図書館情報メディア研究科修士論文

視覚的なパラメータ設定に基づく安定した炎のシミュレーション

2021年3月

201921638

田代　祐美伽

視覚的なパラメータ設定に基づく
安定した炎のシミュレーション
Flame Simulation Based on Visual Parameters

学籍番号：201921638
氏名：田代　祐美伽
Tashiro Yumika

コンピュータグラフィックス (CG) の分野において，自然現象をより正確に再現することができる物理シミュレーション技術の研究が数多く行われている．中でも，炎は映画やゲームなどでも多用される自然現象の一つであり，様々なシミュレーション手法が研究されてきた．炎は流体と同じ性質を持つため，流体シミュレーション手法の一種である粒子法を用いた先行研究が多くある．最新の研究では，GPU などによる並列計算によりリアルタイムでのシミュレーションも可能となっているが，力学ベースでのシミュレーションは計算結果が発散しやすく，現象の複雑さから多くのパラメータを調整しなければならず，形状制御をしながら安定したシミュレーション結果を得ることが難しい．そこで本研究では，位置ベースの PBF(Position Based Fluid) と陰解法を用いて計算を安定させ，視覚的なパラメータによってユーザが容易に形状を制御できるような炎のシミュレーション手法を提案する．

先行研究では力学ベースでシミュレーションを行っており，加速度や速度を時間積分して位置を計算している．そのため，積分による誤差が蓄積し計算結果が発散しやすい．それに対し，制約条件に近づくよう反復的に位置を修正していく位置ベースでのシミュレーション (PBF) を導入することで，不自然な挙動を見せたり発散してしまったりすることをなくすことができる．また，炎のシミュレーションでは熱の計算も必要となるが，従来法では陽的な手法を用いていたため，ユーザが熱の伝わりを表すパラメータやタイムステップ幅を大きくしたときに不安定になっていた．ユーザによるパラメータ制御を容易にするために，本研究では熱計算に陰解法を適用する．以上のように，PBF で位置計算を，陰解法で熱計算を安定化させることで，パラメータを変更しても安定した計算結果が得られるシミュレーションを行う．

安定したシミュレーション手法に加え，ユーザが簡単に炎の形状を制御できるようにするため，炎の規模を変えるパラメータ volume，高さを変えるパラメータ height，広がり具合いを変えるパラメータ spread，乱れ具合いを変えるパラメータ turbulence の 4 つのパラメータを提案する．物理的なパラメータから視覚的な形状変化に直接対応するパラメータに変換することで，CG や物理学に精通していないユーザでも容易に制御できる炎の CG シミュレーションを実現する．

以上の手法を実装し，炎らしいシミュレーション結果が得られることと，提案する 4 つのパラメータが実際に視覚的な形状変化と対応することを確認した．しかし，パラメータの数値と炎の大きさが線形に変化していないパラメータがあるため，フィードバック制御などを用いて調整する必要があることや，リアルタイムでのシミュレーションが実現できなかったため，GPU を用いた高速化が今後の課題である．

研究指導教員：三河　正彦
副研究指導教員：藤澤　誠

視覚的なパラメータ設定に基づく
安定した炎のシミュレーション

筑波大学
図書館情報メディア研究科
2021年3月
田代　祐美伽

目 次

第 1 章　序論 **1**
1.1　研究背景と目的 1
1.2　先行研究 2
1.2.1　炎のシミュレーション 2
1.2.2　Position Based Dynamics 3
1.3　本論文の構成 4

第 2 章　提案手法 **5**
2.1　シミュレーション方法の概要 5
2.2　SPH 法 6
2.2.1　物理量の定義 6
2.2.2　支配方程式の離散化 7
2.2.3　境界粒子 9
2.3　PBF による位置計算の安定化 9
2.3.1　PBD の考え方 9
2.3.2　PBD の流体への適用 12
2.4　陰解法による熱計算の安定化 13
2.5　視覚的なパラメータ設定 14
2.5.1　炎の大きさの計測方法 15
2.5.2　volume パラメータ 15
2.5.3　spread パラメータと height パラメータ 16
2.5.4　turbulence パラメータ 21
2.6　レンダリング 23

第 3 章　結果 **24**
3.1　陰解法による熱計算の安定化 27
3.2　volume による変化 31
3.3　height による変化 33
3.4　spread による変化 35
3.5　turbulence による変化 37
3.6　パラメータの組み合わせ 38

第 4 章　考察 **39**

第 5 章　結論 **40**

謝辞 **41**

参考文献 **42**

図 目 次

1.1 本物の炎の写真 (著者撮影) 1
1.2 Stam らのシミュレーション結果 [1] 2
1.3 稲垣らのシミュレーション結果 [2] 3
1.4 佐々木らのシミュレーション結果 [3] 3
1.5 Macklin らによる PBF を用いた
流体のシミュレーション結果 [4] 4
1.6 Barreiro らによる粘性流体の
シミュレーション結果 [5] 4

2.1 本研究の概要 5
2.2 シミュレーションの手順 6
2.3 粒子 p とその近傍粒子 q のイメージ 7
2.4 炎粒子と境界粒子 9
2.5 速度と時間のグラフ 10
2.6 理想とするシミュレーション結果 10
2.7 Δt が大きすぎた場合のシミュレーション結果 11
2.8 炎の大きさの計測イメージ 15
2.9 volume パラメータによる変化のイメージ 16
2.10 spread パラメータによる変化のイメージ 16
2.11 height パラメータによる変化のイメージ 17
2.12 ΔT を変えた結果 17
2.13 n^{gene} を変えた結果 18
2.14 α を変えた結果 18
2.15 k を変えた結果 19
2.16 ν を変えた結果 19
2.17 β を変えた結果 20
2.18 β と ΔT を変えた場合 20
2.19 表 2.7 の結果と近似した二次関数 2.49 のグラフ 21
2.20 turbulence パラメータによる変化のイメージ 22
2.21 ϵ_{vort} を変えた結果 22
2.22 炎色テクスチャ 23

3.1 3 フレーム目 25
3.2 20 フレーム目 25
3.3 40 フレーム目 25
3.4 60 フレーム目 25
3.5 80 フレーム目 25
3.6 100 フレーム目 25

3.7 $k = 0.01$ . . . 27
3.8 $k = 0.5$ . . . 27
3.9 $k = 3$ . . . 28
3.10 $k = 0.01$ . . . 28
3.11 $k = 0.5$ . . . 29
3.12 $k = 3$ . . . 29
3.13 $k = 0.01$ / 20 フレーム目 . . . 30
3.14 $k = 0.01$ / 50 フレーム目 . . . 30
3.15 volume=0.0 / 9.62fps . . . 31
3.16 volume=0.2 / 8.37fps . . . 31
3.17 volume=0.4 / 7.35fps . . . 31
3.18 volume=0.6 / 6.54fps . . . 31
3.19 volume=0.8 / 5.45fps . . . 31
3.20 volume=1.0 / 4.94fps . . . 31
3.21 volume パラメータによる横幅の変化 . . . 32
3.22 height=0.0 / 10.86fps . . . 33
3.23 height=0.2 / 10.37fps . . . 33
3.24 height=0.4 / 9.99fps . . . 33
3.25 height=0.6 / 9.36fps . . . 33
3.26 height=0.8 / 8.84fps . . . 33
3.27 height=1.0 / 8.07fps . . . 33
3.28 height パラメータによる高さ l_y の変化 . . . 34
3.29 spread=0.0 / 9.62fps . . . 35
3.30 spread=0.2 / 8.54fps . . . 35
3.31 spread=0.4 / 7.64fps . . . 35
3.32 spread=0.6 / 6.83fps . . . 35
3.33 spread=0.8 / 6.25fps . . . 35
3.34 spread=1.0 / 5.91fps . . . 35
3.35 spread パラメータによる広がりの変化 . . . 36
3.36 volume=0.0 / 9.62fps . . . 37
3.37 turbulence=0.2 / 9.53fps . . . 37
3.38 volume=0.4 / 9.45fps . . . 37
3.39 turbulence=0.6 / 9.60fps . . . 37
3.40 volume=0.8 / 9.75fps . . . 37
3.41 turbulence=1.0 / 9.74fps . . . 37
3.42 横幅も縦幅も大きく，広がる炎 . . . 38
3.43 中規模で縦に細長い炎 . . . 38
3.44 小さく丸みのある炎 . . . 38
3.45 細長く，乱れた炎 . . . 38

第1章　序論

1.1　研究背景と目的

近年，映画やゲームなど数多くの映像作品で自然現象のコンピュータグラフィックス (CG) が多用されている．多くの自然現象はその複雑さから，アーティストによる手作業での再現が困難であり，そのため，CG の研究分野では様々な自然現象を物理法則に基づいて自動的にシミュレーションする技術が研究されている．中でも炎は非常に身近な自然現象であり，エンターテインメントとして演出等に多用されやすいだけでなく，そのシミュレーションは災害被害予測等の学術的な用途にも利用される．そのため，できるだけ高速なシミュレーション，そして写実的なレンダリング方法が研究されてきた．

炎は，物質が酸素によって急激に酸化する際に，光と熱によってエネルギーを放出する現象であると考えられている．様々な化学反応・状態変化を伴いながら流体と同じ非線形な挙動を見せ，発生メカニズムは単純でありつつも周囲の物質の変化に左右されるため，完全に再現することは容易ではない．人間がずっと見ていても飽きないと言われるほど挙動が複雑な現象である (図 1.1)．炎に関する反応・挙動すべてを再現しようとすると計算時間が膨大になってしまうため，エンタテインメントの現場における炎の CG は，現象の一部のみの再現やデザイナーのイラストに基づいた単純なアニメーションの作成にとどまっている．近年，GPGPU(General Purpose GPU) の普及によって流体の挙動をリアルタイムでシミュレーション・レンダリングすることが可能になったため，それらを利用した手法の更なる研究もなされている [3].

図 1.1: 本物の炎の写真 (著者撮影)

周囲の物質の変化まで考慮し，かつ GPU を使用してリアルタイムで炎のシミュレーションを行った先行研究として佐々木らの研究がある．彼らの手法では，周囲の酸素との反応も考慮しており，より物理学に忠実なシミュレーションであるが，その分シミュレーションが

複雑であり，多くのパラメータを計算結果が発散しないよう調整しなければならない．また，時間積分をする力学ベースでのシミュレーションであるため，パラメータによっては積分による誤差の蓄積で計算結果が発散してしまう．更に，それぞれの物理パラメータを変更することで炎が視覚的にどう変化するかは，物理学や CG に精通したユーザでないとわかりづらく，不自然な挙動が起こらないように制御することが難しい．

本研究では，計算を安定化することで，不自然な挙動を起こすことなくパラメータを変更することを可能にする．更に，物理学や CG に精通していないユーザも簡単に形状を変更できるよう，物理的なパラメータから視覚的でわかりやすいパラメータへの変換を提案する．

1.2 先行研究

1.2.1 炎のシミュレーション

炎は流体の一種とも考えられるため，流体のシミュレーション手法を応用することでシミュレーションが可能となる．炎のシミュレーションに関する先行研究でも特に古く基礎的なものに Stam らの方法 [1] がある．blob と呼ばれる粒子を用いてシミュレーションを行い，燃料との熱のやり取りや燃焼による温度の減衰まで考慮した温度計算を行い，温度から色を計算してレンダリングしている (図 1.2)．また，Nguyen ら [6] は，ガスやすすに関しても考慮し，流体シミュレーションにおいて液体表面追跡に用いられるレベルセット法を炎に応用した．火炎や煙と物体の相互作用も再現可能とすることで物理的に忠実に炎を再現しているが，その分計算時間が必要となり，リアルタイムでのシミュレーションは難しい．一方で，Zhang ら [7] は短いシミュレーションを使用して長いアニメーションシーケンスを生成する手法で炎を再現した．この方法は，ゲームなどのエンターテインメントの場面では有用であると考えられるが，リアルタイムではなく，物理学に忠実でもない．

図 1.2: Stam らのシミュレーション結果 [1]

リアルタイムでの炎のシミュレーション手法として，稲垣らの研究 (図 1.3) や佐々木らの研究 (図 1.4) がある [2][3]．高速な流体シミュレーション手法である SPH 法 (Smoothed

Particle Hydrodynamics) を用いることで，流体の挙動を忠実に再現し，かつリアルタイムでのシミュレーションを可能にしている．特に佐々木らの研究では，稲垣らの研究に加えて周囲の空気の動きや熱のやり取り，燃焼物の化学反応による変化などまで考慮することで，燃焼過程まで含めた炎のシミュレーションを可能としている．炎のシミュレーションに関する研究の中でもこれらの手法は実際の炎の再現性が高く，かつ，リアルタイム性を実現しているものである．しかし，複数の手法を組み合わせて用いるなどシミュレーションが複雑で，非常に多くのパラメータを調整する必要があり，ユーザ制御が困難という問題点がある．また，計算の安定性が低く，シミュレーションを大規模化する際にタイムステップ幅を大きくできず，高速化できないといった問題もある．本研究では，位置計算を力学ベースから位置ベースに，熱計算を陽解法から陰解法に変換することで計算を安定化する．計算を安定化することで炎の形状制御が容易になるため，視覚的なパラメータへの変換も行い，物理学や CG に精通していないユーザにも簡単な制御を可能にする．

