.1.2 弾性行列Kの計算

モード解析を用いて衝突音を計算するには，式(5.6)より，物体全体の硬さを表す行列である弾性行列Kの固有値を求める必要がある．計算されたn個の固有値(つまりn個の振動モード)全てについて式(5.9)を計算しそれらを合成することで最終的な音の波を計算することができる．バネ質点系でモデル化された物体の弾性行列K及びその固有値の計算手順は以下のような流れになる．

(1) フックの法則から各質点における力の釣り合いの式を立てる

(2) (1) の式より物体全体のバネ定数の行列 (弾性行列 K) を作成する

(3) (2) で作成した行列 K の固有値を計算する

この流れを図 5.2 のような，バネが 4 本で質点が 4 個の単純なバネ-質点系を例に説明する．

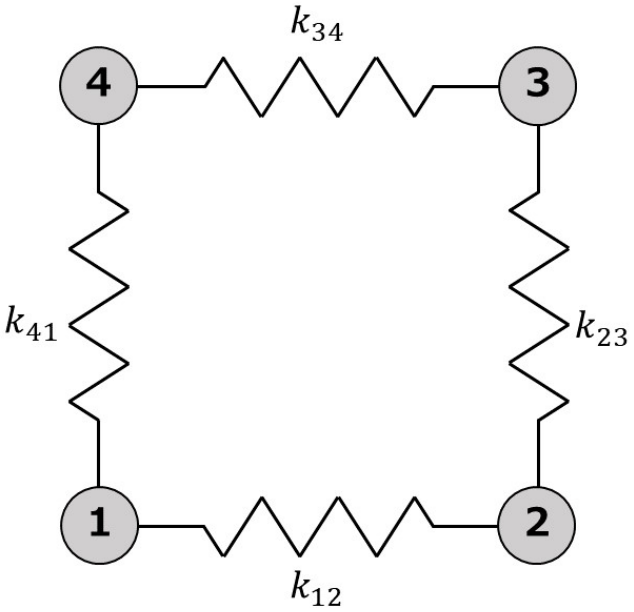

図 5.2: バネ-質点系からなる四角形

まず (1) の力の釣り合いの式は次のようになる．

$$
\begin{aligned}
F_{1x} &= k_{12}x_1 - k_{12}x_2 && (5.10)\\
F_{2x} &= -k_{12}x_1 + k_{12}x_2 && (5.11)\\
F_{3x} &= k_{34}x_3 - k_{34}x_4 && (5.12)\\
F_{4x} &= -k_{34}x_3 + k_{34}x_4 && (5.13)\\
F_{1y} &= k_{14}y_1 - k_{14}y_4 && (5.14)\\
F_{2y} &= k_{23}y_2 - k_{23}y_3 && (5.15)\\
F_{3y} &= -k_{23}y_2 + k_{23}y_3 && (5.16)\\
F_{4y} &= -k_{14}y_1 + k_{14}y_4 && (5.17)
\end{aligned}
$$

ここで，F_{nx} は質点 n における x 方向の外力であり，F_{my} は質点 m における y 方向の外力である．また x_n,y_m はそれぞれ質点 n, m における x 方向，y 方向の変位，k_{nm} は質点 n と質点 m 間のバネのバネ定数である．

これらの式から (2) のバネ定数の行列を作成する．ここで重要なのは物体の材質（弾性力）を表すバネ定数 k と形状を表すバネの接続している質点の情報である．

先ほどの図 5.2 における力の釣り合いの式を線形システムとして行列を用いて表すと次のようになる．

$$\begin{pmatrix} F_{1x} \\ F_{2x} \\ F_{3x} \\ F_{4x} \\ F_{1y} \\ F_{2y} \\ F_{3y} \\ F_{4y} \end{pmatrix} = \begin{pmatrix} k_{12} & -k_{12} & 0 & 0 & 0 & 0 & 0 & 0 \\ -k_{12} & k_{12} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & k_{34} & -k_{34} & 0 & 0 & 0 & 0 \\ 0 & 0 & -k_{34} & k_{34} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & k_{14} & 0 & 0 & -k_{14} \\ 0 & 0 & 0 & 0 & 0 & k_{23} & -k_{23} & 0 \\ 0 & 0 & 0 & 0 & 0 & -k_{23} & k_{23} & 0 \\ 0 & 0 & 0 & 0 & -k_{14} & 0 & 0 & k_{14} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} \tag{5.18}$$

この k による行列が物体全体の弾性力を表す行列 K に対応する．この行列を見てもわかる通り，x 方向については，質点 n, m をつないでいるバネのバネ定数が行列の n 行目の n, m 列目の要素に $k_{nm}, -k_{nm}$ として対応する．y 方向についても，y の要素についての部分に注目してみれば同様である．なお図 5.1 のように，1 つの質点に同じ方向のバネが複数接続していた場合，行列の同じ要素の部分に複数のバネ定数が入るが，この場合は単純に値を足せばよい．立方体や木の葉のような 3 次元の物体について扱う場合も，z 方向について x, y 方向要素と同様に線形システムに変数として加えるだけである．

このように同じ材質であっても形状 (質点の配置とバネの接続方法) が変わると行列 K が変わり，固有振動数も変化する．そのため，葉の衝突音を用いるためには材質の設定だけでなく，その形状に合わせたバネ-質点系によるモデリングが必要となる．実際のユーザの入力からこの弾性行列 K を作成する手順は 5.1.3 項で後述する．

(3) では，ここまでの作業で作成された K 行列の固有値を求める．本論文では固有値の算出にはヤコビ法を用いている．

この力の釣り合いの式から弾性行列 K を作成し固有値を計算する流れは 3 次元物体でも共通であり，5.1.1 項で説明した通り，計算した固有値から式 (5.6) を用いて，固有の共振周波数を求める．

5.1.3 ユーザ入力からの弾性行列の作成

3.2 節で述べたように葉の外形形状はユーザにより，20×30 ピクセル内にドラッグ操作によって入力される．この入力から 5.1.2 項で述べた物体全体の弾性行列を図 5.3 と以下に示す流れで作成する．

(1) 予めすべてのピクセルのフラグを 0 で初期化しておく

(2) ユーザがドラッグ操作したときに通ったピクセルのフラグを 1 にする

(3) 葉形状内部のピクセルを 1 で埋めるために，領域内のピクセルを x, y 方向に走査してフラグが 1 のピクセル間にあるピクセルのフラグも 1 にする．

(4) フラグ 1 のピクセルからなる葉形状ピクセルを高さ 1 ピクセルのボクセル形状として葉形状ボクセルに変換する

(5) ボクセルの頂点と辺に順に質点番号とバネ番号を振ることで葉形状のバネ質点系を作成する

(6) ユーザの決定したバネ定数の値と質点接続情報から 5.1.2 項の手順で葉の弾性行列を作成する

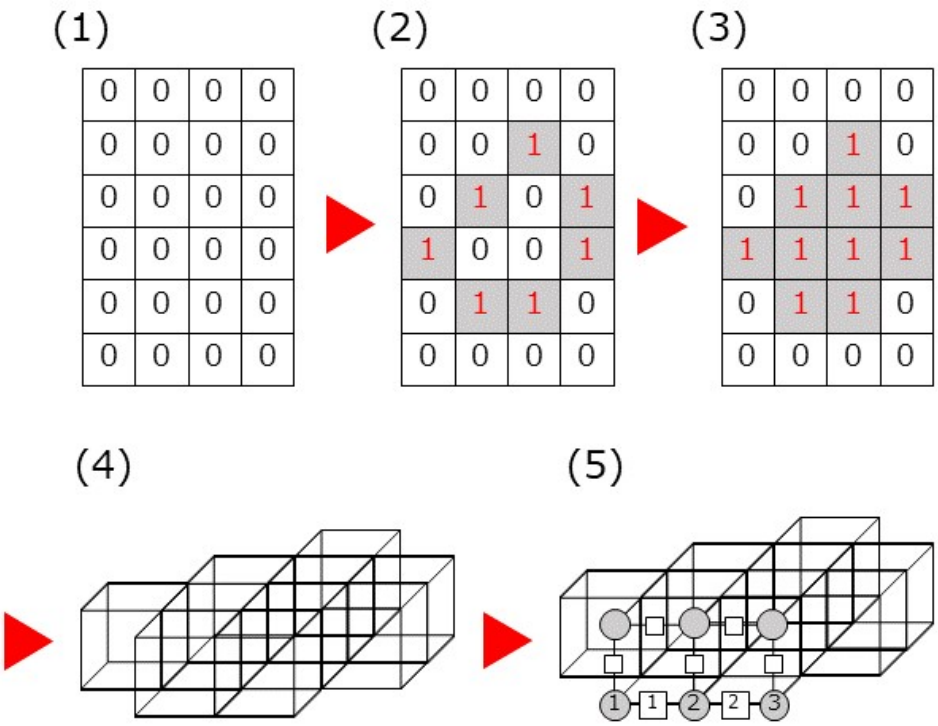

図 5.3: 弾性行列作成の流れ

なお，ここで行列の各項に値として格納するバネ定数 k は，[3] より以下の式で定義できる．

$$k = Yt \tag{5.19}$$

この式において Y はヤング率, t はバネの自然長である．バネの自然長はボクセルを構成する立方体の 1 辺の長さであるので一般的な大きさの木の葉が 20×30 ピクセルの範囲に収まるように，入力範囲の最大値を縦 $150[mm]$ 横 $100[mm]$ とし，$t = 5.0[mm]$ と設定した．

木の葉のヤング率は木の葉の種類に依存することもあり，具体的な値は定められていないため，金属や木材のようなヤング率がある程度定まっているものをサンプルとして，その強度の比較から決定する．具体的には物体の曲がりやすさであるヤング率と引っ張った時の強さである引張強度を材質ごとにまとめたものが表 5.1 であり，これらの値に相関がみられることから木の葉の引張強度からヤング率を定める．

表 5.1: ヤング率と引張強度の例

	引張強度 (MPa)	ヤング率 (GPa)
木材	20	13
ガラス	50	71.3
純チタン	320	106
一般構造用延鋼材	450	206

木の葉の引張強度は，Onoda ら [20] によると広葉樹の場合 4.0 から 8.0(MPa) であり，先程の相関から考えるとヤング率は約 3.0 から 5.0(GPa) であるといえる．この範囲内の値を想定した木の葉の硬さによって設定する．

5.1.4 衝突音の再生方法

このようにして生成した衝突音を衝突時に鳴らすのだが，葉を表す球同士がめり込んでいる間，常に衝突していると判定され，毎フレーム衝突と判定された葉の枚数回衝突音を鳴らしてしまうと，音が多くなりすぎてしまい不自然な上に処理速度も遅くなる．そのため前フレームの接触状態を葉毎に記録しておき，図 5.4 に示すように衝突判定の状態によって非接触状態から接触状態を遷移させる．木の葉の衝突音が鳴るのは衝突した瞬間のみであることを踏まえて，この状態が非接触状態から接触状態に変化した時のみ衝突音を鳴らすという処理を行う．

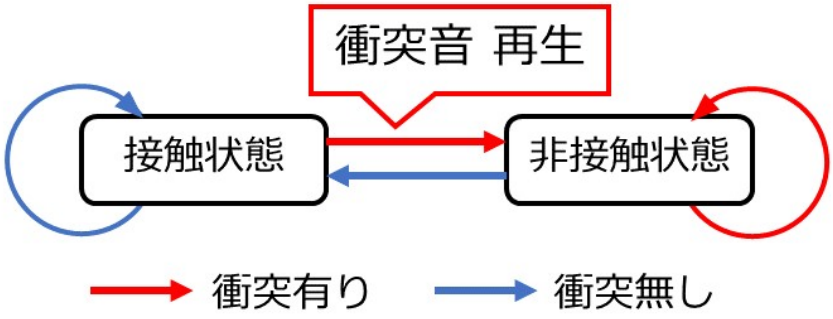

図 5.4: 接触状態の遷移図

また，衝突音は音源が各葉であり，[5] で用いられている以下の式を利用して空間による音の減衰を考慮すると，受聴点での音の大きさ $A(d)$ は音源点 d_0 での音の大きさ A_0 を使用して，次式で表される．

$$A(d) = A_0(d_0/d)^2 S(\mu) \tag{5.20}$$

ここで，d は音源と受聴点との距離である．また，$S(\mu)$ は受聴点の仮想マイクロホンの単一指向性を表すカーディオイド関数であり，次式で示される．

$$S(\mu) = C + ((1 + \cos(\mu))/2)^k \tag{5.21}$$

本論文では係数として [5] と同様に $k = 1, C = 0$ 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．

5.2 摩擦音生成

自然界の摩擦音は，フラクタル図形などに見られる自己相似性と呼ばれる全体と部分が相似である性質を持っており，このような性質を持つ音は 1/f ノイズと呼ばれる．この 1/f ノイズのパワースペクトル図を見ると，図 5.5 に示すようにパワースペクトル密度が周波数に反比例しており，フラクタル状になっていることがわかる．こうした 1/f ゆらぎ成分を持つものは摩擦音だけではなく，波の音や風の吹き方，太陽光や炎の燃え方など自然界の様々な箇所に見られ，人間にとって心地よいと感じるゆらぎであることが知られている．本手法では音再生のために使用した PortAudio[21] の機能を用いて Gardner 法によって 1/f ノイズを生成し，それを元に摩擦音を再生する．

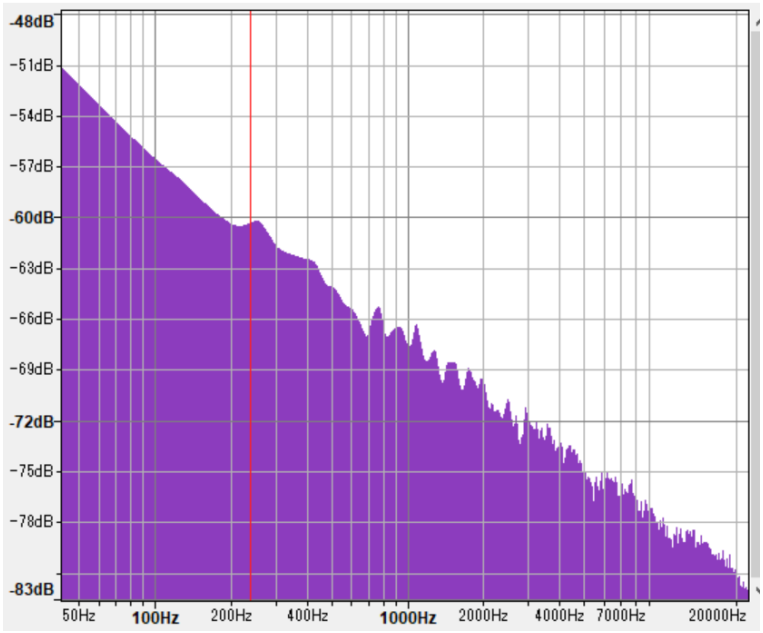

図 5.5: 1/f ノイズのパワースペクトル図

5.2.1 摩擦音の再生方法

摩擦音に関しては衝突音と異なり，接触状態の木の葉が存在し，風による木の葉の動きがある場合は鳴り続けると考えられるため，木全体が風により揺れ動くシーンでは木全体を音源として捉え，衝突の際にその都度鳴らすという方法ではなく，常に鳴らし続けたうえで音量を変化させることでシーンと連動させる．

具体的には葉の総枚数 L_{size} のうち接触している木の葉の枚数 $L_{contact}$ と，風の強さ U という 2 つの値に比例して音量 $P_{friction}$ を以下の式のように調節する．

$$P_{friction} = \frac{L_{contact}}{L_{size}} U \tag{5.22}$$

なお，外力が風ではなく，手によって揺れ動かされるシーンでは，音源は木全体ではなく一部の葉の接触であるため，衝突音と同様に接触に応じて再生するという方法をとる．

5.3 風切り音生成

風切り音とは風が枝の間をすり抜けていく際に発生する音であり，これは風が吹いたときに物体の後方にできるカルマン渦の周期的な渦放出が原因で発生する．この音の周波数は渦放出周波数と同じであり，流速 U の一様な流れの中に直径 D の円柱が存在する場合のカルマン渦列の周波数 f_0 は以下の式で求められる．

$$f_0 = StU/D \tag{5.23}$$

ここでの St はストローハル数である．

また，f_0 の振幅 I_0 は，枝と観測点の距離を r とすると以下の式のようになる．

$$I_0 \propto U^6/r^2 \tag{5.24}$$

式 (5.23) のストローハル数 St は，流体力学において慣性力と粘性力との比で定義される無次元量であるレイノルズ数 Re の関数であり，$4 \times 10^2 < Re < 2 \times 10^5$ での範囲内あれば St はほぼ一定値で 0.2 であるとされているため，本手法ではストローハル数 St を 0.2 の定数として扱う．

5.2 章で説明したように自然界の音のいくつかは 1/f ゆらぎの特性を持ち，風切り音もそのうちの 1 つである．そのため式 (5.23) を用いて計算した基底周波数を元に 5.2 項の手法で生成した 1/f ノイズに reson filter をかけて特徴を持たせることで，風切り音を生成する．

5.3.1 reson filter

reson filter[4] は指定した周波数帯域を強調するフラクタルフィルターで，次の式で表される．

$$y_t = x_t - Rx_{t-2} + 2Ry_{t-1}\cos\theta - R^2 y_{t-2} \tag{5.25}$$

ここでの x_t, y_tがそれぞれ時刻 t における 入力と出力であり，R とθ は強調する帯域幅によって決定される変数である．R とθ は以下に示す式で求めることができる．

$$R = 1 - \frac{bw}{Fs}\pi \tag{5.26}$$

$$\theta = \cos^{-1}(\frac{1+R^2}{2R})\cos(2\pi\frac{Fr}{Fs}) \tag{5.27}$$

この式において Fr は強調させたい周波数帯域の中央値, bw は強調させたい幅の半幅 であり，Fs はサンプリングレートである．図 5.6 はサンプリングレートは $44100Hz$ のホワイトノイズに, $Fs = 44100, Fr = 1000, bw = 2000$ と設定し reson filter を掛けたパワースペクトル図であり，周波数 $1000Hz$ の箇所が強調され，それ以上の周波数帯域でフラクタル形状になっていることが確認できる．

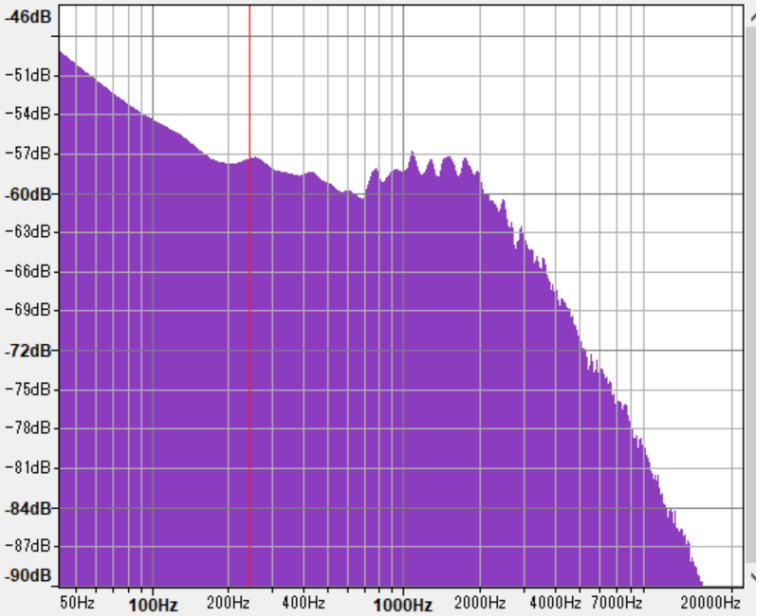

図 5.6: reson filter を適用した 1/f ノイズ

5.3.2 風切り音の再生方法

風切り音の計算を毎ステップ行うことは計算量が多く困難であるが，本手法では構築された木構造の枝の本数や直径はユーザによるパラメータ変更がない限りシーンの途中では変化せず，風速のみがステップによって異なる．そのため，あらかじめ幹と枝の形状から風速 U に関する風切り音の波形表 $WindSoundTable(U)$ を作成し，シーンの風速に応じて音量を変化させながら再生し，パラメータ変更があればその都度波形表を更新する方法をとる．なお，本来各枝での流速は枝に垂直な風の成分であり，風に対する枝の角度から計算するが，樹木を音源とする場合，風は地面に対して平行に吹き，木は地面に対してほぼ垂直に立っていると考えると，大局的な風切り音ではこの角度の情報を無視できると考え，流速を風速で代用する．

具体的には，波形表 $WindSoundTable(U)$ は以下のようにして作成する．

(1) 式 (5.23) を用いて風速 Ui の時の直径 Di の枝からなる音の基底周波数を計算する．

(2) 用意した 1/f ノイズに (1) で求めた基底周波数の周波数帯が強調されるように reson filter をかける．

(3) いくつかの D_i について (1)(2) を行い生成された波形を足し合わせ，流速 U_i の時の波形を作成し，波形表に格納する．

(4) すべての U_i について上記 (1)〜(3) を行う．

シミュレーション内では，このようにして作成した波形表 $WindSoundTable(U)$ からシーンの風速に応じて適切な波形を選択し，式 (5.24) を用いて計算した振幅を基に音量を変化させながら再生することで風切り音が再生できる．

5.4 音の出力

ここまでの手法を用いて，衝突音，摩擦音，風切り音それぞれの音の波の式もしくは波形表を求められるが，これらをPCから音として出力する必要がある．音の出力にはPortAudio[21]というC++向けのライブラリを使用した．PortAudioではストリームを開始すると，1/SamplingRate秒ごとにコールバック関数が呼ばれる (SamplingRateはユーザーが設定したサンプルレート)．このコールバック関数では生成した音を格納しておくためにあらかじめ用意したバッファ(以降，生成音バッファと呼ぶ) に格納されている音圧データを別スレッドで音として出力する．