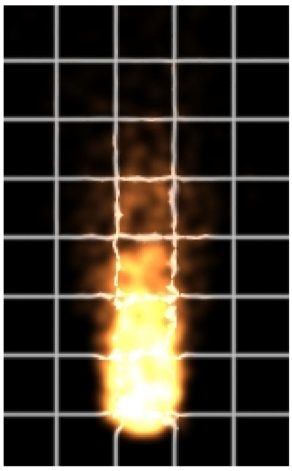


図 1.3: 稲垣らのシミュレーション結果 [2]

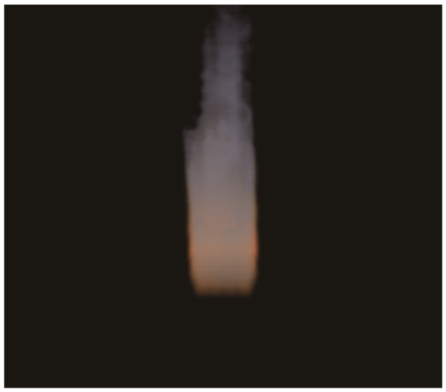
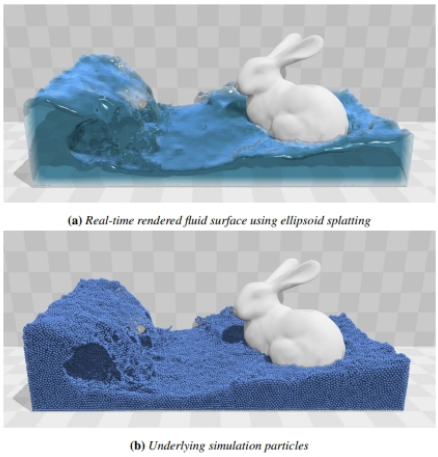


図 1.4: 佐々木らのシミュレーション結果 [3]

1.2.2 Position Based Dynamics

CG における物理シミュレーションで計算を安定化することができる方法として，位置ベース法 (Position Based Dynamics)[8] がある．剛体，弾性体，流体など様々な物体のシミュレーションに適用できる手法 [9] で，流体に適用するものは特に PBF(Position Based Fluid) と呼ばれる [4](図 1.5)．PBF では，シミュレーションによるエネルギー減衰の補填も考慮されている．また，粘性流体のシミュレーションの研究 [5][10](図 1.6) も行われており，様々な流体のシミュレーション手法として有用であることが知られているが，PBF で炎のシミュレーションを行った研究はない．そこで，本研究では，計算の安定化に有用なこの PBF を用いて炎のシミュレーションを行うことで，ユーザが容易にパラメータで制御できる炎の CG を実現する．



(a) *Real-time rendered fluid surface using ellipsoid splatting*

(b) *Underlying simulation particles*

図 1.5: Macklin らによる PBF を用いた流体のシミュレーション結果 [4]

図 1.6: Barreiro らによる粘性流体のシミュレーション結果 [5]

1.3 本論文の構成

まず，第 2 章で提案手法の詳細を説明する．2.1 節でシミュレーション方法の概要と手順を述べ，2.2 節から 2.4 節は，本研究に用いた手法をそれぞれ説明していく．2.5 節では，シミュレーションの結果から視覚的なパラメータへの変換を提案し，第 3 章ではそのパラメータを用いたシミュレーション結果を示す．第 4 章で各パラメータとシミュレーション結果の考察を行い，最後に第 5 章で結論を述べる．

第2章　提案手法

本章では，本研究で提案するシミュレーション手法を解説する．

2.1　シミュレーション方法の概要

多くの先行研究と同様に，本研究でも粒子法を使って炎の挙動をシミュレーションしていく．粒子法は，計算空間を限定しない・移流項を考えなくてよい・並列計算向きでリアルタイム化が容易といったメリットがあり，CG 向けの高速な流体シミュレーションに向いている．粒子法によるシミュレーション方法は稲垣ら [2] や佐々木ら [3] の先行研究と同じく SPH 法をベースとする．そこに，水や粘性流体のシミュレーションをする際に用いられ，安定した位置計算を実現する PBF[4] を取り入れることで，安定した炎のシミュレーション手法を提案する．PBF を用いることで粒子の位置分布による計算の不安定化は防げるが，パラメータ設定による熱計算での不安定化は防げない．そこで，更に熱計算に陰解法を採用することで，どのような熱のパラメータでも安定した計算を可能にする．以上のようにして，安定した炎のシミュレーションを実現することで安定性のためのパラメータ調整の必要性を少なくできるが，ユーザが望む炎の形を得るためには形状とは 1 対 1 で対応しない多くの物理パラメータ調整が必要となる．そこで，シミュレーションに使用している物理的なパラメータを視覚的にわかりやすいパラメータに変換することで，物理学や CG に精通していないユーザも容易に炎の形状を変更できるような UI を実現する．研究の全体像を下の図 2.1 に示す．

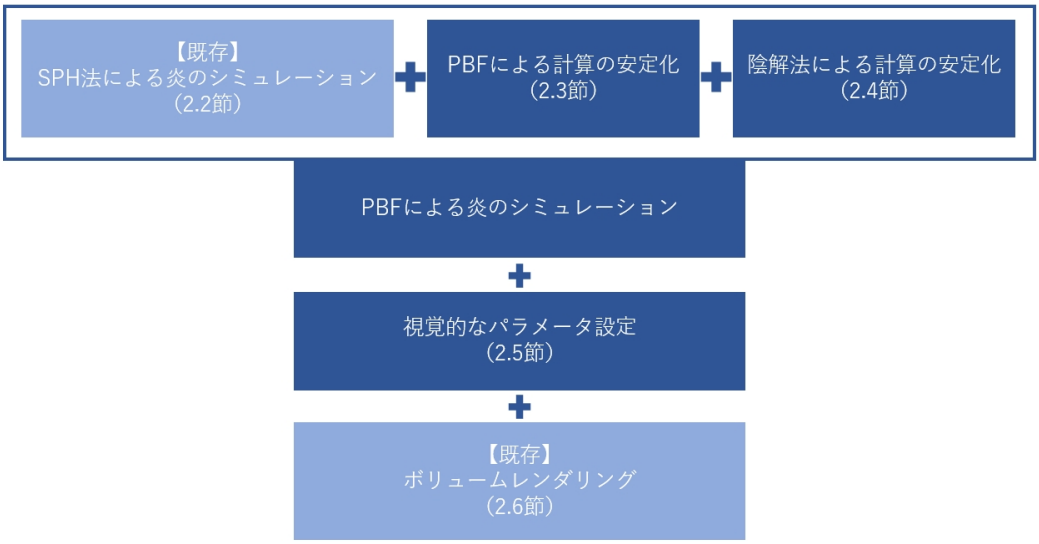


図 2.1: 本研究の概要

図 2.2 に，本研究でのシミュレーション手順を示す．炎は粒子の集合体として近似され，各粒子は，質量，位置，速度，加速度，密度，温度，熱伝導率の情報をそれぞれ持つ．炎は物質が燃焼し光と熱を放出する現象であり，本研究ではその挙動のみを再現する (周囲の物質や化学変化などは考慮しない)．そのため，熱源から粒子を噴出させ，1 ステップごとに粒子の位置・速度，密度や温度を更新していく．そして，粒子の密度や温度に応じてボリュームレンダリングする．

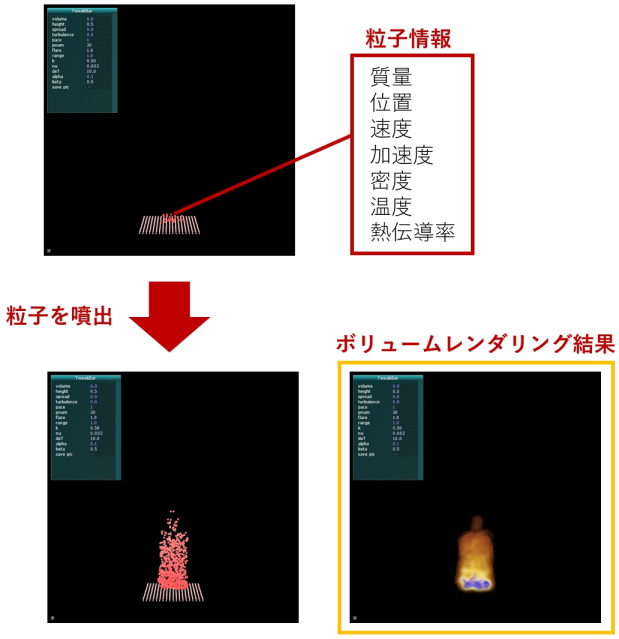


図 2.2: シミュレーションの手順

粒子は，シミュレーション空間の底面の中心を中心とする半径 R の円内から，ランダムに発生させる．1 ステップに発生させる粒子の数はユーザが任意に定めることができる．また，炎が穂のような形状をしているのは，中心に上昇気流が発生してるためである．これを再現するため，中心からの距離が短いほど y 方向の初速を速く設定し，かつ，ある一定の温度 $T_{\min}$ 以下になった粒子は，放出するエネルギーが弱まったとみなし消去する．

2.2 SPH 法

本節では，炎シミュレーションのベースとなる SPH 法について説明する．

2.2.1 物理量の定義

粒子法は，任意の粒子位置 (空間中の点) において密度や温度などの物理量を離散的に定義する方法であるが，支配方程式に含まれる空間微分を計算するため，それらの物理量を連続量として近似するための式が必要になる．そこで用いられるのが，カーネル関数を用いて任意位置での物理量を関数として表す SPH 法である [3]．SPH 法では，次の式を用い

て粒子 p(中心位置 $\boldsymbol{p}$) における任意の物理量 ϕ を定義する．

$$\phi(\boldsymbol{p}) = \sum_{q \in N} m_q \frac{\phi_q}{\rho_q} W(\boldsymbol{r}, h) \tag{2.1}$$

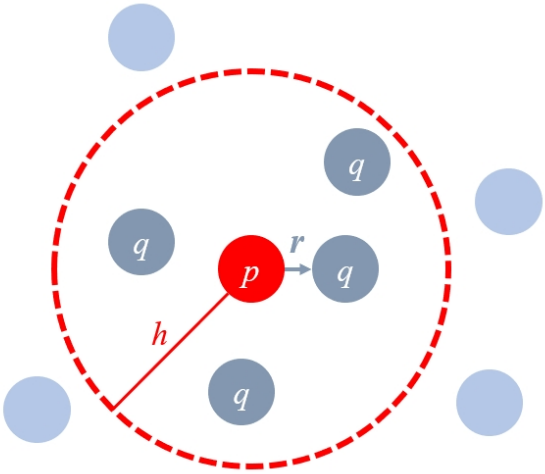


図 2.3: 粒子 p とその近傍粒子 q のイメージ

q は粒子 p を中心とする半径 h(=有効半径) の範囲内にある近傍粒子を示し (図 2.3)，N は近傍粒子集合を表す．m は質量，$\boldsymbol{r}$ は粒子 p から粒子 q へのベクトル，ρ は密度，W はカーネル関数である．有効半径 h は，有効半径の球内に，粒子が近傍粒子の個数の最大値 $N_{\max}$ 個存在する時に，密度が静密度 ρ_0 となるように設定する．三次元の場合 $h = \sqrt[3]{\frac{3mN_{\max}}{4\pi\rho_0}}$ となる．定義式 (式 2.1) より，例えば粒子 p の密度 ρ_p は次式のように求められる．

$$\rho_p = \sum_{q \in N} m_q W(\boldsymbol{r}, h) \tag{2.2}$$

SPH 法ではカーネル関数 W に $\int W = 1$ などの条件をつけることで，任意の物理量の勾配とラプラシアンを次のように表すことができる．

$$\nabla\phi(\boldsymbol{p}) = \sum_{q \in N} m_q \frac{\phi_q}{\rho_q} \nabla W(\boldsymbol{r}, h) \tag{2.3}$$

$$\nabla^2\phi(\boldsymbol{p}) = \sum_{q \in N} m_q \frac{\phi_q}{\rho_q} \nabla^2 W(\boldsymbol{r}, h) \tag{2.4}$$

2.2.2 支配方程式の離散化

炎を流体として捉えるため，流体の運動を記述する方程式である Navier-Stokes 方程式 (式 2.5,2.6) に従って，粒子の動きをシミュレーションしていく．

$$\nabla \cdot \boldsymbol{v} = 0 \tag{2.5}$$

$$\frac{\partial \boldsymbol{v}}{\partial t} = -(\boldsymbol{v} \cdot \nabla)\boldsymbol{v} + \nu\nabla^2\boldsymbol{v} - \frac{1}{\rho}\nabla P + \boldsymbol{F}^{ext} \tag{2.6}$$

ここで，$\boldsymbol{v}$ は粒子速度，P は圧力である．ν は動粘性係数で，この値が大きいほど流体は粘っこくなる．式 2.5 は質量保存式であり，ある微小領域で流入する流体と流出する流体の量が等しいことを表す．式 2.6 は運動量保存式であり，第一項は流体が流体自身の流れに従って移動することを表す移流項である．第二項は粘性拡散項で，流体の粘っこさを表す項である．第三項が圧力項で，圧力分布を均一にしようとする力，すなわち密度を一定に保とうとする力を表す項であり，第四項 $\boldsymbol{F}^{ext}$ が浮力や重力などを表す外力項である．粒子法においては，粒子自体が移動するため移流項を計算する必要がない．また，粒子自体に質量を持たせているので，質量保存式を計算する必要もないため，次式のように簡略化できる．

$$\frac{D\boldsymbol{v}}{Dt} = \nu\nabla^2\boldsymbol{v} - \frac{1}{\rho}\nabla P + \boldsymbol{F}^{ext} \tag{2.7}$$