衝突音に関しては，衝突判定を行い木の葉同士の接触が発生したタイミングで衝突音を生成音バッファに格納する関数を呼び出すことで，生成音バッファが書き換えられ音が鳴る．摩擦音の場合，外力を風としているシーンでは5.2.1項でも述べたように，木全体を音源として捉え，常に鳴らし続けたうえで音量を変化させている．つまり，用意した1/fノイズに適切な音量調節関数をかけながら生成音バッファに毎フレーム書き込み続ける．これは風切り音も同様であり，生成した波形表から適切なものを選択し，音量を調節しながら生成音バッファに書き込み続ける．なお，風ではなくユーザが外力を与えて揺れ動かすシーンでは，摩擦音も衝突音と同様のタイミングで生成音バッファに書き込むことで音を鳴らしている．

第6章　結果

本論文の提案手法を CPU で実装し実験を行った結果を示す．表 6.1 に実験開発環境を，表 6.2 に全シーンで共通するパラメータを示す．なお，各シーンにおいて赤色の葉は接触状態，緑色の葉は非接触状態を表す．

表 6.1: 実行環境

OS	Microsoft Windows 10 64bit
CPU	Core i7-8700 3.2GHz
メインメモリ	16GB
開発言語	C++
グラフィックス API	OpenGL
サウンド API	PortAudio

表 6.2: 全シーンに共通するパラメータ

タイムステップ幅 [s]	0.01
重力加速度 $[m/s^2]$	9.8
位置ベース法の反復回数	5
Sampling Rate [Hz]	44100
生成音バッファサイズ	44100
木の葉のバネ定数	1.0×10^7
減衰係数 α_1	1.0×10^{-8}
減衰係数 α_2	1000
衝突の衝撃 Δtg	500.0

6.1　入力パラメータの反映

本論文で扱う樹木モデルや音は 3.2 節で述べたようにユーザの入力によって異なる形状や音にインタラクティブに変化する．本節では木構造，葉の大きさ，衝突音についてユーザが異なる値を入力した際の結果を示す．

6.1.1　木構造

低木と高木それぞれにおいて異なる樹木の形状パラメータ $tree\_h$，$tree\_w$，$leaf\ num$ を入力した結果を示す．各パラメータを表 6.3 に，生成された木構造を図 6.1，図 6.2，図

6.3，図 6.4 に示す．

表 6.3: 木構造のパラメータ

	図 6.1	図 6.2	図 6.3	図 6.4
樹木形状	低木	低木	高木	高木
樹木の高さ $tree\_h$	3	5	3	5
樹木の幅 $tree\_w$	4	5	4	5
エッジあたりの葉の枚数 $leaf\ num$	3	4	3	4
葉の大きさ $leaf\_rad$	0.05	0.05	0.05	0.05

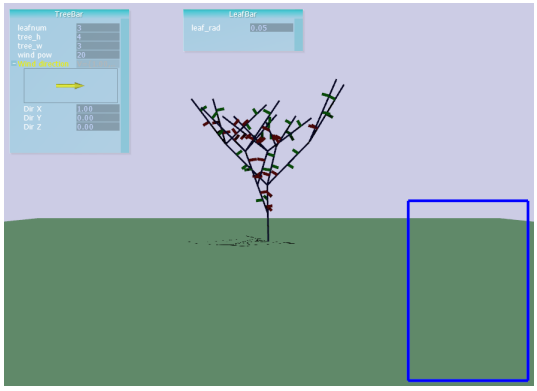

図 6.1: 樹木の形状 低木 1

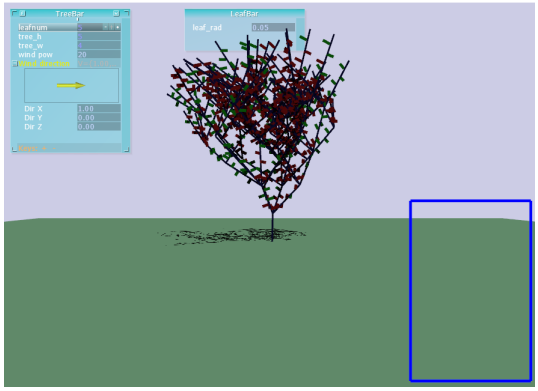

図 6.2: 樹木の形状 低木 2

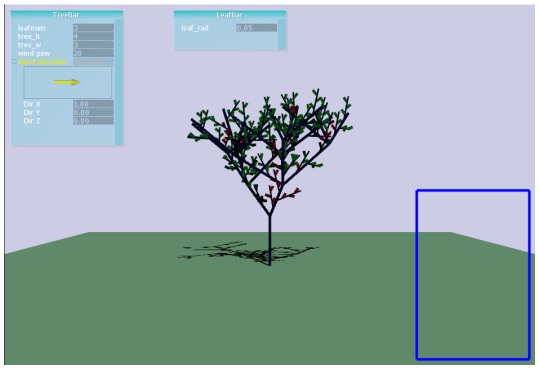

図 6.3: 樹木の形状 高木 1

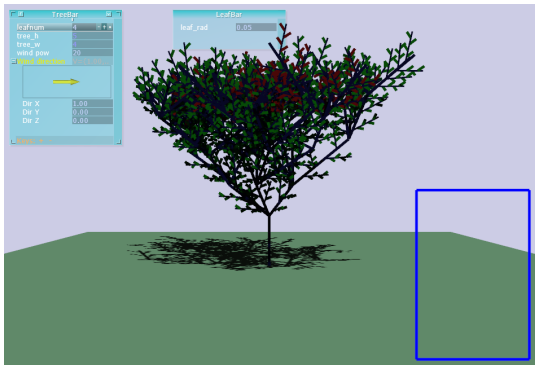

図 6.4: 樹木の形状 高木 2

6.1.2 葉の大きさ

図 6.4 から葉の大きさ $leaf\_rad$ のみ 0.15 に変更した結果を図 6.5 に示す．前述のとおり赤で描画された葉は，葉同士で衝突していることを表す．

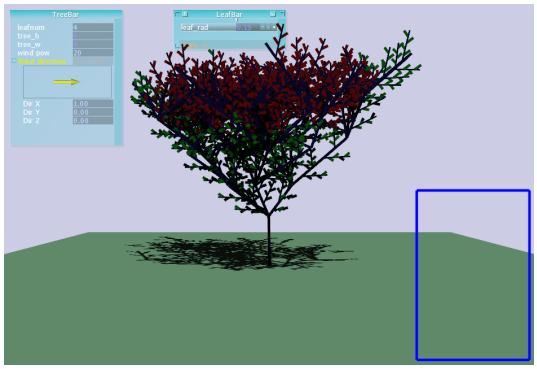

図 6.5: 樹木モデル：高木 3

6.1.3 衝突音

異なる葉形状を入力したときの衝突音生成結果を示す．入力した葉の形状を図 6.6，図 6.7 に，それぞれの波形とスペクトログラムを図 6.10，6.11 に，パワースペクトル図を図 6.8，6.9 に示す．ここでの図 6.6 の形状は笹のような細長い葉を，図 6.7 の形状は広葉樹のような幅の広い葉を想定している．このシーンにおける衝突音は一対の葉を任意のタイミングで複数回衝突させている．

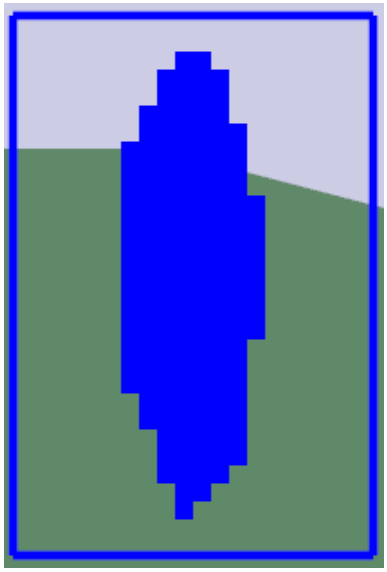

図 6.6: 入力葉形状 細長い葉

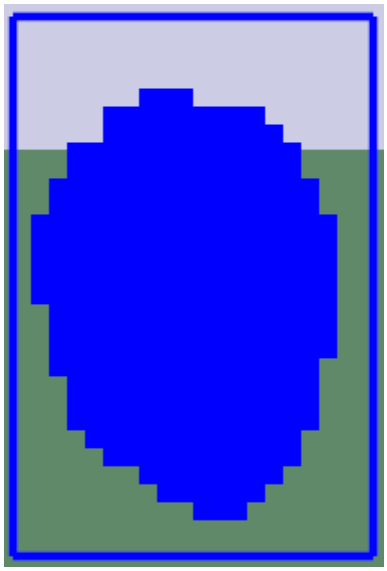

図 6.7: 入力葉形状 幅の広い葉

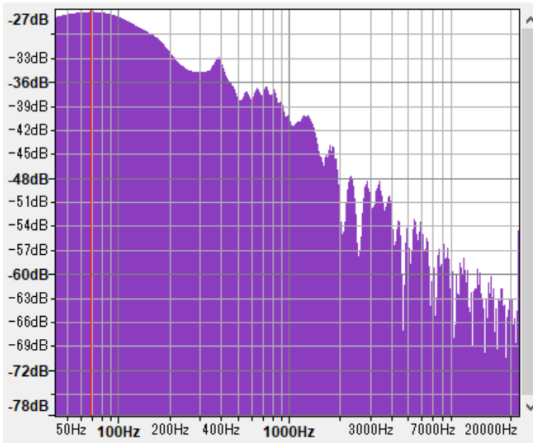

図 6.8: 細長い葉の衝突音のパワースペクトル図

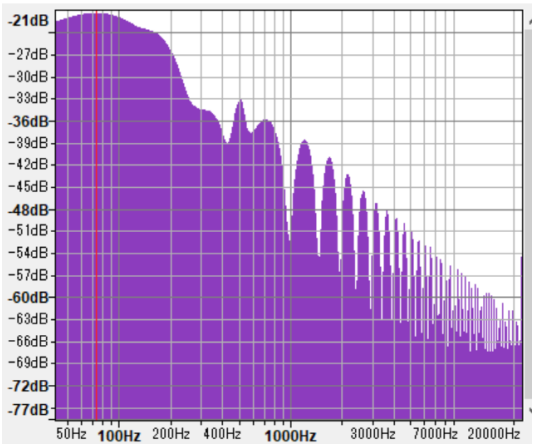

図 6.9: 幅の広い葉の衝突音のパワースペクトル図

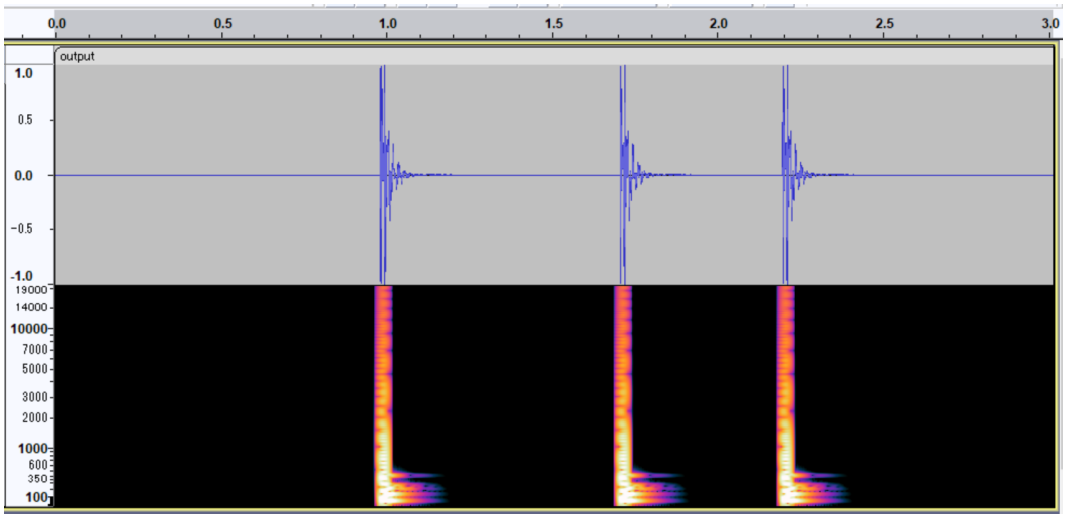

図 6.10: 細長い葉の衝突音

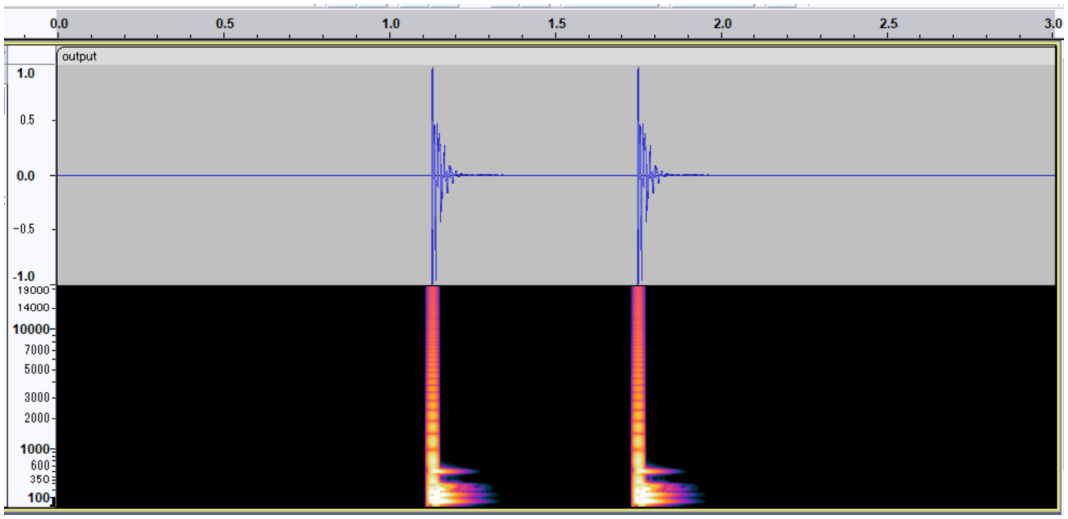

図 6.11: 幅の広い葉の衝突音

6.2　音生成

本節ではシミュレーションにおいて，衝突音，摩擦音，風切り音のみをそれぞれ出力した際の結果を示す．なお，これらのシーンの樹木パラメータはすべて図 6.3 と同様のものであり，外力となる風については周期 200 タイムステップ（2[s]），振幅が風の強さパラメータ $wind\ pow$ である sin 波を用いることで周期的に変化させている．音生成シーンにおけるパラメータを表 6.4 に示す．

衝突音のみを出力した際の波形表とスペクトログラムを図 6.12 に，摩擦音のみを出力した際の波形表とスペクトログラムを図 6.13 に，パワースペクトル図を図 6.14 に，風切り音のみを出力した際の波形表とスペクトログラムを図 6.15 に，パワースペクトル図を図 6.16 に示す．

表 6.4: 音生成シーンのパラメータ

樹木形状	高木
樹木の高さ $tree\_h$	3
樹木の幅 $tree\_w$	4
エッジあたりの葉の枚数 $leaf\ num$	3
葉の大きさ $leaf\_rad$	0.05
基本となる風の強さ [m/s]	20
幹ノードの質量 [kg]	2.0
枝ノードの質量 [kg]	1.5
葉ノードの質量 [kg]	0.1