ここで $\frac{D}{Dt}$ は時間微分に移流項を含む物質微分を表す．式 2.7 右辺第 1 項，第 2 項を式 2.3,2.4 を用いて離散化することで，SPH 法において粒子にかかる力は以下のように表すことができ，これに従って各粒子の速度を更新していく．

$$\rho_p\frac{\partial\boldsymbol{v}}{\partial t} = \boldsymbol{F}_p^{visc} + \boldsymbol{F}_p^{pres} + \boldsymbol{F}_p^{ext} \tag{2.8}$$

$$\boldsymbol{F}_p^{visc} = \nu\sum_{q\in N} m_q\frac{\boldsymbol{v}_q - \boldsymbol{v}_p}{\rho_q}\nabla^2 W_{viscosity}(\boldsymbol{r}, h) \tag{2.9}$$

$$\boldsymbol{F}_p^{pres} = -\sum_{q\in N} m_q\frac{P_p + P_q}{2\rho_q}\nabla W_{spiky}(\boldsymbol{r}, h) \tag{2.10}$$

$$P_p = G(\rho_p - \rho_0) \tag{2.11}$$

$$\boldsymbol{F}_p^{ext} = \alpha\rho_p\boldsymbol{g} - \beta(T_p - T_{amb})\boldsymbol{g} \tag{2.12}$$

ここで，ρ_0 は静密度，G はガス定数，T は温度，T_{amb} は環境温度，$\boldsymbol{g}$ は重力であり，α, β はそれぞれ重力と浮力を調節するユーザ指定のパラメータである．外力項の計算は Fedkiw ら [11] と同様の式 (式 2.12) を用いる．ただし，粘性拡散項を上記の方法で計算すると，動粘性係数 ν が大きい場合にシミュレーションが不安定になるという問題があるため，XSPH 人工粘性 [10] を用いて次のように求める．

$$\boldsymbol{v}_i^{new} = \boldsymbol{v}_i + \epsilon_{visc}\sum_j \boldsymbol{v}_{ij}\cdot W(\boldsymbol{p}_i - \boldsymbol{p}_j, h) \tag{2.13}$$

ここで，$\boldsymbol{v}_{ij}$ は $\boldsymbol{v}_{ij} = \boldsymbol{v}_j - \boldsymbol{v}_i$，$\epsilon_{visc}$ は粘性の調整パラメータであり，本研究では 1.0×10^{-4} としている．

使用したカーネル関数は Müller ら [12] と同様に，密度計算には $Poly6$ カーネル (式 2.14)，勾配を使う圧力計算には $Spiky$ カーネル (式 2.15)，ラプラシアンを使う粘性拡散の計算には $Viscosity$ カーネル (式 2.16) である．

$$W(\boldsymbol{r}, h) = \frac{315}{64\pi h^9}\begin{cases}(h^2 - \|\boldsymbol{r}\|^2)^3 & \text{if } 0 \le \|\boldsymbol{r}\| \le h\\ 0 & \text{otherwise}\end{cases} \tag{2.14}$$

$$\nabla W(\boldsymbol{r}, h) = -\frac{45}{\pi h^6}\begin{cases}(h - \|\boldsymbol{r}\|)^2\dfrac{\boldsymbol{r}}{\|\boldsymbol{r}\|} & \text{if } 0 \le \|\boldsymbol{r}\| \le h\\ 0 & \text{otherwise}\end{cases} \tag{2.15}$$

$$\nabla^2 W(\boldsymbol{r}, h) = \frac{45}{\pi h^6}\begin{cases}(h - \|\boldsymbol{r}\|) & \text{if } 0 \le \|\boldsymbol{r}\| \le h\\ 0 & \text{otherwise}\end{cases} \tag{2.16}$$

以上のようにして，シミュレーション空間内にあるすべての粒子に関して式 2.8 よりかかる力を算出し，それぞれ加速度を更新する．そこから速度を更新し，更にその速度から位置を更新するのが，力学ベースでのシミュレーション手法である．

2.2.3 境界粒子

流体シミュレーションでは流体内部の力の影響だけでなく，壁面などの境界処理も考える必要がある．粒子法では単純に壁面と粒子の衝突処理で境界を考慮できるが，シミュレーション空間の境界付近において，密度が低くなり粒子が溜まってしまう問題 (Particle Stacking) を解消するため，境界粒子を用いる．

境界粒子は図 2.4 に示すように境界内に固定した粒子を配置することで Particle Stacking を防ぐ方法であり，本研究では炎粒子の挙動に対して影響が大きい底面にのみ境界粒子を配置する．炎粒子への影響を正確に再現するには，底面には複数層の境界粒子を大量に敷き詰める必要がある．しかし，近傍粒子探索にかかる時間が増えてしまうため，仮想体積を利用することで境界粒子の数を減らして計算する手法を用いる [13]．

境界粒子 p_{border} の仮想体積 $V_{p_{border}}$ は，近傍の境界粒子 q_{border} から次の式で計算する．

$$V_{p_{border}} = \frac{m_{p_{border}}}{\sum_{q_{border}} m_{q_{border}} W(\boldsymbol{r}_{p_{border}q_{border}}, h)} \tag{2.17}$$

炎粒子の密度計算を行う際，境界粒子の質量 $m_{p_{border}}$ は，この仮想体積 $V_{p_{border}}$ を用いて $\rho_0 V_{p_{border}}$ として計算する．

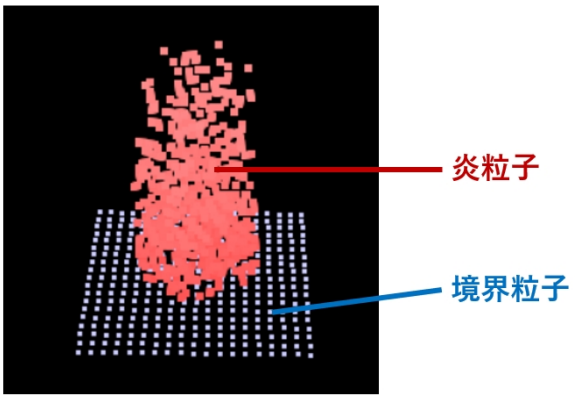


図 2.4: 炎粒子と境界粒子

2.3 PBF による位置計算の安定化

既存の SPH 法を用いた炎のシミュレーション手法では，パラメータによっては計算が不安定になり，粒子が大きく振動しながら座標値が大きくなり，計算領域外に飛んでいってしまう．本研究では，この不安定性の問題を粒子位置計算と熱計算の 2 つに分けてそれぞれ解決していく．本節では，粒子位置を安定させるため，PBF(Position Based Fluid) を用いる．PBF は，PBD(Position Based Dynamics) の一種で，流体のシミュレーションを安定させるための手法である．

2.3.1 PBD の考え方

前述の力学ベースの計算方法では，計算ステップを進めると，速度は加速度，位置は速度の時間積分をしている形になる．よって，積分による誤差の蓄積が不安定性を生み出し，

結果としてタイムステップ幅 Δt を大きくできない．例えば，図 2.5 に示すように，速度を時間のグラフで表すとグラフの面積が移動距離であるから，Δt あたり黄緑色の面積分の誤差が生まれ，Δt 秒後のシミュレーション結果として実際の $\Delta t + \varepsilon$ 秒後の結果が描画されてしまうことになる．

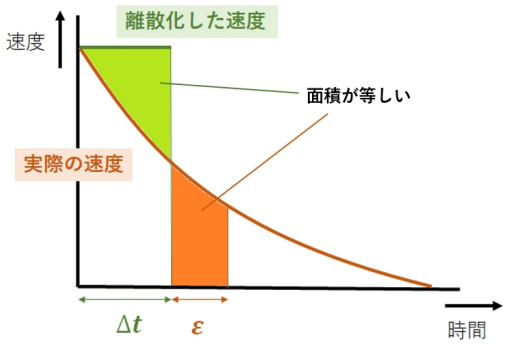


図 2.5: 速度と時間のグラフ

一方，位置ベース (PBD) の計算方法では，速度や加速度から位置を求める代わりに最終的に収束すべき位置の制約条件をかけるため，不自然な挙動を見せることなく計算が安定し，物理シミュレーションに有用であることが知られている [9]．例えば，ゴム紐の両端 (点 p_1, p_2) を引っ張り，それが元の長さに戻るまでのシミュレーションを考える．実際にゴム紐を引っ張った場合，弾性力によって縮む方向に力がかかり加速度が働く．そして同じ方向に速度が働き，徐々に点 p_1, p_2 間がゴム紐の自然長となる位置に戻るはずである (図 2.6)．

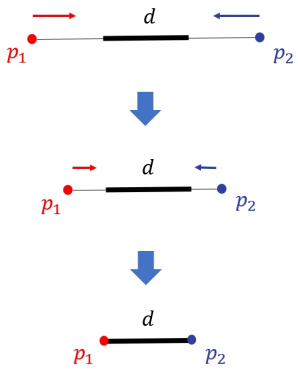


図 2.6: 理想とするシミュレーション結果

力学ベースのシミュレーションにおいて Δt が大きすぎた場合，1 ステップで移動する距離が大きくなり，Δt の値によっては両端点 p_1, p_2 の座標が元の長さを超えて図 2.7 に示すように位置が逆転する．そして，その長さがステップを進める毎に長くなっていき，結果としてシミュレーションが発散してしまう．

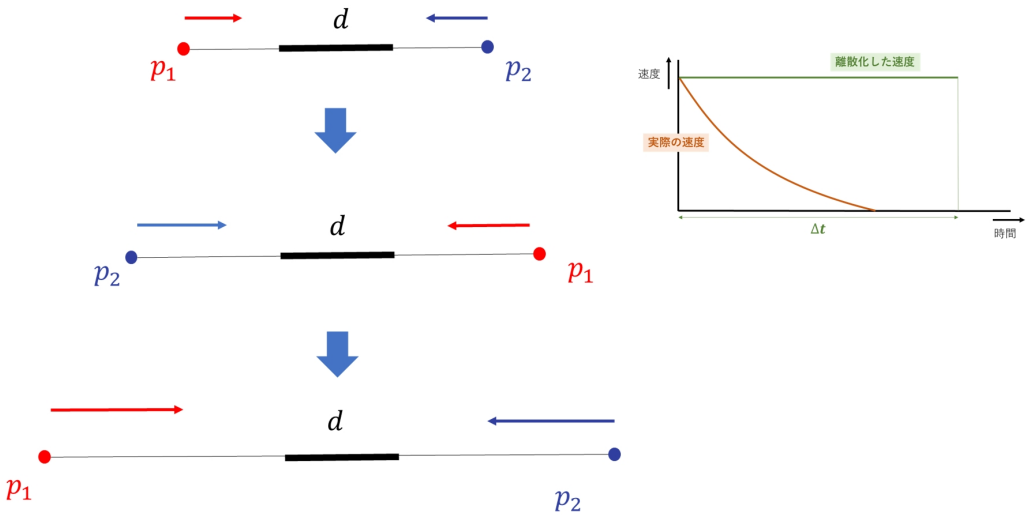


図 2.7: Δt が大きすぎた場合のシミュレーション結果

位置ベースでシミュレーションを行う場合，計算点 $\boldsymbol{p}_1, \boldsymbol{p}_2$ 間の距離 $\|\boldsymbol{p}_2 - \boldsymbol{p}_1\|$ がゴム紐の自然長 d_0 となるよう制約式 $C_{spring}(\boldsymbol{p}_1, \boldsymbol{p}_2) = \|\boldsymbol{p}_2 - \boldsymbol{p}_1\| - d_0 = 0$ をかけ，その制約条件に近づくよう位置を反復的に修正する方法をとる．位置を直接計算するので制御しやすく，誤差の蓄積もないため，不自然に発散することなく計算が安定する．

制約式による位置修正の方法を説明する．前述の通り，計算点 $\boldsymbol{p}$ に対して制約条件 $C(\boldsymbol{p}) = 0$ をかけるとき，位置修正量を $\Delta\boldsymbol{p}$ とすると，修正後の位置 $\boldsymbol{p} + \Delta\boldsymbol{p}$ も制約条件を満たすので，次の式が成り立つ．

$$C(\boldsymbol{p} + \Delta\boldsymbol{p}) = 0 \tag{2.18}$$

この式をテイラー展開すると

$$C(\boldsymbol{p} + \Delta\boldsymbol{p}) = C(\boldsymbol{p}) + \nabla_p C(\boldsymbol{p}) \cdot \Delta\boldsymbol{p} + O(\|\Delta\boldsymbol{p}\|)^2 = 0 \tag{2.19}$$

したがって，制約条件は次のように近似できる．

$$C(\boldsymbol{p}) + \nabla_p C(\boldsymbol{p}) \cdot \Delta\boldsymbol{p} = 0 \tag{2.20}$$