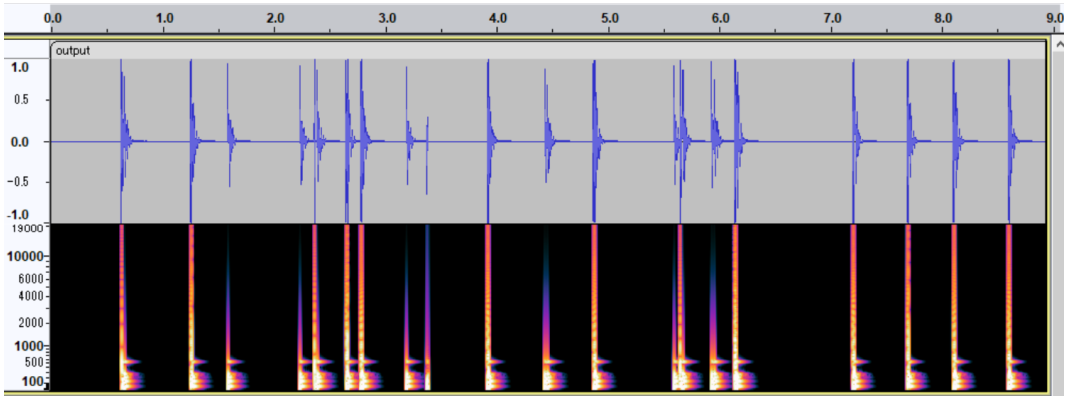

図 6.12: 衝突音のみを出力した際の波形とスペクトログラム

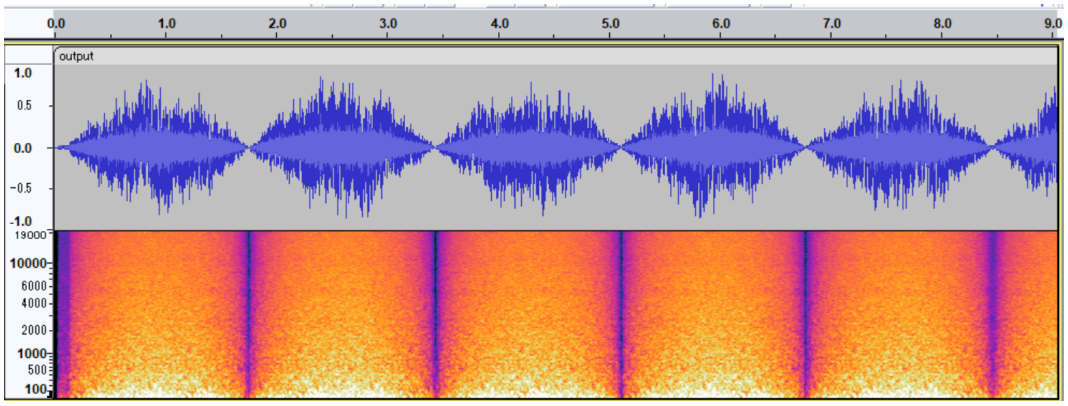

図 6.13: 摩擦音のみを出力した際の波形とスペクトログラム

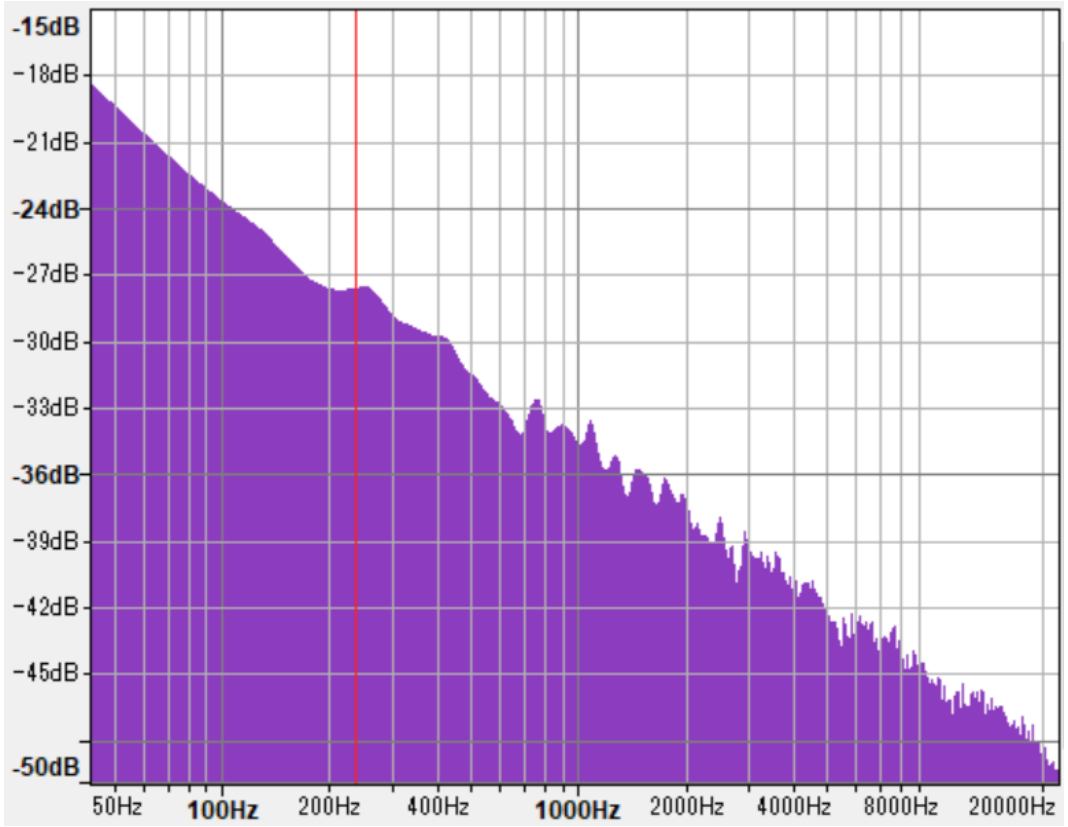

図 6.14: 摩擦音のパワースペクトル図

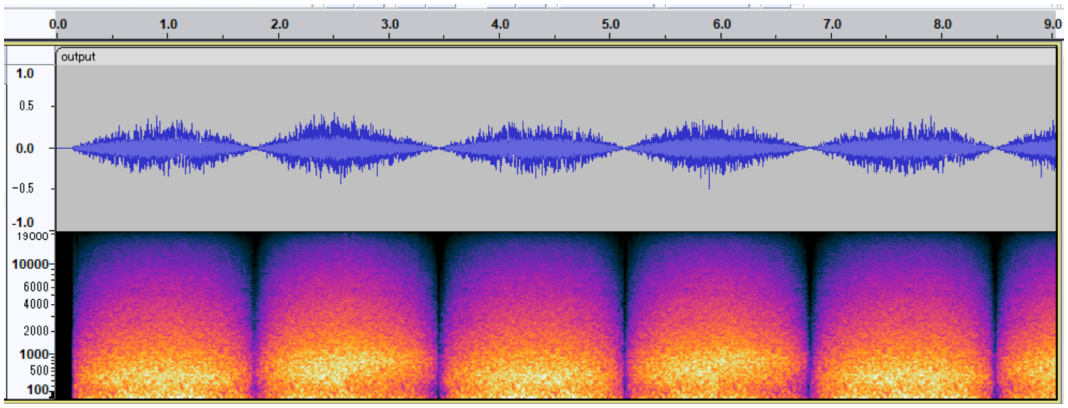

図 6.15: 風切り音のみを出力した際の波形とスペクトログラム

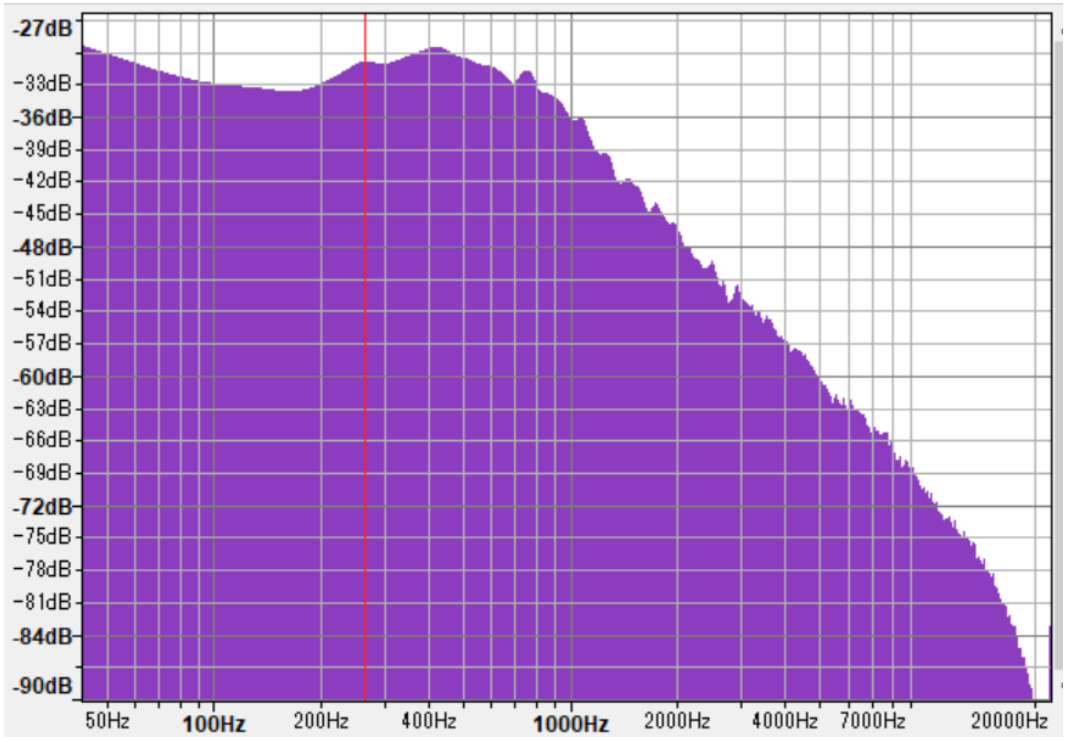

図 6.16: 風切り音のパワースペクトル図

6.3　シミュレーション結果

4 種類のシーンにおいてパラメータを変えながら，樹木の揺れ動きのシミュレーションとシーン連動した音の出力を行った結果を示す．表 6.5 に各シーンにおけるフレームレートについてまとめる．なお，シーン 4 においては音を出力せず揺れ動きのみのシミュレーションを行った際は 60fps 前後を保つため，大量の衝突による生成音バッファへの書き込みの負荷が大きいと考えられ，揺れ動きのシミュレーション自体はリアルタイムに行うことができている．

シーン 1~3 については風で揺れる木々の音，シーン 4 ではユーザが手動で揺らした時の音をシミュレーションしており，風については 6.2 節と同様の sin 波を用いて周期的に変化させており，いずれのシーンでも風の強さを 10[s] あたりから 15[s] にかけて 30[m/s] に変化させ，20[s] にかけて 10[m/s] に弱くしている．

表 6.5: 各シーンにおけるフレームレート (fps)

シーン	フレームレート (fps)	図
シーン 1	62-67	図 6.17
シーン 2	20-67	図 6.19
シーン 3	62-67	図 6.21
シーン 4	5-67	図 6.23

6.3.1　シーン 1

シーン 1 では葉の少ない高木を風で揺らした場合の揺れ動きと発生する音をシミュレーションした．シーン 1 のパラメータを表 6.6 に，シミュレーション画面を図 6.17 に，出力された音の波形とスペクトログラムを図 6.18 に示す．

表 6.6: シーン 1 のパラメータ

樹木形状	高木
樹木の高さ $tree\_h$	3
樹木の幅 $tree\_w$	4
エッジあたりの葉の枚数 $leaf\ num$	3
葉の大きさ $leaf\_rad$	0.05
基本となる風の強さ [m/s]	$10 \sim 30$
幹ノードの質量 [kg]	2.0
枝ノードの質量 [kg]	1.5
葉ノードの質量 [kg]	0.1

6.3.2　シーン 2

シーン 2 では葉の多い高木を風で揺らした場合の揺れ動きと発生する音をシミュレーションした．シーン 2 のパラメータを表 6.7 に，シミュレーション画面を図 6.19 に，出力された

音の波形とスペクトログラムを図 6.20 に示す.

表 6.7: シーン 2 のパラメータ

樹木形状	高木
樹木の高さ $tree\_h$	4
樹木の幅 $tree\_w$	5
エッジあたりの葉の枚数 $leaf\ num$	4
葉の大きさ $leaf\_rad$	0.05
基本となる風の強さ [m/s]	$10 \sim 30$
幹ノードの質量 [kg]	2.0
枝ノードの質量 [kg]	1.5
葉ノードの質量 [kg]	0.1

6.3.3 シーン 3

シーン 2 では低木を風で揺らした場合の揺れ動きと発生する音をシミュレーションした. シーン 3 のパラメータを表 6.8 に，シミュレーション画面を図 6.21 に，出力された音の波形とスペクトログラムを図 6.22 に示す.

表 6.8: シーン 3 のパラメータ

樹木形状	低木
樹木の高さ $tree\_h$	5
樹木の幅 $tree\_w$	5
エッジあたりの葉の枚数 $leaf\ num$	4
葉の大きさ $leaf\_rad$	0.05
基本となる風の強さ [m/s]	$10 \sim 30$
幹ノードの質量 [kg]	1.5
葉ノードの質量 [kg]	0.1

6.3.4 シーン 4

シーン 4 は外力を風でなく，人の手による力としマウスで枝をピックし直接動かした場合の揺れ動きと発生する音をシミュレーションした．シーン 4 のパラメータを表 6.9 に，シミュレーション画面を図 6.23 に，出力された音の波形とスペクトログラムを図 6.24 に示す.

表 6.9: シーン 4 のパラメータ

樹木形状	高木
樹木の高さ $tree\_h$	3
樹木の幅 $tree\_w$	4
エッジあたりの葉の枚数 $leaf\ num$	3
葉の大きさ $leaf\_rad$	0.05
基本となる風の強さ [m/s]	0
幹ノードの質量 [kg]	2.0
枝ノードの質量 [kg]	1.5
葉ノードの質量 [kg]	0.1

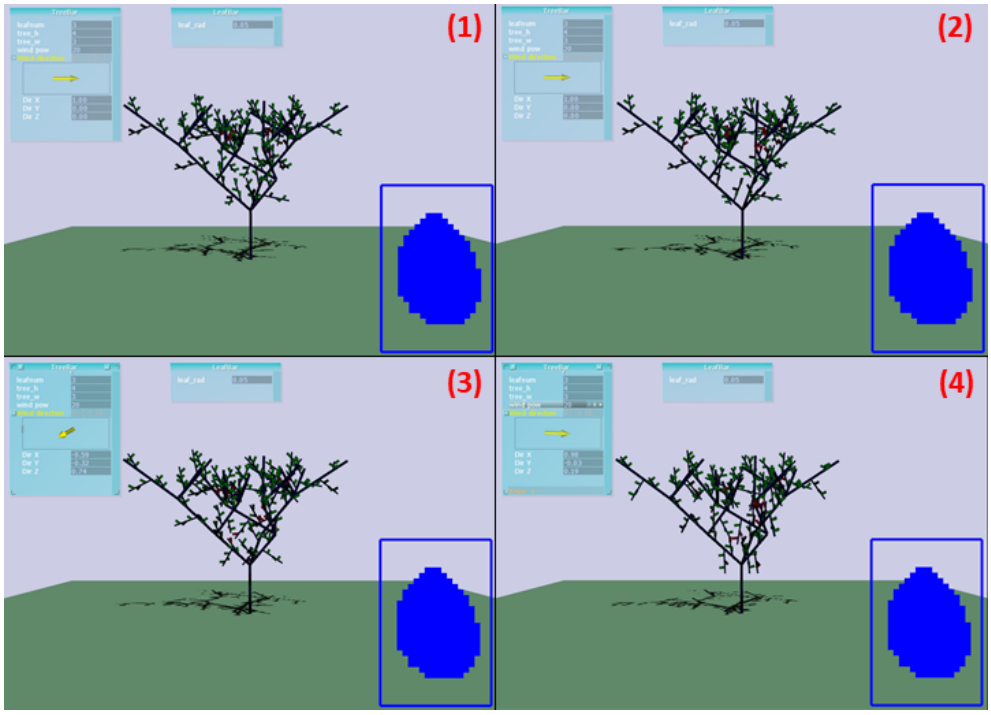

図 6.17: シーン 1

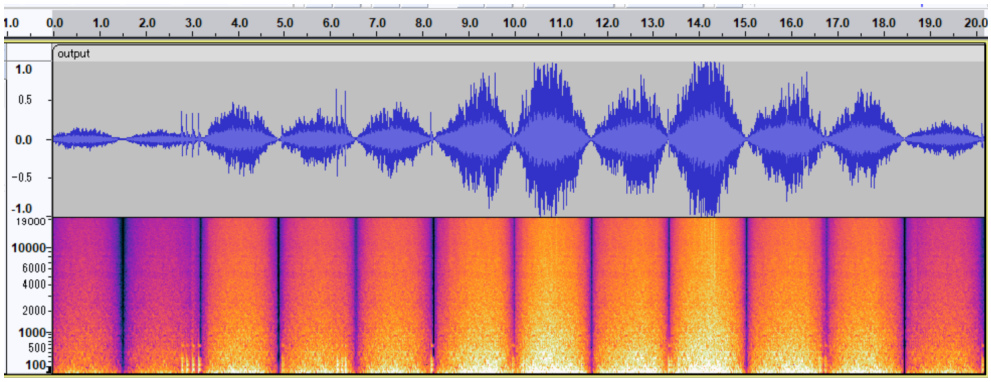

図 6.18: シーン 1 の波形とスペクトログラム

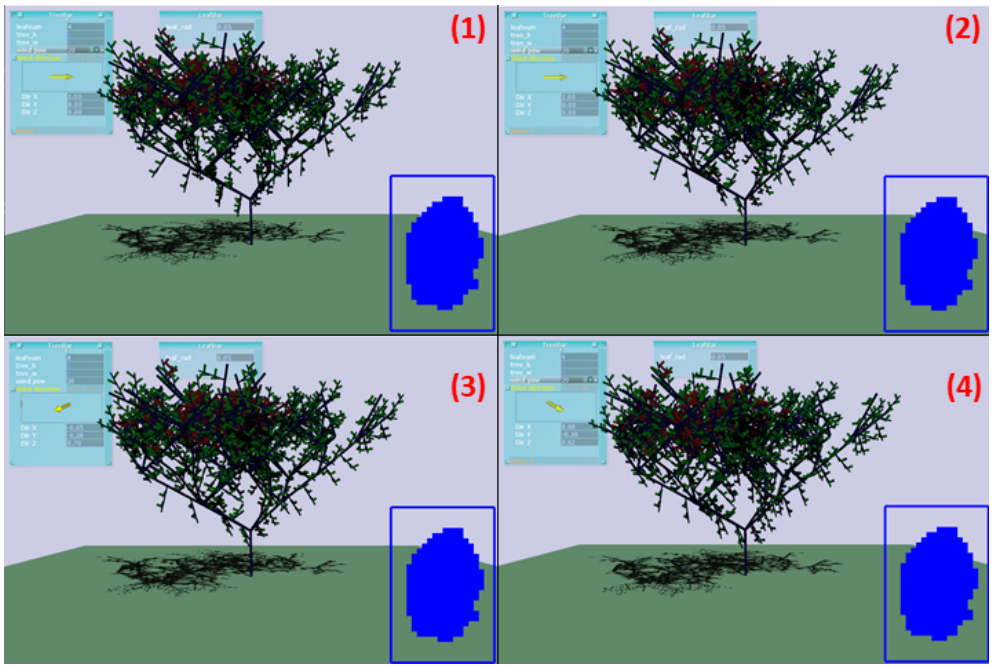

図 6.19: シーン 2

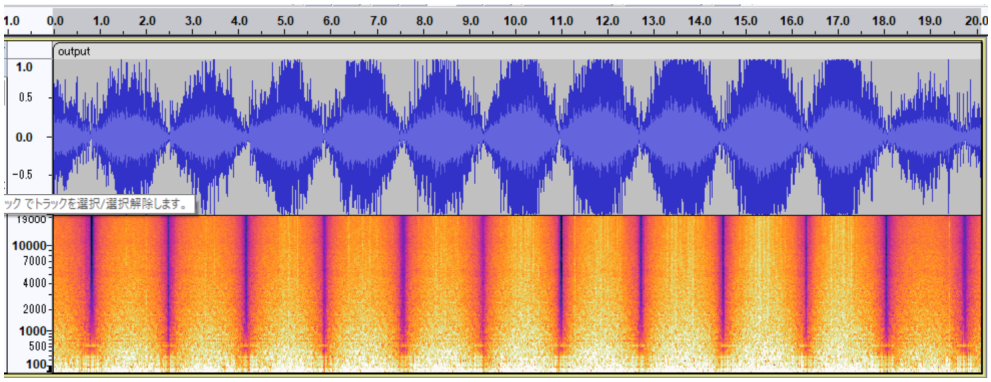

図 6.20: シーン 2 の波形とスペクトログラム

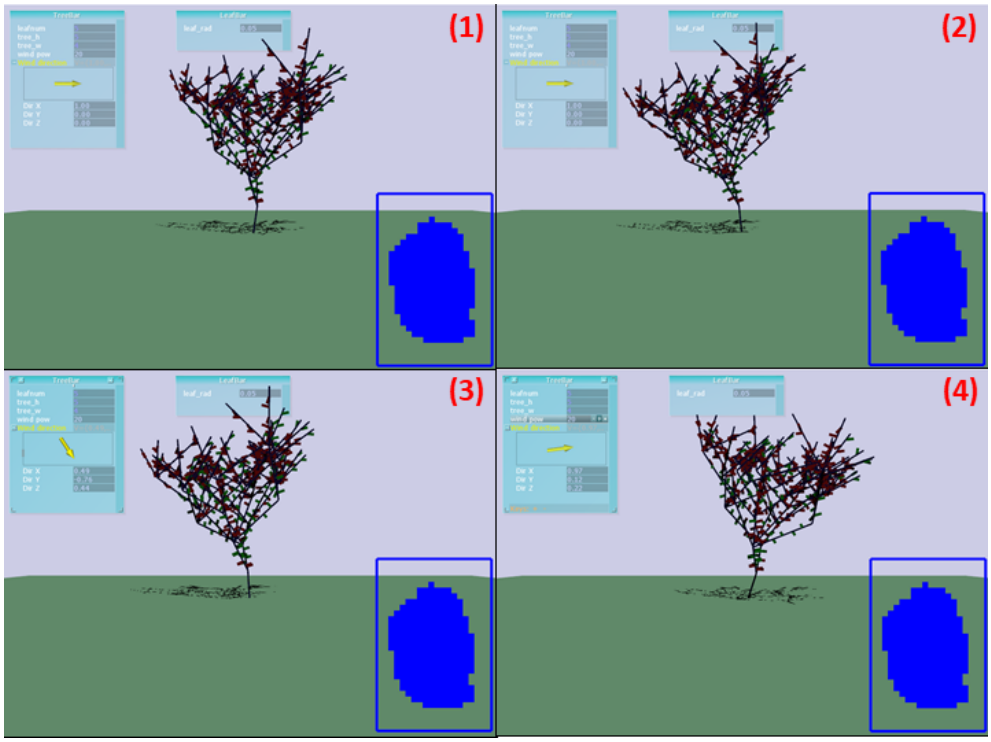

図 6.21: シーン 3

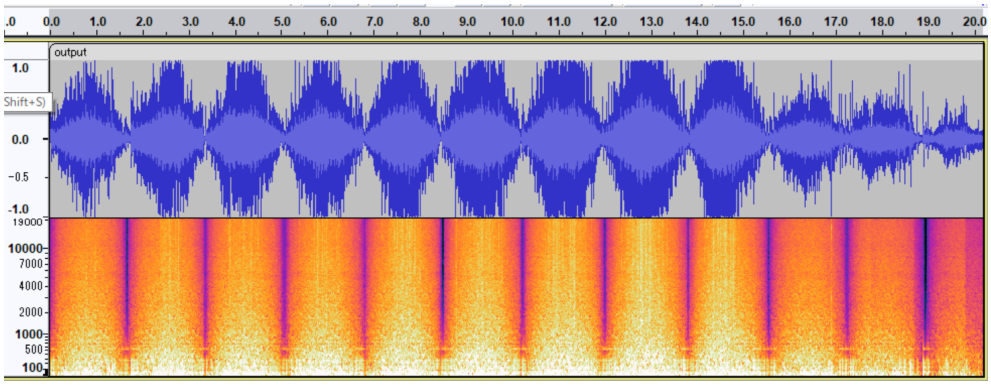

図 6.22: シーン 3 の波形とスペクトログラム

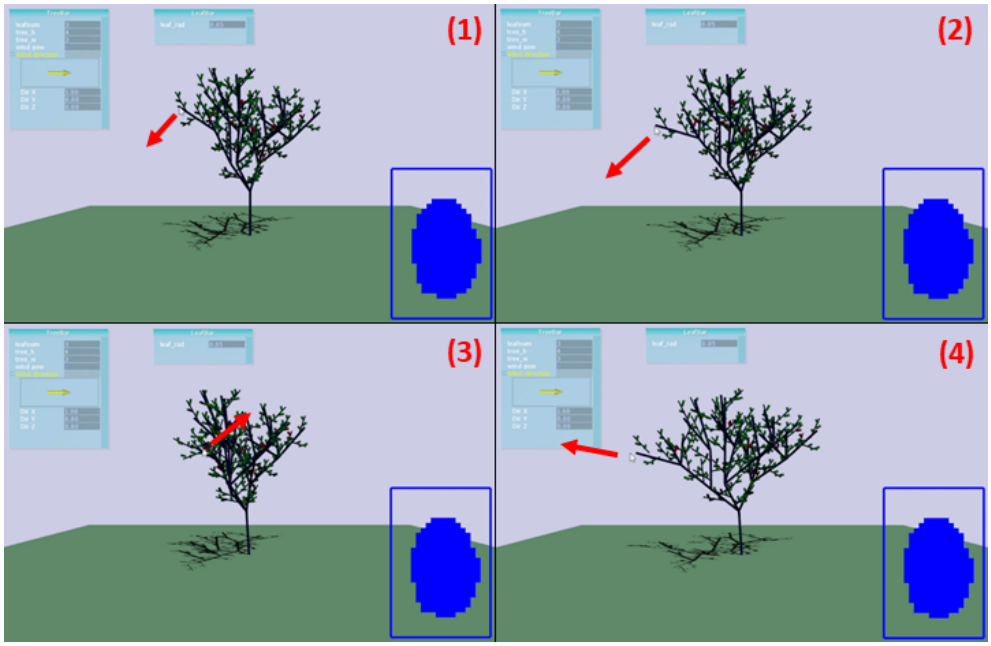

図 6.23: シーン 4

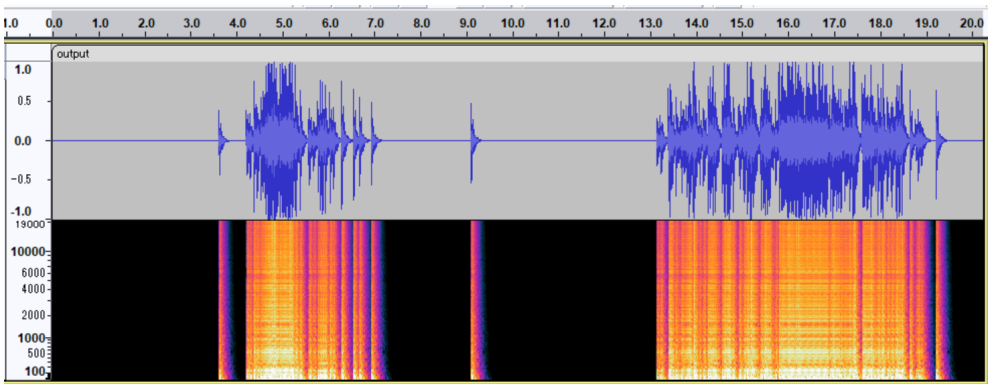

図 6.24: シーン 4 の波形とスペクトログラム

第7章　考察

7.1　ユーザの入力による形状と音の変化

3.2 節で述べたように，本論文においてユーザの指定できる樹木の形状に関わるパラメータは樹木の高さ $tree\_h$，幅 $tree\_w$，葉の枚数 $leaf\ num$ であり，図 6.1，図 6.2，図 6.3，図 6.4 に示すように，低木と高木共に，これらの値を変更するとパラメータに応じた樹木の形状が構築されている．