$C(\boldsymbol{p})$ は物体内部の状態に対する制約なので，その勾配方向 ∇C は物体全体の動きに影響せず，その方向に計算点 $\boldsymbol{p}$ を動かせば全体の運動量保存を満たせる．$\Delta\boldsymbol{p}$ を $\nabla_p C$ の方向に制限すると，次式が得られる．

$$\Delta\boldsymbol{p} = s\nabla_p C(\boldsymbol{p}) \tag{2.21}$$

ここで，s は未知のスケーリング係数であるが，式 2.20 から求めることができる．

$$s = -\frac{C(\boldsymbol{p})}{\|\nabla_p C(\boldsymbol{p})\|^2} \tag{2.22}$$

よって求める $\Delta\boldsymbol{p}$ は

$$\Delta\boldsymbol{p} = -\frac{C(\boldsymbol{p})}{\|\nabla_p C(\boldsymbol{p})\|^2} \cdot \nabla_p C(\boldsymbol{p}) \tag{2.23}$$

これを計算点が複数ある場合に拡張し，計算点 $\boldsymbol{p}_1, \boldsymbol{p}_2, ...\boldsymbol{p}_n$ に制約条件 $C(\boldsymbol{p}_1, \boldsymbol{p}_2, ...\boldsymbol{p}_n) = 0$ をかける場合を考える．計算点 $\boldsymbol{p}_i$ の質量を m_i とすると，その逆数 $w_i = \frac{1}{m_i}$ を重みとして，$\Delta\boldsymbol{p}_i$ は次の式で求めることができる．

$$\Delta\boldsymbol{p}_i = s_i w_i \nabla_{\boldsymbol{p}_i} C(\boldsymbol{p}) \tag{2.24}$$

$$s_i = -\frac{C(\boldsymbol{p})}{\sum_j w_j \|\nabla_{\boldsymbol{p}_j} C(\boldsymbol{p})\|^2} \tag{2.25}$$

以上のようにして修正した $\boldsymbol{p}$ と，1 ステップ前の位置 $\boldsymbol{p}_{prev}$ より，速度 $\boldsymbol{v}$ を更新する．

$$\boldsymbol{v} \leftarrow \frac{\boldsymbol{p} - \boldsymbol{p}_{prev}}{\Delta t} \tag{2.26}$$

速度 $\boldsymbol{v}$ は各計算ステップの最後に求め，次のステップの位置更新に用いることで慣性による運動を表すことができる．

先ほどのゴム紐の例での位置修正は以下のようになる．制約式は前述の通り

$$C^{spring}(\boldsymbol{p}_1, \boldsymbol{p}_2) = \|\boldsymbol{p}_2 - \boldsymbol{p}_1\| - d_0 = 0 \tag{2.27}$$

となる．計算点 $\boldsymbol{p_1}, \boldsymbol{p_2}$ それぞれに関して偏微分すると

$$\nabla_{\boldsymbol{p}_1} C^{spring}(\boldsymbol{p}_1, \boldsymbol{p}_2) = -\frac{\boldsymbol{p}_2 - \boldsymbol{p}_1}{\|\boldsymbol{p}_2 - \boldsymbol{p}_1\|} \tag{2.28}$$

$$\nabla_{\boldsymbol{p}_2} C^{spring}(\boldsymbol{p}_1, \boldsymbol{p}_2) = +\frac{\boldsymbol{p}_2 - \boldsymbol{p}_1}{\|\boldsymbol{p}_2 - \boldsymbol{p}_1\|} \tag{2.29}$$

よって，式よりスケーリング係数 s^{spring} は

$$s^{spring} = -\frac{1}{w_1 + w_2}(\|\boldsymbol{p}_2 - \boldsymbol{p}_1\| - d_0) \tag{2.30}$$

以上より，修正量は次のように求められる．

$$\Delta\boldsymbol{p}_1 = +\frac{w_1}{w_1 + w_2}(\|\boldsymbol{p}_2 - \boldsymbol{p}_1\| - d_0)\frac{\boldsymbol{p}_2 - \boldsymbol{p}_1}{\|\boldsymbol{p}_2 - \boldsymbol{p}_1\|} \tag{2.31}$$

$$\Delta\boldsymbol{p}_2 = -\frac{w_1}{w_1 + w_2}(\|\boldsymbol{p}_2 - \boldsymbol{p}_1\| - d_0)\frac{\boldsymbol{p}_2 - \boldsymbol{p}_1}{\|\boldsymbol{p}_2 - \boldsymbol{p}_1\|} \tag{2.32}$$

2.3.2 PBD の流体への適用

PBD の考え方を流体シミュレーションに導入した PBF では，まず，式 2.8 のうち粘性拡散項 ($\boldsymbol{F}^{visc}$) と外力項 ($\boldsymbol{F}^{ext}$) からのみ力を算出し，それから加速度，速度を計算する．そして，圧力項を計算する代わりに，密度が静密度 ρ_0 と等しくなるように制約をかけて位置を修正するという 2 段階の計算を行う．

密度が静密度 ρ_0 と等しくなるような制約式は，次の通りである．

$$C_i^{fluid}(\boldsymbol{p}_1, ...\boldsymbol{p}_n) = \frac{\rho_1}{\rho_0} - 1 \tag{2.33}$$

密度 ρ_i の計算に SPH 法 (式 2.1) を用い，勾配をとると

$$\nabla_{\boldsymbol{p}_k} C_i^{fluid} = \frac{1}{\rho_0}\begin{cases} \sum_j \nabla_{\boldsymbol{p}_k} W(\boldsymbol{p}_j - \boldsymbol{p}_i, h) & \text{if } k = i \\ -\nabla_{\boldsymbol{p}_k} W(\boldsymbol{p}_j - \boldsymbol{p}_i, h) & \text{if } k = j \end{cases} \tag{2.34}$$

PBF ではスケーリング係数 s_i を解く際に，シミュレーションの安定化のためにユーザが定義可能な緩和パラメータ ε_{pbf} を導入しており，スケーリング係数 s_i は次のようになる．

$$s_i = -\frac{C(\boldsymbol{p}_1, ...\boldsymbol{p}_n)}{\sum_k \|\nabla_{\boldsymbol{p}_k} C_i(\boldsymbol{p}_1, ...\boldsymbol{p}_n)\|^2 + \varepsilon_{pbf}} \tag{2.35}$$

本研究では，緩和パラメータ ε_{pbf} は 1.0×10^{-6} としている．以上より，修正量は

$$\Delta \boldsymbol{p}_i = \frac{1}{\rho_0} \sum_j (s_i + s_j) \nabla W(\boldsymbol{p}_j - \boldsymbol{p}_i, h) \tag{2.36}$$

となる．そして，前述の通り，修正後の位置から速度を更新する．

また，位置ベース法を用いた場合にエネルギーが減衰してしまう問題に対処するため，Macklin ら [4] と同様に渦のエネルギーを補填する．渦のエネルギー $\boldsymbol{F}_i^{vorticity}$ は次のように計算される．

$$\boldsymbol{F}_i^{vorticity} = \epsilon_{vort}(\mathbf{N} \times \boldsymbol{\omega}_i) \tag{2.37}$$

$$\boldsymbol{\omega}_i = \nabla \times \boldsymbol{v} = \sum_j \boldsymbol{v}_{ij} \times \nabla_{\boldsymbol{p}_j} W(\boldsymbol{p}_i - \boldsymbol{p}_j, h) \tag{2.38}$$

ここで，$\mathbf{N} = \frac{\boldsymbol{\eta}}{|\boldsymbol{\eta}|}$，$\boldsymbol{\eta} = \nabla |\boldsymbol{\omega}|_i$ である．ϵ_{vort} は渦度を調整するパラメータであり，乱れ具合いをユーザが定めることができる．

2.4 陰解法による熱計算の安定化

PBF によって粒子位置の計算を安定化させたが，粒子にかかる浮力は粒子温度に依存するため，粒子温度の計算も安定化させる必要がある．温度計算の安定化には陰解法を用いる．

SPH 法における温度の更新 (熱拡散) は，粘性拡散項と同様の計算方法である．各粒子が異なる熱伝導率を持つ場合も考慮し，Hochstettr ら [14] の式を用いる．

$$\frac{dT_p}{dt} = \frac{1}{\rho_p c} \sum_{q \in N} \frac{4 k_p k_q}{k_p + k_q} \frac{m_q}{\rho_q} (T_q - T_p) \nabla^2 W_{visc}(\boldsymbol{r}, h) \tag{2.39}$$

ここで，c は熱容量，k_p，k_q はそれぞれ粒子 p，q の熱伝導率である．従来の研究 [2][3] では，$n+1$ ステップ目の温度 $T^{n+1} = T(t + \Delta t)$ を求める際に陽解法を用いて，n ステップ目の温度 T^n から次の式のように求めていた．

$$T_p^{n+1} = T_p^n + \Delta t \frac{1}{\rho_p c} \sum_{q \in N} \frac{4 k_p k_q}{k_p + k_q} \frac{m_q}{\rho_q} (T_q^n - T_p^n) \nabla^2 W_{visc}(\boldsymbol{r}, h) \tag{2.40}$$

すなわち，$f(T) = \frac{dT}{dt}$ とするとき，次のように表される．

$$T^{n+1} = T^n + \Delta t f(T^n) \tag{2.41}$$

この式を変形すると $\frac{dT^t}{dt} = \frac{T^{n+1} - T^n}{\Delta t}$ となる．$\Delta t \to 0$ の極限をとると微分の定義と同じになるが，Δt が大きい値のときその誤差は大きくなり，結果として不安定になる．T^n ではなく T^{n+1} に対して微分を考えると次のステップで満たすべき条件を求めることになるため，Δt の大きさに依存せずに安定した計算が可能となる．そこで，$\frac{dT^{n+1}}{dt} = \frac{T^{n+1} - T^n}{\Delta t}$ となるよう，次の式のように T^{n+1} を求める．

$$T_p^{n+1} = T_p^n + \Delta t \frac{1}{\rho_p C} \sum_{q \in N} \frac{4 k_p k_q}{k_p + k_q} \frac{m_q}{\rho_q} (T_q^{n+1} - T_p^{n+1}) \nabla^2 W_{visc}(\boldsymbol{r}, h) \tag{2.42}$$

このようにして解く方法は陰解法と呼ばれる．陰解法で解くと，タイムステップ幅 Δt を大きくしても計算が安定するというメリットがある [15]．しかし，未知変数が右辺にも含まれているので，式 2.42 を直接計算はできない．各粒子で式 2.42 が成り立つので，それらを連立させて線形システムとして解くことで解が得られる．線形システムの係数を求めるために式 2.43 を以下のように書き換える．

$$T_p^{n+1} = T_p^n + a_{p1}T_1^{n+1} + a_{p2}T_2^{n+1} + \cdots + a_{pp}T_p^{n+1} + \cdots + a_{pN}T_N^{n+1} \tag{2.43}$$

$$a_{pq} = \begin{cases} \Delta\frac{1}{\rho_p C}\frac{4k_pk_q}{k_p+k_q}\frac{m_q}{\rho_q}\nabla^2 W_{visc}(\boldsymbol{r},h) & \text{if } q \neq p \\ -\Delta\frac{1}{\rho_p C}\sum_{s\neq p}\frac{4k_pk_s}{k_p+k_s}\frac{m_s}{\rho_s}\nabla^2 W_{visc}(\boldsymbol{r},h) & \text{if } q = p \end{cases} \tag{2.44}$$

ここで未知変数 T_p^{n+1}，既知変数 T_p^n，係数 a_{pq} を以下のようにベクトル，行列で表記する．

$$\boldsymbol{T}^{new} = \begin{pmatrix} T_1^{n+1} \\ T_2^{n+1} \\ \vdots \\ T_p^{n+1} \\ \vdots \\ T_N^{n+1} \end{pmatrix}, \boldsymbol{T} = \begin{pmatrix} T_1^{n} \\ T_2^{n} \\ \vdots \\ T_p^{n} \\ \vdots \\ T_N^{n} \end{pmatrix}, \boldsymbol{D} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1q} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2q} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{p1} & a_{p2} & \cdots & a_{pq} & \cdots & a_{pN} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{Nq} & \cdots & a_{NN} \end{pmatrix} \tag{2.45}$$

そうすると式 2.43 は $\boldsymbol{T}^{new} = \boldsymbol{T} + \boldsymbol{D}\boldsymbol{T}^{new}$ と書け，$\boldsymbol{A} = \boldsymbol{I} - \boldsymbol{D}$ とおくと，

$$\boldsymbol{A}\boldsymbol{T}^{new} = \boldsymbol{T} \tag{2.46}$$

となり，$\boldsymbol{A}$ を係数行列とした線形システムとなる．

以上より，$\boldsymbol{T}^{new} = \boldsymbol{A}^{-1}\boldsymbol{T}$ を算出することで，炎粒子同士の熱拡散を計算することができる．陰解法による計算安定化の効果は，結果の章で示す．また，炎の周りの空気との熱のやり取りを簡易的に考慮するため，熱拡散の計算の後，全ての粒子に関して ΔT だけ温度を下げることとする．この温度低下 ΔT は任意のパラメータであり，ユーザが指定することができる．

2.5 視覚的なパラメータ設定

前節までのような手法で安定した炎のシミュレーションを実現させたが，炎の形の制御にはまだ多くの物理パラメータの調整が必要となる．CG や物理学に精通していないユーザでも容易に制御できるように，本節では新しい視覚的なパラメータとして，規模のパラメータ「volume」，広がりのパラメータ「spread」，高さのパラメータ「height」，乱れのパラメータ「turbulence」の 4 種類を提案する．本研究では，各パラメータの満たすべき条件として以下の 3 つを設定する．