次に葉に関わるパラメータについては，葉の大きさ $leaf\_rad$ と形状があり，葉の大きさに関しては 4.4 節で述べたように衝突判定の影響範囲を決定する値である．図 6.4 と図 6.5 はこの $leaf\_rad$ 以外が共通のパラメータであり，図 6.5 の $leaf\_rad$ の方が大きい値に設定されている．これらを比較すると赤色で示した接触状態の葉が増えており，ユーザの入力した葉の大きさが正しく衝突判定に反映されていることがわかる．形状に関しては，5.1.3 項で述べたように入力された葉の形状から弾性行列を作成し，モード解析を行うことで衝突音を生成しているため，図 6.10，図 6.11 の波形とスペクトログラムや，図 6.8，図 6.9 のパワースペクトル図に示すように形状によって異なる音が生成されている．

このようにユーザの入力によって形状や音を変化させることを可能にしたが，ユーザが詳細に樹木の情報を設定したいと考えた場合現在設定できるパラメータだけでは不十分である．例えば樹木全体の概形においては一般的に広葉樹は横に大きく広がり，針葉樹は横にはあまり広がらず縦に長く伸びていく傾向があるが，本手法では中心となる幹からランダムな幅で枝分かれを繰り返すため，どちらかというと広葉樹のような形状しか構築できない．他にも葉の材質は樹種や時期に用って水分量などが異なり，当然発生する音も変化するが本手法では指定することはできない．ただ，このように指定できるパラメータを増やすとユーザの自由度は高くなるが設定の手間が増加する．そのため，例えばいくつか用意した選択肢の中から任意の樹種を選択すると，すべてのパラメータが自動で設定されるようなユーザインターフェースを実現することも必要である．

7.2　音生成

衝突音に関しては，パワースペクトル図（図 6.8，図 6.9）に注目するといくつもの特徴のある周波数帯域が表れており，これはモード解析により正しく固有周波数が計算され，音に反映できていることを示している．加えて，どちらも材質を決定するバネ定数が共通のため顕著な差ではないが，細長い形状の木の葉（図 6.6）の方が幅の広い木の葉（6.7）より高周波の成分を多く含んでおり，実際の剛体の衝突音の特性も正しく再現できていると言える．

しかし，本論文では木の葉を変形しない剛体として扱っているが，実際の木の葉は変形する弾性体であるため，より正確に音を生成しようと考えた場合変形を考慮し，シーン毎にその形状に応じてモード解析を行い音を変化させる必要がある．この計算にはモード解析に加え木の葉の変形も計算する必要があり，かなりの計算コストがかかるためリアルタイムでの実現は難しい．そのためより高速かつ正確な音生成手法を考える必要がある．

摩擦音に関しては実際の自然界の摩擦音に見られる自己相似性を持つ 1/f ノイズを用いることで生成した．図 6.14 に示すパワースペクトル図からも 1/f ノイズが生成できていることが確認でき，図 6.13 から樹木の揺れに合わせて音量が変化していることがわかる．

しかし，本提案手法はあくまでノイズベースの手法であり，1 つ 1 つの摩擦音自体をシミュレーションにより生成しているわけではない．木々のざわめきのような大局的な音を再現する際には問題なく聞こえるが，シーン 4 のような木を揺らした時のガサガサという音に関しては必ずしも正確な音であるとは言えない．このようなシーンをより正確に再現するためには中塚ら [10] のような物体表面の変形や摩擦を考慮した物理シミュレーションに基づく手法で実装することが必要になる．

風切り音に関しても同様であり，図 6.13，図 6.14 に示すように，1/f ノイズに基底周波数を考慮して reson filter を掛けることで周波数特性を再現した音を生成できているが，これも正確な物理シミュレーションに基づいている訳ではない．風切り音に関しては Dobashi ら [11] のように数値流体力学を用いることが考えられるが，この手法では物体の各部分における空気の流れの事前計算を大量に行う必要があるため，そのまま本手法のようなユーザ入力により木の形状などをリアルタイムに変更されるシステムには適用できない．そのため，この手法を本論文でも用いている [5] らの流速に応じた波形表をあらかじめ作成する手法と組み合わせるなどして，正確かつ処理が高速な風切り音生成を考える必要がある．

シーンに連動した音生成の観点からは，シーン 1, 2, 3 のそれぞれにおいてシーンにおける接触の状態による衝突音と風による周期的な音の変化を反映した摩擦音，風切り音が生成出来ていることがわかる．それぞれのスペクトログラム見ても低周波成分が多く，高周波にかけて減少しており，図 7.1 に示す実際のざわめきの波形とスペクトログラム音と比較しても近い周波数特性を持つ音が生成出来ていると言える．しかし，実際のざわめき音のパワースペクトル図（図 7.2，図 7.3，図 7.4）は大きく見て 1/f ノイズであることは共通だが，樹種によってやや異なる周波数特性を持つ．これは小松 [22] らの樹木葉擦れ音の物理特性を検証した論文からも葉の形状による音の違いだと考えることができるが，本手法では再現できていない．これを実現する方法としては，本手法で用いている 1/f ノイズによる摩擦音に風切り音に使用している reson filter を用いることで異なる箇所の周波数を強調し周波数特性を再現できると考えられる．

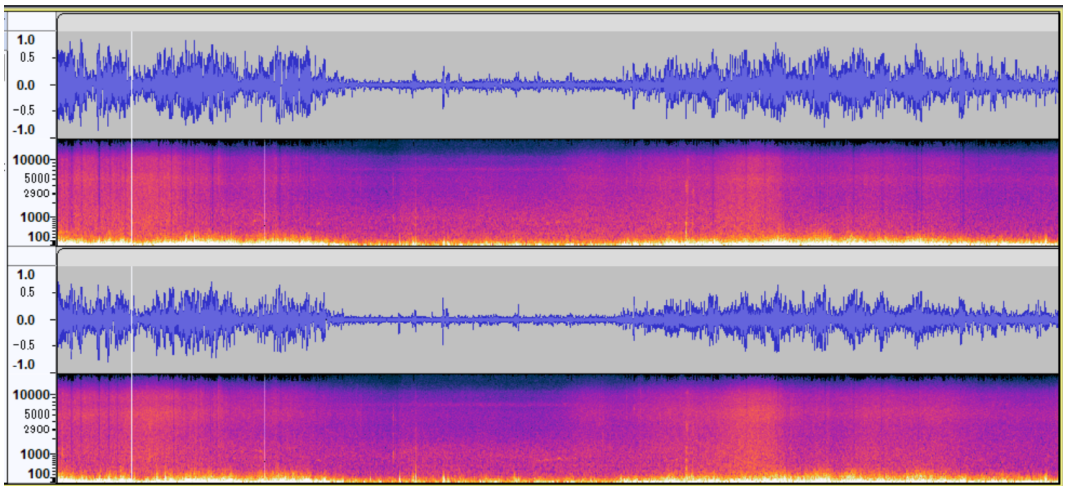

図 7.1: 録音した木々のざわめきの波形とスペクトログラム

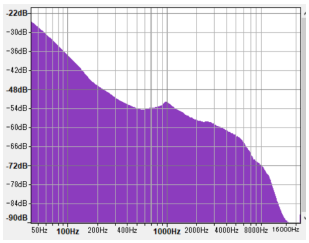

図 7.2: 録音した木々のざわめきのパワースペクトル図 (1)

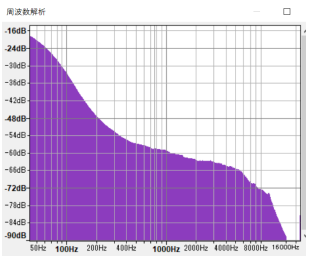

図 7.3: 録音した木々のざわめきのパワースペクトル図 (2)

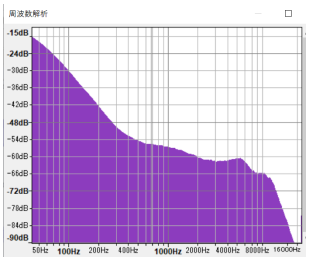

図 7.4: 録音した木々のざわめきのパワースペクトル図 (3)

また，シーン 4 のようなユーザが直接木を揺らすシーンでも接触状態に応じて衝突音と摩擦音を鳴らすことで，シーンと連動した葉擦れ音を生成することができるが，この音は図 7.5 に示す実際の木を揺らしたシーンの波形とスペクトログラムと相違がみられる．これは本手法では摩擦音に 1/f ノイズに減衰をかけたものを利用しているが，実際の木の葉は強い衝撃で擦れた際に高周波のあたりに特徴のある音になることや，枝の衝突などの葉以外の要素からなる音を考慮できていないことが原因として考えられる．さらに，ノイズベースの摩擦音では人間の耳に届く際のいくつもの音が重なり合った状態の音を再現しているため，それをさらに重ね合わせると違和感のある音に聞こえてしまうという問題もある．この問題を解決するには先述したように物理シミュレーションに基づく摩擦音を生成することや，ノイズベースであっても実際の摩擦音の 1 つの音に近い周波数特性を再現することが必要になる．

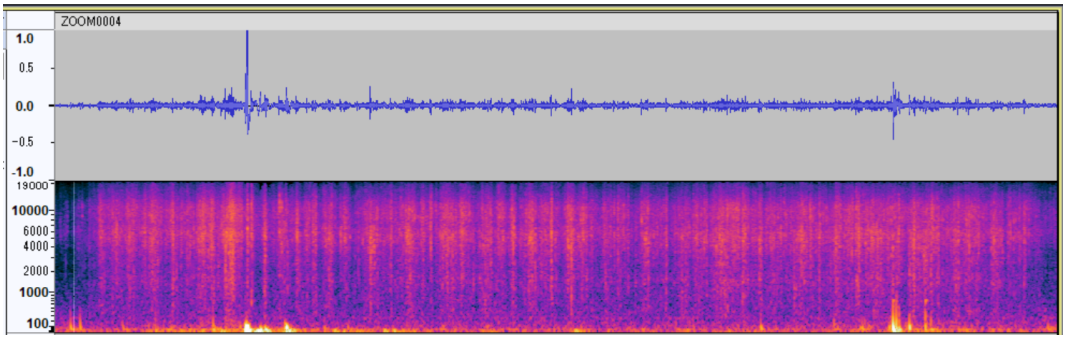

図 7.5: 録音した木を手で揺らしたシーンの波形とスペクトログラム

7.3 位置ベース法を用いた木の揺れ動きのシミュレーション

シーン 1, 2 ,3 ,4 すべてにおいて，外力に関わらず木が概形を保ったまま揺れ動くシーンがシミュレーションにより再現されている．詳しく見ると，シーン 1, 2 の高木の幹エッジは風の影響が小さく，シーン 3 の低木は木全体が大きく揺れていることが確認できる．これは幹・枝・葉ノードの質量に差があるためであり，質量が小さい方がより風の影響を受けやすく，さらにエッジ間の変形も大きいという実際の樹木の特性を再現できている．しかし，本論文では木におけるエッジを根元と先端の太さの変わらない幹，枝という 2 種類で構成