- 0.0〜1.0 で正規化されている
- 他のパラメータによる形状変化に影響しない
- パラメータの数値に対して変化が線形

2.5.1　炎の大きさの計測方法

パラメータによる炎の形状変化を定量的に評価するため，まず各物理パラメータの変化による炎の大きさの変化を計測する．”炎の大きさ”の定義のために，粒子の AABB(Axis-Aligned Bounding Box) を用いる．x 座標，y 座標，z 座標それぞれに関して，あるタイムステップの時点での炎粒子の位置座標の最大値と最小値を出し，(最大値-最小値) を AABB 各辺の長さ l_x, l_y, l_z として算出する．乱流の影響で炎は常に揺れており，不規則な変化によって一時的に大きく，あるいは小さくなることがある．この影響を除外するために，算出した長さに外れ値が存在するとして，数十ステップ分の長さを計測し，その中央値をそのシーンの炎の大きさとする．本研究でのシミュレーション設定では，シミュレーションを開始して粒子が噴出し初めてから約 50 ステップ後から形状が安定することから，余裕をみて 71 ステップ目から 100 ステップ目の長さを計測して中央値を求めることとする．また，イレギュラーに飛び出てしまった粒子を除外するため，近傍粒子の最大個数 N_{max} の 10 %以上の個数の粒子が近傍にいない粒子に関しては対象外とする (図 2.8)．

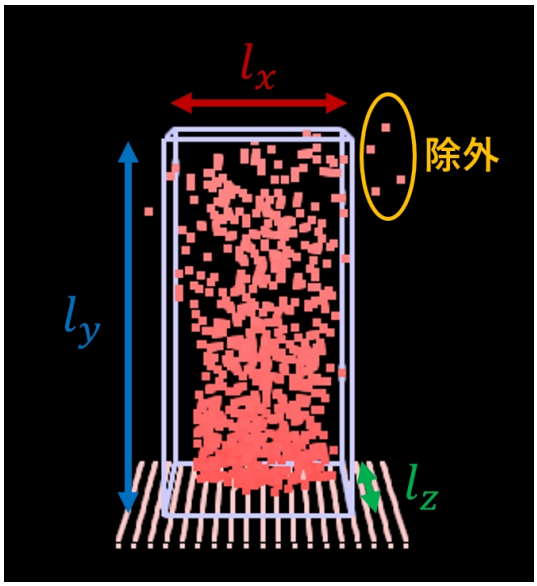


図 2.8: 炎の大きさの計測イメージ

2.5.2　volume パラメータ

図 2.9 のように炎の規模を変えるパラメータとして,「volume」を設定する．

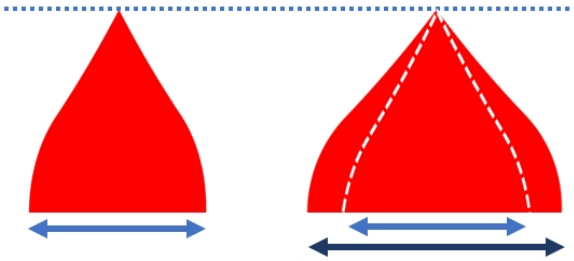


図 2.9: volume パラメータによる変化のイメージ

この「volume」を変更するには，炎粒子の噴出範囲の半径 R を変更する．任意の最小半径 $R_{\min}$ と最大半径 $R_{\max}$ を定め，以下の式で R を決定する．

$$R = R_{\min} + \mathrm{volume}(R_{\max} - R_{\min}) \tag{2.47}$$

また，噴出時の粒子の密度を維持するため，1 ステップあたりの噴出個数 n^{gene} を噴出範囲の面積に比例して決定する．噴出範囲の最小半径 $R_{\min}$ のときの 1 ステップあたりの噴出個数を $n^{gene}_{\min}$ とすると，volume によって次のように変更される．

$$n^{gene} = (\frac{R}{R_{\min}})^2 n_{\min} \tag{2.48}$$

任意のパラメータ $R_{\min}$, $R_{\max}$, $n_{\min}$ は，本研究では $R_{\min} = 0.5$, $R_{\max} = 0.75$, $n^{gene}_{\min} = 30$ とする．

2.5.3 spread パラメータと height パラメータ

図 2.10 のように炎の広がりを spread，図 2.11 のように炎の高さを height として，それぞれ変更するパラメータを設定するため，これらの要素を変更し得る物理パラメータを変え，それぞれ炎の大きさにどう影響するかを実験した．

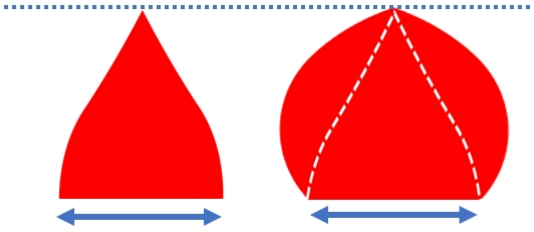


図 2.10: spread パラメータによる変化のイメージ

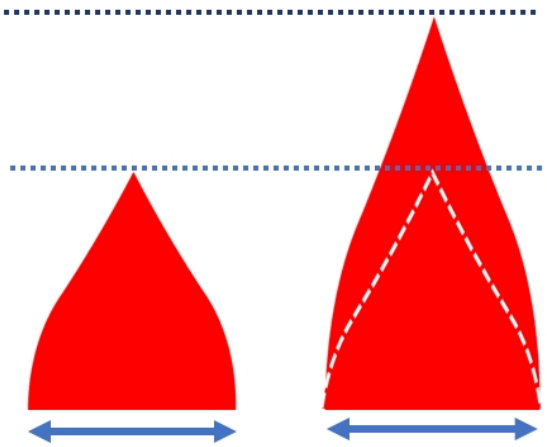


図 2.11: height パラメータによる変化のイメージ

温度低下 ΔT，噴出個数 n^{gene}，重力にかかる係数 α，浮力にかかる係数 β，熱伝導率 k，動粘性係数 ν をそれぞれ変えた結果を示していく．以降のすべての表において l_x, l_y, l_z の数値は，シミュレーション空間の大きさを 1 としたときの値である．

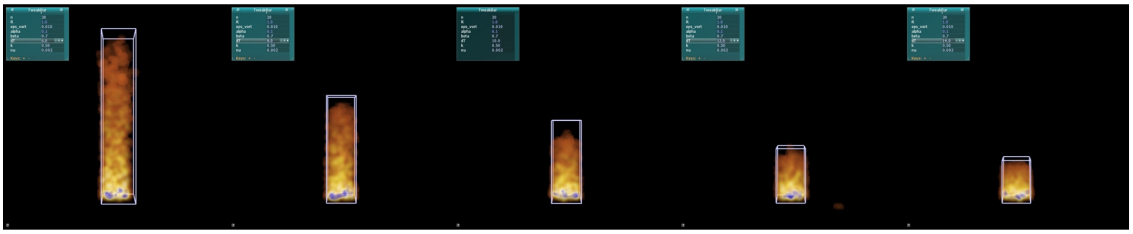
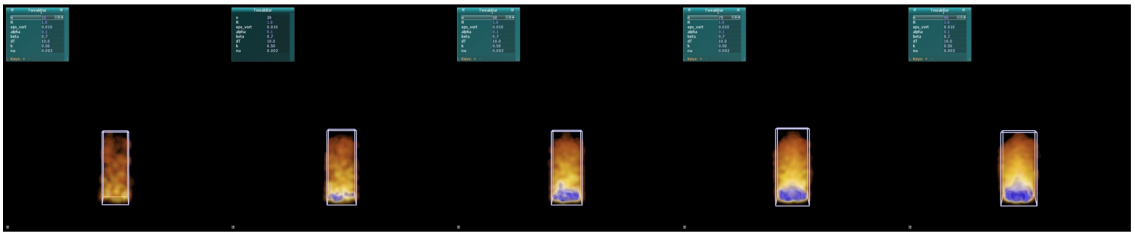


図 2.12: ΔT を変えた結果

表 2.1: ΔT による大きさの変化

ΔT	l_x	l_y	l_z
6	0.114463	0.611416	0.110703
8	0.104542	0.370870	0.102962
10	0.102855	0.256397	0.103192
12	0.101114	0.196383	0.098870
14	0.100431	0.155409	0.101777

ΔT を変えた結果が，図 2.12 と表 2.1 である．横幅 (l_x, l_z) はほぼ変わることなく高さ (l_y) が変わるため，高さのパラメータ「height」に利用できることがわかった．ΔT が大きくなるほど，高さは低くなる．



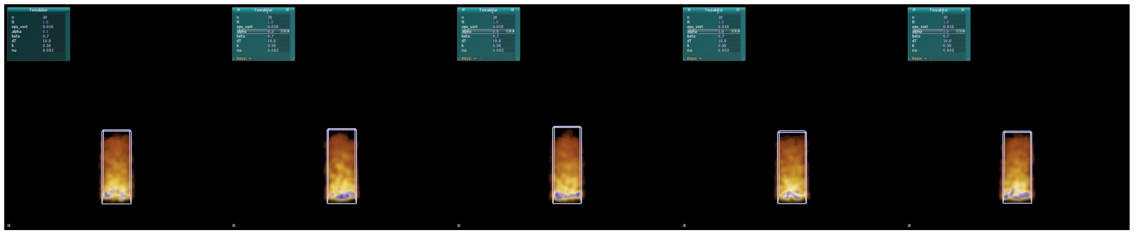
$n = 10$ $n = 30$ $n = 50$ $n = 70$ $n = 90$

図 2.13: n^{gene} を変えた結果

表 2.2: n^{gene} による大きさの変化

n^{gene}	l_x	l_y	l_z
10	0.097194	0.245550	0.094363
30	0.100242	0.261358	0.100541
50	0.104700	0.262841	0.101708
70	0.112502	0.265863	0.113013
90	0.126590	0.265744	0.129211

噴出個数 n^{gene} を変えた結果が，図 2.13 と表 2.2 である．密度に違いは出たものの，形状が広がるような変化は見られなかった．

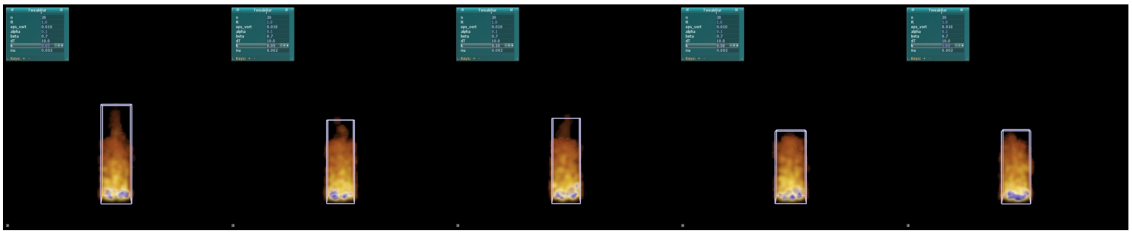


$\alpha = 0.1$ $\alpha = 0.3$ $\alpha = 0.5$ $\alpha = 1.0$ $\alpha = 2.0$

図 2.14: α を変えた結果

表 2.3: α による大きさの変化

α	l_x	l_y	l_z
0.1	0.100806	0.257250	0.103314
0.3	0.102055	0.258629	0.100981
0.5	0.104852	0.257638	0.100484
1.0	0.103347	0.256025	0.102934
2.0	0.101322	0.254704	0.100890

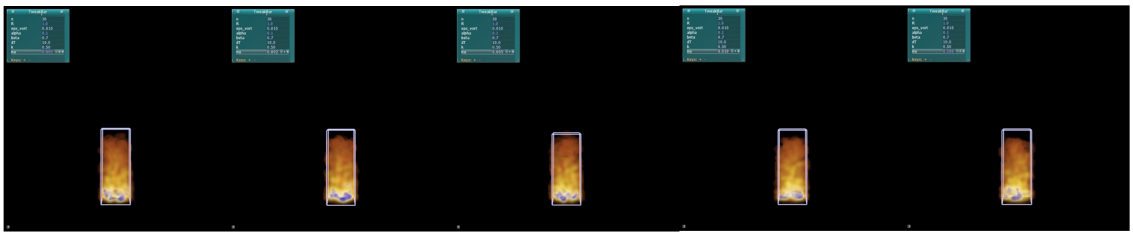


$k = 0.01$ $k = 0.05$ $k = 0.1$ $k = 0.5$ $k = 1.0$

図 2.15: k を変えた結果

表 2.4: k による大きさの変化

k	l_x	l_y	l_z
0.01	0.103977	0.297014	0.100306
0.05	0.102966	0.296130	0.104856
0.10	0.102451	0.283563	0.103375
0.50	0.102923	0.263093	0.102438
1.00	0.100229	0.250371	0.102388



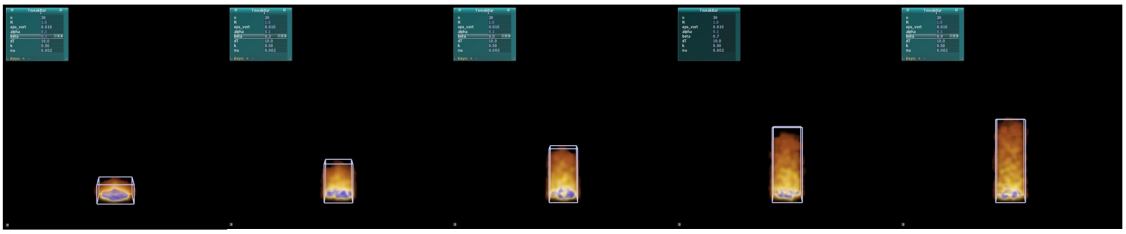
$\nu = 0.001$ $\nu = 0.002$ $\nu = 0.005$ $\nu = 0.01$ $\nu = 0.1$

図 2.16: ν を変えた結果

表 2.5: ν による大きさの変化

ν	l_x	l_y	l_z
0.001	0.099929	0.256235	0.101043
0.002	0.098775	0.260891	0.103361
0.005	0.101008	0.257871	0.103286
0.010	0.101181	0.258901	0.100425
0.100	0.103905	0.260581	0.100953

重力にかかる係数 α，熱伝導率 k，動粘性係数 ν を変えた結果が，それぞれ図 2.14 と表 2.3，図 2.15 と表 2.4，図 2.16 と表 2.5 である．幅 (l_x, l_y, l_z) や形状，ともにあまり変化が見られなかった．k については高さの変化はあったが，ΔT と比較して小さな変化でしかなかったため，高さパラメータの制御には用いない．



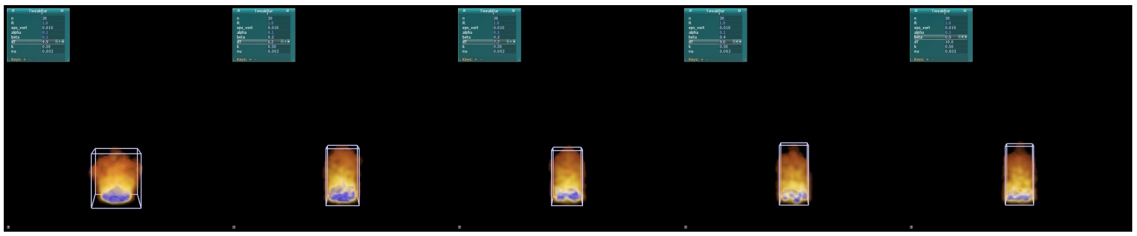
$\beta = 0.1$ $\beta = 0.3$ $\beta = 0.5$ $\beta = 0.7$ $\beta = 0.9$

図 2.17: β を変えた結果

表 2.6: β による大きさの変化

β	l_x	l_y	l_z
0.1	0.128638	0.079616	0.129227
0.3	0.102834	0.138834	0.102883
0.5	0.101327	0.197974	0.100863
0.7	0.098590	0.254465	0.100016
0.9	0.102755	0.319991	0.101443

熱による浮力に関わる係数 β を変えた結果が，図 2.17 と表 2.6 である．β を小さくしたときに，やや横幅 (l_x, l_z) が長くなり形状も変化したが，高さが大きく変わってしまった．そこで，高さの調節に有効である ΔT によって高さを等しく調節した場合に，横幅 (l_x, l_z) がどう変化するかを実験した．



$\beta = 0.1, \Delta T = 4.9$ $\beta = 0.2, \Delta T = 6.2$ $\beta = 0.3, \Delta T = 7.7$ $\beta = 0.4, \Delta T = 9.0$ $\beta = 0.5, \Delta T = 10.0$

図 2.18: β と ΔT を変えた場合

表 2.7: β と ΔT による大きさの変化

β	ΔT	l_x	l_y	l_z
0.1	4.9	0.172562	0.198350	0.170298
0.2	6.2	0.112560	0.197833	0.114681
0.3	7.7	0.108732	0.196108	0.105999
0.4	9.0	0.100783	0.198904	0.102024
0.5	10.0	0.105321	0.198096	0.100858

β と ΔT を変え，高さが等しくなるよう調節した結果が，図 2.18 と表 2.7 である．spread パラメータを変化させたとき，横幅 (l_x, l_z) が線形に広がるように β を定めるため，β と横幅 (l_x, l_z) の関係を関数化する．表 2.7 の実験結果から，$\beta = [0.1, 0.4]$ の範囲において，β と $\frac{l_x+l_z}{2}$ の関係は次の二次関数で近似できる (図 2.19)．

$$\frac{l_x + l_z}{2} = 0.74\beta^2 - 0.6\beta + 0.22 \tag{2.49}$$

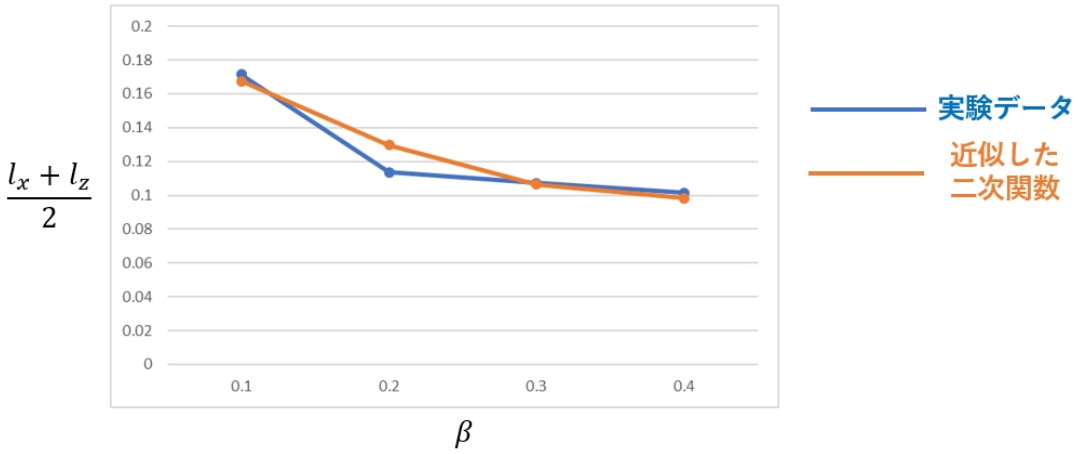


図 2.19: 表 2.7 の結果と近似した二次関数 2.49 のグラフ

所望の横幅を l_{xz}^{ideal} としたとき，式 2.49 の逆関数 (式 2.50) をとることで横幅が $l_x^{ideal}z$ に近づく β を求めることができる．

$$\beta = \frac{0.6 - \sqrt{2.96 l_{xz}^{ideal} - 0.2912}}{1.48} \tag{2.50}$$

表 2.7 の結果より，l_{xz}^{ideal} のとり得る値は $l_{xz}^{ideal} = [0.1, 0.17]$ であるから，l_{xz}^{ideal} は spread パラメータに応じて次の式で定める．

$$l_{xz}^{ideal} = 0.1 + 0.07\text{spread} \tag{2.51}$$

また，表 2.7 より，β を変えたときに高さを維持するための ΔT は，次の式で近似できる．

$$\Delta T = 3.66667 + 13.3333\beta \tag{2.52}$$

更に，図 2.12 と表 2.1 の結果より，ΔT は height パラメータにも応じて変更させる．height パラメータによる ΔT の変域を τ として，デフォルトの高さを height=0.5 とすると，2.52 と合わせて ΔT は次の式で定まる．

$$\Delta T = 3.66667 + 13.3333\beta + \frac{\tau}{2} - \text{height}\tau \tag{2.53}$$

2.5.4 turbulence パラメータ

図 2.20 のように炎の乱れ具合いを変えるパラメータとして，「turbulence」を設定する．

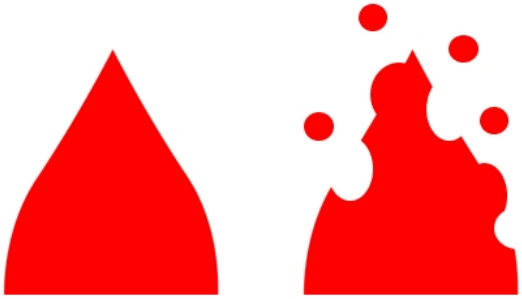
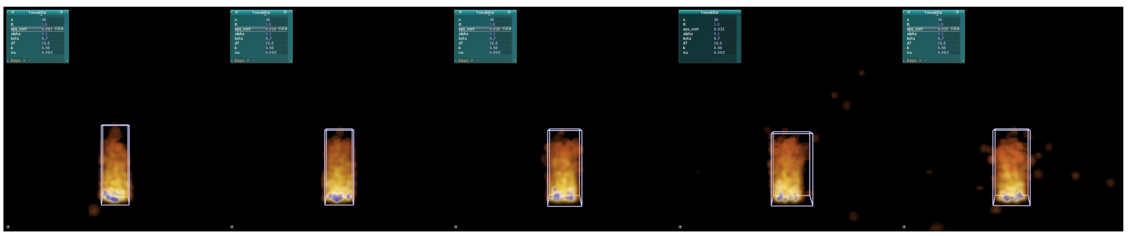


図 2.20: turbulence パラメータによる変化のイメージ

この turbulence の調節には，2.3 節で述べた渦度 ϵ_{vort} を用いる．渦度 ϵ_{vort} によって炎の見た目がどう変化するかを実験した．



$\epsilon_{vort} = 0.001$　$\epsilon_{vort} = 0.01$　$\epsilon_{vort} = 0.02$　$\epsilon_{vort} = 0.022$　$\epsilon_{vort} = 0.025$

図 2.21: ϵ_{vort} を変えた結果

表 2.8: ϵ_{vort} による大きさの変化

ϵ_{vort}	l_x	l_y	l_z
0.001	0.980750	0.270534	0.100309
0.010	0.100955	0.258024	0.100588
0.020	0.117219	0.245404	0.120210
0.022	0.120670	0.245040	0.122052
0.025	0.129758	0.242432	0.135895

ϵ_{vort} を変えた結果が，図 2.21 と表 2.8 である．0.01 以下は見た目や大きさにほぼ変化がなく，0.02 より大きいと粒子が散らばりすぎてしまい炎らしい見た目が失われてしまうため，ϵ_{vort} は $[0.01, 0.02]$ とする．よって，ϵ_{vort} は turbulence パラメータを用いて次の式で定める．

$$\epsilon_{vort} = 0.01 + 0.01\text{turbulence} \tag{2.54}$$

2.6　レンダリング

本研究では炎のレンダリング方法としてテクスチャベースボリュームレンダリング [16] を用いる．シミュレーション空間を格子分割し，粒子のある各セル毎に粒子の個数や温度に応じた色の数値を入れ，これを 3D テクスチャとして半透明スライスを用いてボリュームレンダリングを行う．炎は，温度が高いほど青色に近く，温度が低いほど赤に近く，温度によって色合いが変わる現象である．これを表すために，本研究では，図 2.22 のテクスチャを用いてセルに格納する色を決定した．

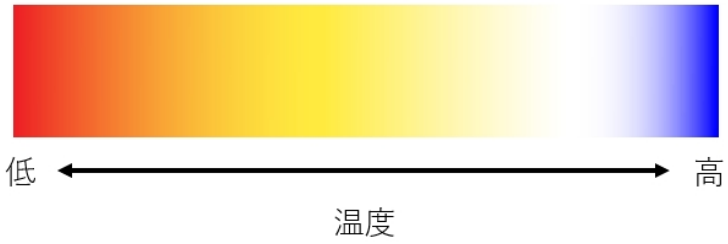


図 2.22: 炎色テクスチャ

第3章　結果

本章では，パラメータを変えたいくつかのシーンでのシミュレーション結果を示す．表3.1 に開発・実行環境を示す．

表 3.1: 開発・実行環境

CPU	Intel Core i7-4720HQ 2.60GHz
Main Memory	16GB
OS	Microsoft Windows 10
開発言語	C++
グラフィックス API	OpenGL

volume，spread，height，turbulence の 4 つのパラメータによって変わらないパラメータとその数値を表 3.2 に示す．

表 3.2: 変更しないパラメータ

重力加速度 g	9.8[m/s^2]
粒子半径 pSize	0.01[m]
粒子質量 pm	0.001[kg]
熱容量 c	1[J/K]
近傍粒子最大数 $N_{\max}$	30
静密度 ρ_0	1.0[kg/m^3]
粒子初期温度 T_0	1000[K]
環境温度 t_{amb}	300[K]
タイムステップ幅 Δt	0.001[s]
ガス定数 G	3.0
熱伝導率 k	0.5[W/m・K]
動粘性係数 ν	0.002[m^2/s]

各シミュレーション結果の基準として，デフォルトの設定 (volume=0.0，height=0.5，spread=0.0，turbulence=0.0) での結果を図 3.1〜3.6 と表 3.3 に示す．シミュレーション速度は 9.62fps であった．