しているが，本来の樹木は同じ枝であっても先端に行くにつれて細くなり，さらに木全体でみても先端にあるエッジの方が細く大きく揺れ動く．また，本手法では両端のノードのみで離散化しているため，1 本の幹・枝は変形することの無い直線的なエッジになっている．しかし本来の樹木では同じ幹・枝内でも曲がり，ねじれが起こることで多様な揺れ方が起こっている．このような特性を再現するためには，より多様な木を構築するエッジを用意することや，幹や枝の枝分かれの箇所以外にも細かくノードを配置し，同じ枝内での変形を再現することを可能にすることが必要になると考えられる．

第8章　まとめ

本論文では，樹木シミュレーションに基づく環境音合成システムとして，木や葉の形状にユーザの指定した値をインタラクティブに反映させることで任意のパラメータを持つ樹木の揺れ動きと音を簡単に得ることができるシステムを提案した．樹木の揺れ動きにおいては，樹木の初期形状構築時に制約条件を適用する親ノードと親エッジを記憶させることで，位置ベース法を離散化した木構造に用いることを可能にし，細かい枝の揺れ動きや葉の接触までリアルタイムにシミュレーションすることを可能にした．さらに，衝突音にはバネ-質点系を用いたモード解析を，摩擦音には 1/f ノイズを用いることで，樹木の揺れ動きから発生する音を生成し，さらに 1/f ノイズに reson filter を適用することで生成した風切り音を加えることで，シーンと連動した環境音の生成も可能にした．提案システムはゲームや映画の木が登場するシーンにおいて，クリエイターがシーンと音を別々に生成する手間を削減することや，用意された音だけではない，インタラクティブな音の変化を実現するなどへの応用が期待できる．

一方で，本研究には様々な課題も残されている．樹形や葉の材質などをより詳細に設定したいユーザにとっては現状のパラメータだけでは不十分であるため，各種パラメータを詳細に設定できるようにすると共に，その入力の手間が増えないように樹種指定などにより一括で登録できるユーザインターフェースを構築することや，ユーザが手で動かすシーンにおける摩擦音の重なり合いから不自然な音に聞こえてしまう事を改善しなければならない．こうしたよりリアルな音生成のためには葉を弾性体として扱い葉の変形まで考慮した衝突音生成を可能にすることや，本手法ではノイズベースである摩擦音と風切りを物理シミュレーションに基づいた音生成手法にすることなどが必要だと考えられる．

木構造の揺れ動きにおいても広葉樹や針葉樹のような概形の異なる樹種の特徴を再現した初期形状の構築と，幹，枝についても両端点のみで離散化し太さや形状の変化しない 2 種類のエッジだけでなく，先端に向かうにつれて細くなり，より大きく揺れ動く事や，1 本の幹，枝内で変形するなどの特徴を反映した樹木の揺れ動きの再現を行うことが今後の課題である．

そして，このようにシミュレーションのリアルさを追求していくと計算量が増加することになるため，GPU による計算の並列化などを行い，再現度を高めると共に，1 本の樹木だけでなく森林シーンのような多数の樹木の存在するシーンの再現も可能にしたい．

謝辞

はじめに，自分が筑波大学学群生の頃から 3 年もの期間手厚い指導して下さった筑波大学図書館情報メディア系 藤澤誠准教授に心から感謝申し上げます．藤澤先生は，まだ研究のことをよく知らない学群生の自分が，散歩とその時に聞こえてくる音が好きというあまりに個人的な理由から提案した木々のざわめきの音生成というテーマを快く受け入れて下さり，研究の道筋を立て，その過程で何度も相談に乗り，ここまで導いて下さいました．自分の興味が持てるテーマで研究ができたこと，それに対して熱心に指導して下さったことで実りある 3 年間となりました．また，合同ゼミや発表練習の場で研究に関して多くの助言を頂いた筑波大学図書館情報メディア系 三河正彦准教授に深く感謝いたします．そして，日々研究や発表をより良いものにするために，議論を行って下さった物理ベースコンピュータグラフィックス研究室ならびにソーシャルロボット研究室の皆様にも深く感謝いたします．特に物理ベースコンピュータグラフィックス研究室の古川翔大さんと今井翔輝さんは，同期として切磋琢磨しながらここまで共に研究を進めてきました．ありがとうございました．また，自分が不自由なく研究できるように金銭面での援助を惜しまず，帰省のたびに励ましの言葉をかけてくれた家族にも深く感謝いたします．

コミュニケーション理解研究室の石田哲也さん，融合知能デザイン研究室の金承彦さん，上保研究室の船越大輝さんとは学群 1 年次から学群は違えど共に日々を過ごし，大学院入学後は同じ学位プログラムとなったことで同期の研究仲間として毎日のようにパソコンを突き合わせていました．自分の 6 年間の学生生活及び研究活動が楽しい思い出となったのは皆様のおかげです．ありがとうございました．

また，コンテンツ工学研究室の稲福和史さん，大峠和基さん，熊田大雅さん，コミュニケーション理解研究室の劉依泓さん，妹尾考さん，知識獲得システム研究室の新田洸平さん，池内淳研究室の佐藤千晴さんらとは，7D 棟 140 号室に集まり自主的な研究発表会を行ってきました．皆さんには研究に関して多くの助言を頂くとともに大学院でのあるべき過ごし方も教えていただき，コロナ禍で人との交流が少なくなってしまった大学院生活を充実したものにしてくださいました．ありがとうございました．

参考文献

[1] Matthias Müller, Bruno Heidelberger, Marcus Hennix, and John Ratcliff. Position based dynamics. *Journal of Visual Communication and Image Representation*, Vol. 18, No. 2, pp. 109–118, 2007.

[2] James F. O'Brien, Chen Shen, and Christine M. Gatchalian. Synthesizing sounds from rigid-body simulations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 175–181, 2002.

[3] Nikunj Raghuvanshi and Ming C. Lin. Interactive sound synthesis for large scale environments. In *Proceedings of the 2006 symposium on Interactive 3D graphics and games*, pp. 101–108, 2006.

[4] Kees van den Doel, Paul G. Kry, and Dinesh K. Pai. Foleyautomatic: Physically-based sound effects for interactive simulation and animation. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, p. 537–544, 2001.

[5] 松山克胤, 藤本忠博, 村岡一信, 千葉則茂. 風による樹木の揺らぎの効果音の効率的な生成法. 芸術科学会論文誌, Vol. 3, No. 1, pp. 76–85, 2004.

[6] Eston Schweickart, Doug L. James, and Steve Marschner. Animating elastic rods with sound. *ACM Transactions on Graphics (TOG)*, Vol. 36, No. 4, pp. 1–10, 2017.

[7] Gabriel Cirio, Ante Qu, George Drettakis, Eitan Grinspun, and Changxi Zheng. Multi-scale simulation of nonlinear thin-shell sound with wave turbulence. *ACM Transactions on Graphics (TOG)*, Vol. 37, No. 4, pp. 1–14, 2018.

[8] Dominik Rausch, Bernd Hentschel, and Torsten Kuhlen. Efficient modal sound synthesis on gpus. In *Proceedings of the 2014 IEEE VR Workshop: Sonic Interaction in Virtual Environments (SIVE)*, pp. 13–18, 2014.

[9] Xutong Jin, Sheng Li, Guoping Wang, and Dinesh Manocha. Neuralsound: learning-based modal sound synthesis with acoustic transfer. *ACM Transactions on Graphics (TOG)*, Vol. 41, No. 4, pp. 1–15, 2022.

[10] 中塚貴之, 森島繁生. 凝着説に基づく物体表面の弾性変形を考慮した摩擦音の生成手法の提案. 情報処理学会全国大会講演論文集, Vol. 78, No. 4, pp. 187–188, 2016.

[11] Yoshinori Dobashi, Tsuyoshi Yamamoto, and Tomoyuki Nishita. Real-time rendering of aerodynamic sound using sound textures based on computational fluid dynamics. In *Proceedings of the 2003 ACM SIGGRAPH Papers*, pp. 732–740. 2003.

[12] Bohan Wang, Yili Zhao, and Jernej Barbič. Botanical materials based on biomechanics. *ACM Transactions on Graphics (TOG)*, Vol. 36, No. 4, pp. 1–13, 2017.

[13] Shin Ota, Machiko Tamura, Tadahiro Fujimoto, Kazunobu Muraoka, and Norishige Chiba. A hybrid method for real-time animation of trees swaying in wind fields. *The Visual Computer*, Vol. 20, No. 10, pp. 613–623, 2004.

[14] Zhengze Li, Fen Kuang, and Yanci Zhang. Real-time physically plausible simulation of forest. *Graphics and Visual Computing*, Vol. 4, p. 200025, 2021.

[15] Matthias Müller, Tae-Yong Kim, and Nuttapong Chentanez. Fast simulation of inextensible hair and fur. In *Proceedings of the 9th Workshop on Virtual Reality Interaction and Physical Simulation*, pp. 39–44, 2012.

[16] Nobuyuki Umetani, Ryan Schmidt, and Jos Stam. Position-based elastic rods. In *Proceedings of the 2014 Eurographics/ ACM SIGGRAPH Symposium on Computer Animation*, pp. 21–30, 2014.

[17] Tassilo Kugelstadt and Elmar Schömer. Position and orientation based cosserat rods. In *Proceedings of the 2016 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 169–178, 2016.

[18] Simon Green. Particle simulation using cuda. *NVIDIA whitepaper*, Vol. 6, pp. 121–128, 2010.

[19] 松浦一輝. 樹木の揺らぎによる接触音の合成に関する研究. 筑波大学卒業論文, 2021.

[20] Yusuke Onoda, Mark Westoby, Peter B Adler, Amy MF Choong, Fiona J Clissold, Johannes HC Cornelissen, Sandra Díaz, Nathaniel J Dominy, Alison Elgart, Lucas Enrico, et al. Global patterns of leaf mechanical properties. *Ecology letters*, Vol. 14, No. 3, pp. 301–312, 2011.

[21] Ross Bencina and Phil Burk. Portaudio - an open source cross platform audio API. In *Proceedings of the 2001 International Computer Music Conference*, 2001.

[22] 小松正史, 加藤徹, 桑野園子, 難波精一郎, 近藤明, 井上義雄, 山口克人. 樹木葉擦れ音の物理特性. 騒音制御, Vol. 24, No. 4, pp. 268–276, 2000.

参考論文

(1) 松浦一輝，藤澤誠，三河正彦. 樹木シミュレーションに基づくリアルタイム環境音合成システムの開発. 画像電子学会 第 301 回研究会. pp.13:1-4, 2022.