図 3.1: 3 フレーム目

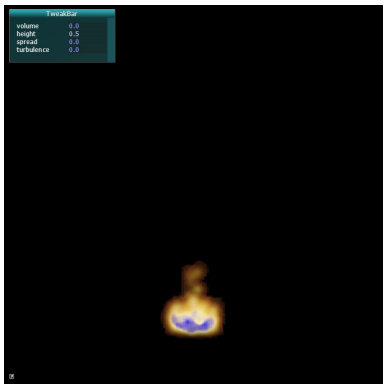


図 3.2: 20 フレーム目

図 3.3: 40 フレーム目

図 3.4: 60 フレーム目

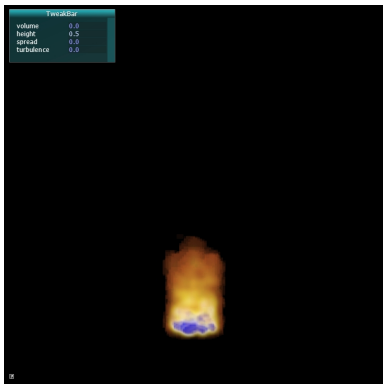


図 3.5: 80 フレーム目

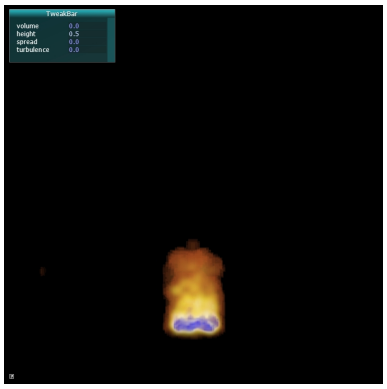


図 3.6: 100 フレーム目

表 3.3: デフォルト設定での大きさ

l_x	l_y	l_z
0.107614	0.197851	0.104180

3.1　陰解法による熱計算の安定化

熱計算に陽解法を用いた場合のシミュレーション結果は図 3.7〜3.9 のようになった．

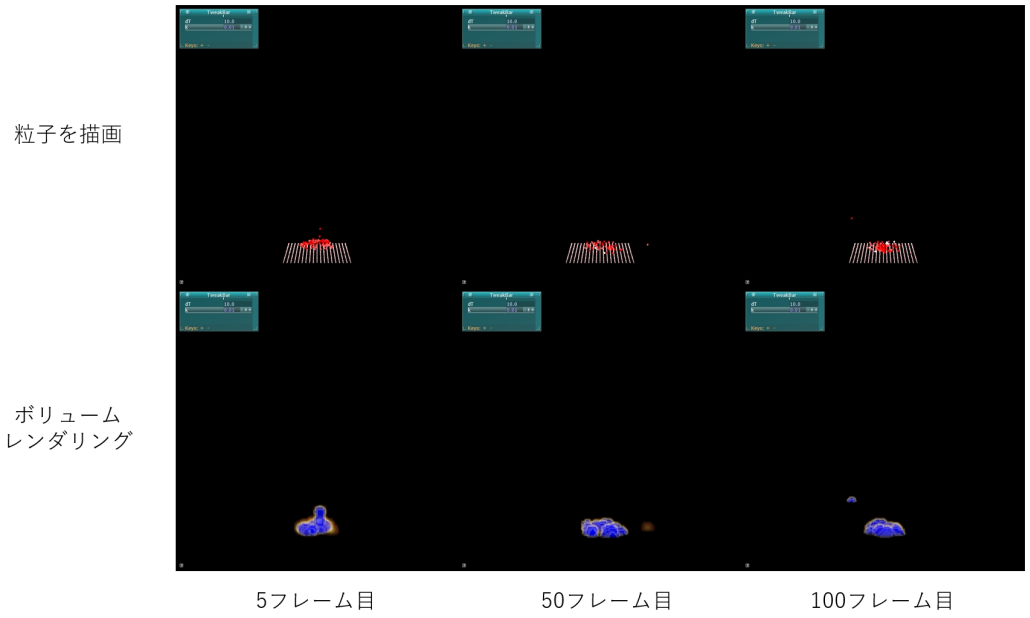


図 3.7: $k = 0.01$

図 3.8: $k = 0.5$

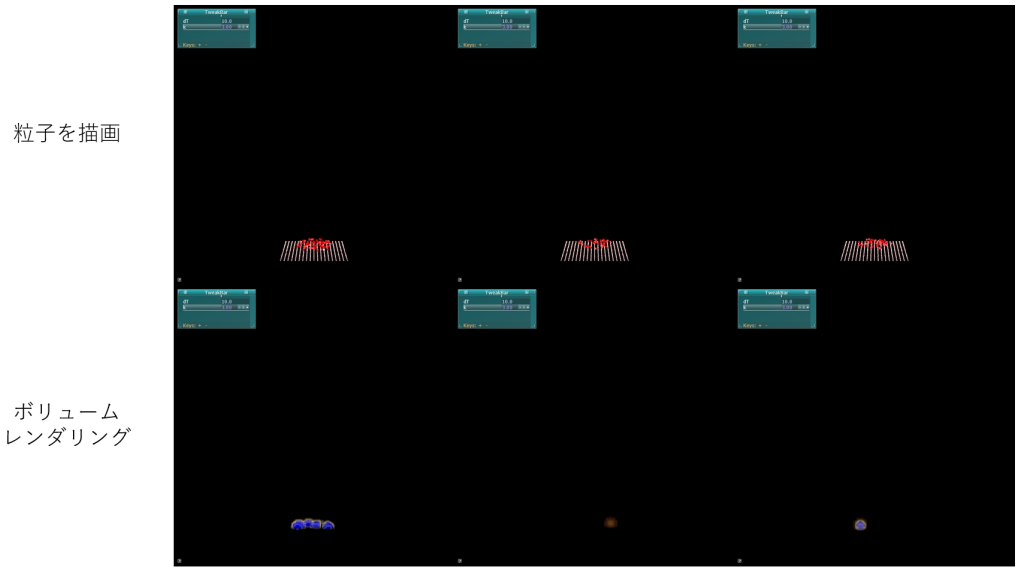


図 3.9: $k = 3$

同じ熱伝導率で，熱計算に陰解法を用いた場合のシミュレーション結果は図 3.10〜3.12 のようになった．

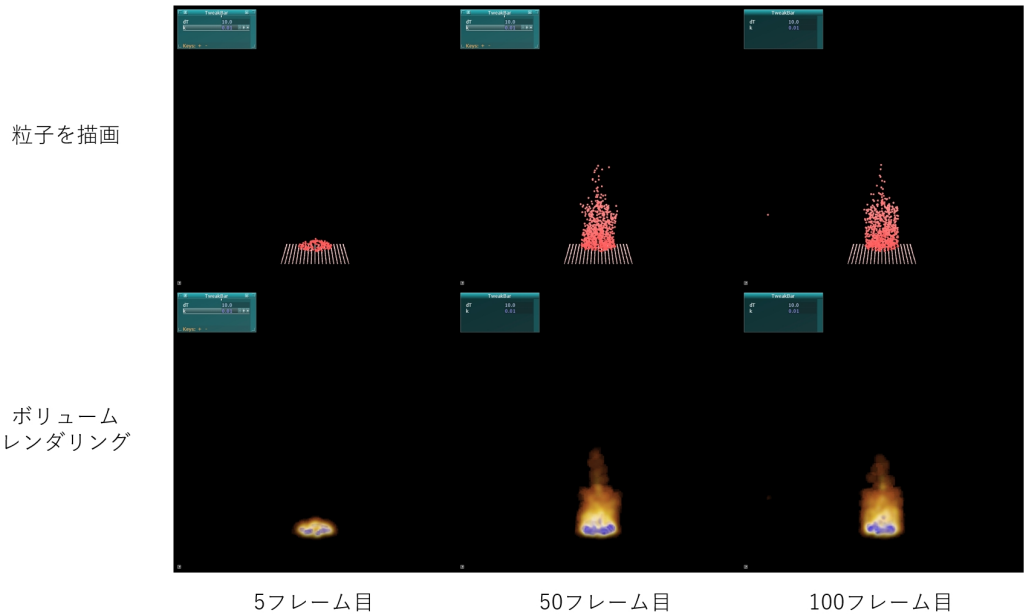


図 3.10: $k = 0.01$

図 3.11: $k = 0.5$

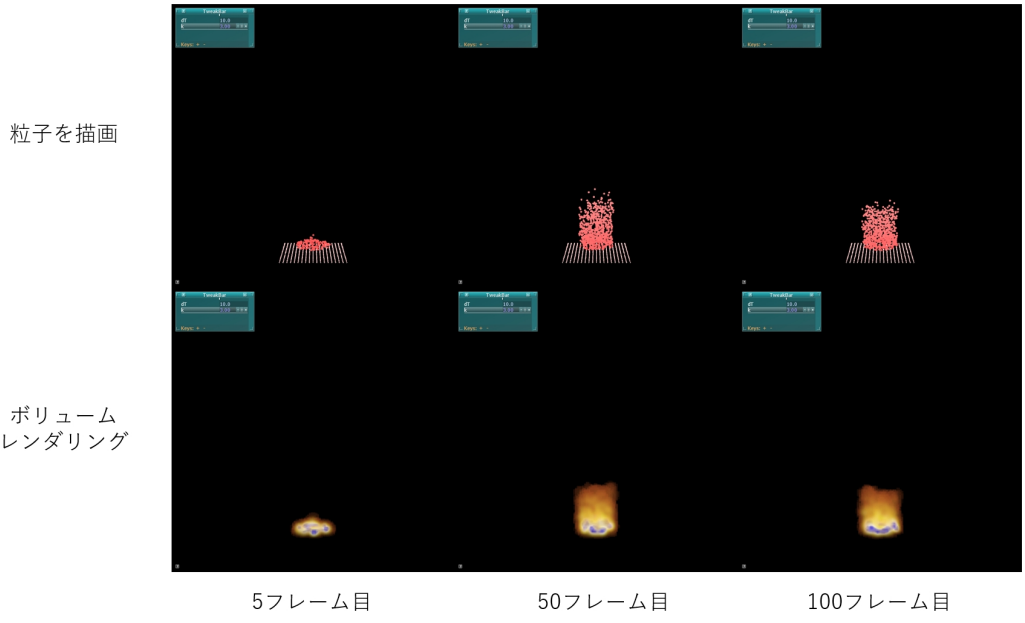


図 3.12: $k = 3$

粒子の色は温度が高いほど赤く，低いほど白く描画している．陽解法では，どの熱伝導率の場合でも粒子温度が安定せず (真っ赤な粒子から真っ白な粒子まで，粒子温度が様々である)，粒子がすぐに消えてしまった．粒子を温度によって消去せずに描画し続けた場合も，図 3.13，図 3.14 のようになり，安定したシミュレーションができなかった．それに対し，陰解法ではパラメータを変更しても安定して炎らしいシミュレーション結果となったことから，熱計算の安定化に陰解法が有用であることがわかった．

図 3.13: $k = 0.01$ / 20 フレーム目

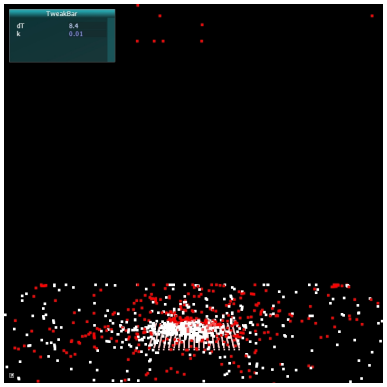


図 3.14: $k = 0.01$ / 50 フレーム目

3.2 volume による変化

volume パラメータを変えたシミュレーション結果を図 3.15〜図 3.20 と表 3.4，図 3.21 に示す．

図 3.15: volume=0.0 / 9.62fps

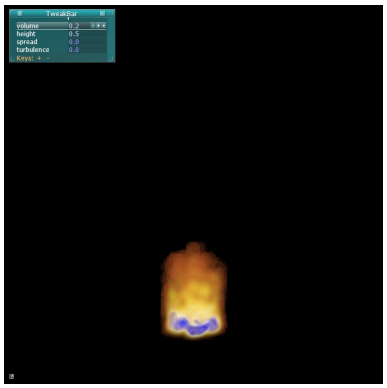


図 3.16: volume=0.2 / 8.37fps

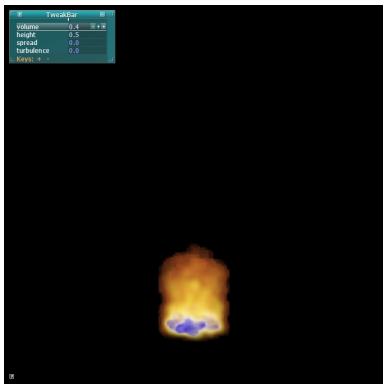


図 3.17: volume=0.4 / 7.35fps

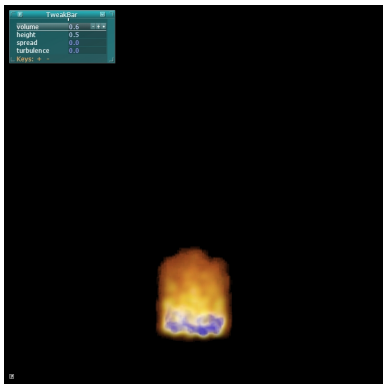


図 3.18: volume=0.6 / 6.54fps

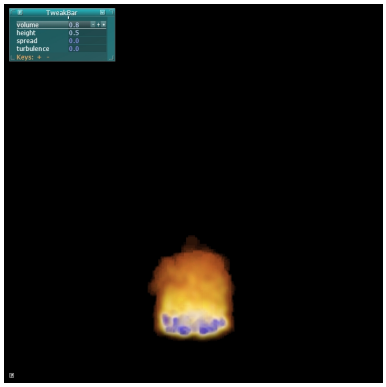


図 3.19: volume=0.8 / 5.45fps

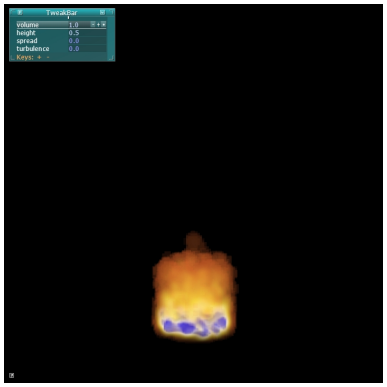


図 3.20: volume=1.0 / 4.94fps

表 3.4 と図 3.21 において l_x と l_z を平均した $\frac{l_x+l_z}{2}$ を炎の横幅としている．

表 3.4: volume パラメータによる大きさの変化

volume	l_x	l_y	l_z	横幅
0.0	0.107046	0.195630	0.102644	0.1048450
0.2	0.113803	0.195237	0.116331	0.1150670
0.4	0.124912	0.194423	0.124635	0.1247735
0.6	0.134089	0.199484	0.138178	0.1361335
0.8	0.147457	0.196864	0.145403	0.1464300
1.0	0.154873	0.196653	0.154076	0.1544745

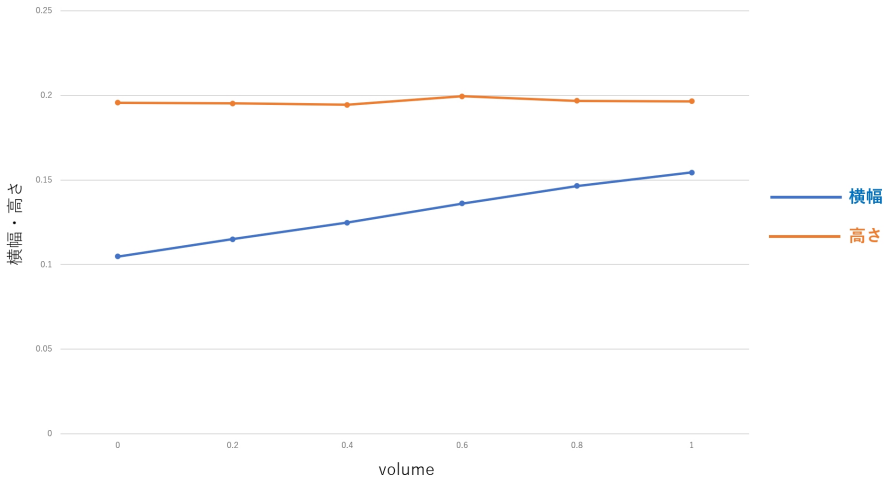


図 3.21: volume パラメータによる横幅の変化

縦幅 l_y や視覚的な炎の密度はほぼ変わらず，volume パラメータに応じて横幅が線形に変化している．それに対して，高さはほぼ変わらないことから，volume パラメータは高さに影響を与えず，炎の規模のみを変えられることがわかる．

3.3 height による変化

height パラメータを変えたシミュレーション結果を図 3.22〜図 3.27 と表 3.5，図 3.28 に示す．

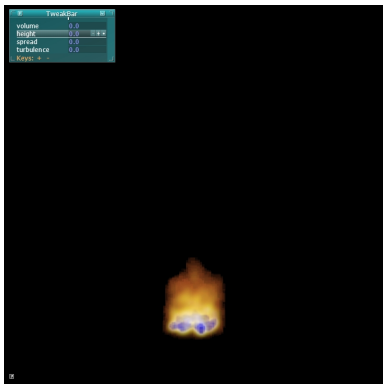


図 3.22: height=0.0 / 10.86fps

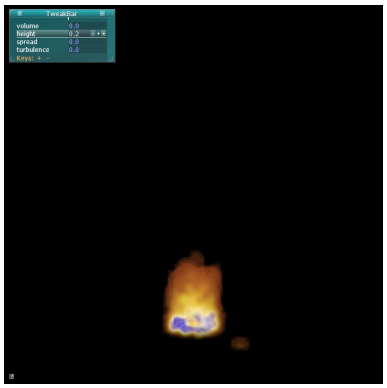


図 3.23: height=0.2 / 10.37fps

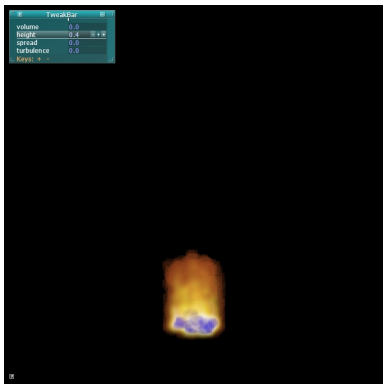


図 3.24: height=0.4 / 9.99fps

図 3.25: height=0.6 / 9.36fps

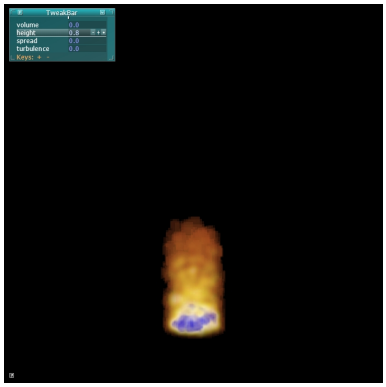


図 3.26: height=0.8 / 8.84fps

図 3.27: height=1.0 / 8.07fps

表 3.5 と図 3.28 において l_x と l_z を平均した $\frac{l_x+l_z}{2}$ を炎の横幅としている．

表 3.5: height パラメータによる大きさの変化

height	l_x	l_y	l_z	横幅
0.0	0.100903	0.151949	0.100420	0.100662
0.2	0.101992	0.162470	0.101158	0.101575
0.4	0.104729	0.190484	0.103182	0.103956
0.6	0.104375	0.212547	0.101276	0.102826
0.8	0.105495	0.245356	0.106980	0.062380
1.0	0.111498	0.298812	0.109442	0.110470

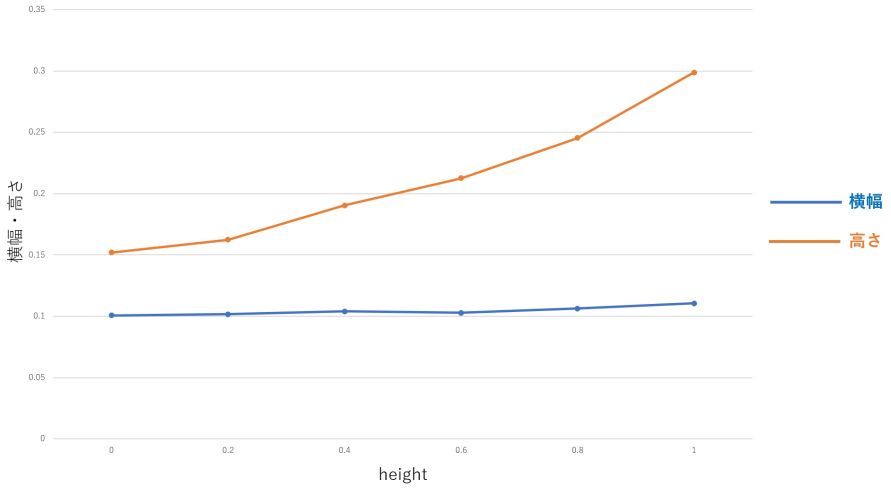


図 3.28: height パラメータによる高さ l_y の変化

height パラメータに応じて高さ l_y がおおよそ線形に変化しているが，粒子の高さと粒子温度が正確に比例しているわけではないため，温度による制御では高さにぶれがある．また，横幅はほぼ変わらないことから，height パラメータは炎の横幅に影響を及ぼさずに高さのみを変えられることがわかる．

3.4 spread による変化

spread パラメータを変えたシミュレーション結果を図 3.29〜図 3.34 と表 3.6，図 3.35 に示す.

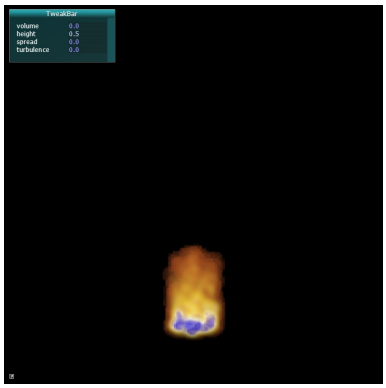


図 3.29: spread=0.0 / 9.62fps

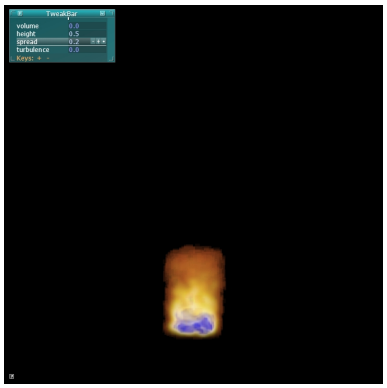


図 3.30: spread=0.2 / 8.54fps

図 3.31: spread=0.4 / 7.64fps

図 3.32: spread=0.6 / 6.83fps

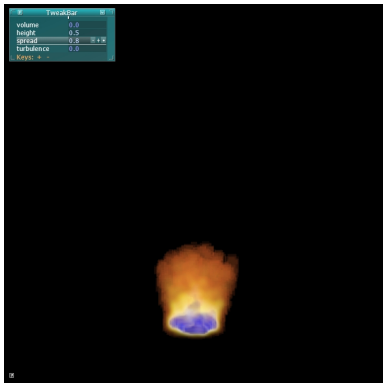


図 3.33: spread=0.8 / 6.25fps

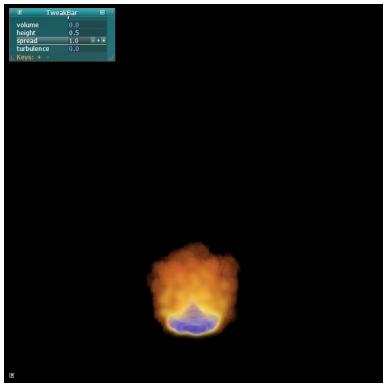


図 3.34: spread=1.0 / 5.91fps

表 3.6 と図 3.35 において横幅 $\frac{l_x+i_z}{2}$ から噴出直径 $2R$ を引いた値を広がりとしている.

表 3.6: spread パラメータによる大きさの変化

spread	l_x	l_y	l_z	広がり
0.0	0.107709	0.192876	0.102055	0.004882
0.2	0.108181	0.202638	0.101598	0.0048895
0.4	0.110004	0.199965	0.111859	0.0109315
0.6	0.125970	0.192925	0.131328	0.028649
0.8	0.158122	0.193343	0.151473	0.0547975
1.0	0.175821	0.204944	0.170919	0.07337

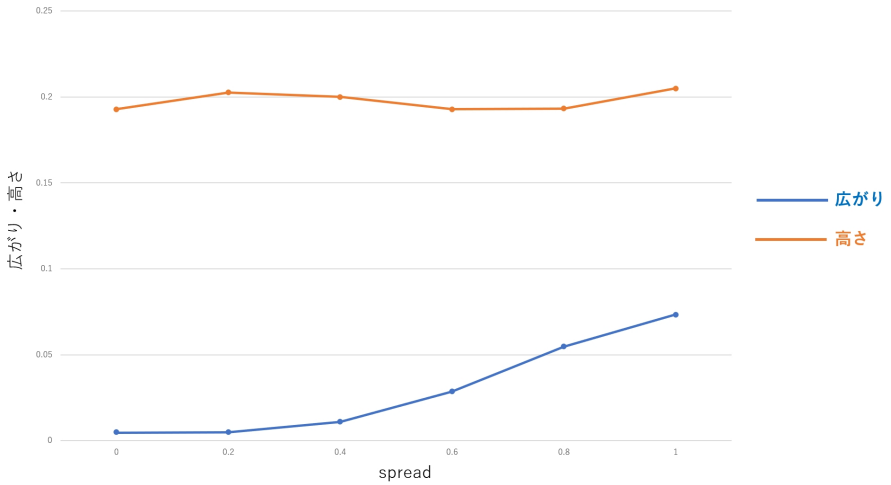


図 3.35: spread パラメータによる広がりの変化

[0.4, 1.0] の範囲では spread パラメータに応じて広がりがおおよそ線形に変化したが，[0.0, 0.4] の範囲では線形にならなかった．線形にならなかった原因として，β と横幅の関係の近似 (式 2.49，図 2.19) が不十分であることが考えられる．また，β による変化を ΔT で調整するという二段階で調整しているため，高さにぶれが出てしまっているが，横幅に比べて変化は小さい．

3.5 turbulence による変化

turbulence パラメータを変えたシミュレーション結果を図 3.36〜図 3.41 に示す.

図 3.36: volume=0.0 / 9.62fps

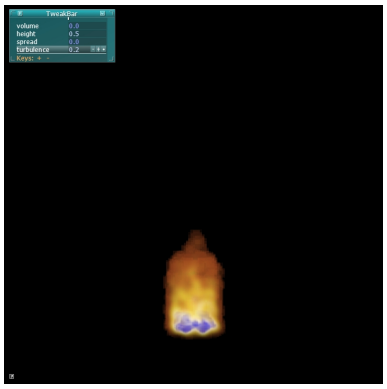


図 3.37: turbulence=0.2 / 9.53fps

図 3.38: volume=0.4 / 9.45fps

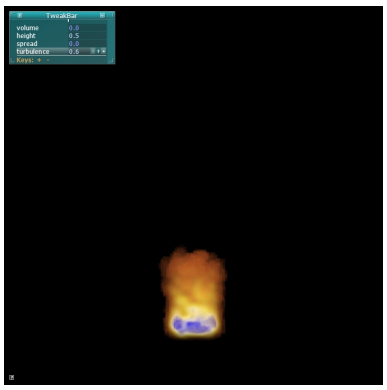


図 3.39: turbulence=0.6 / 9.60fps

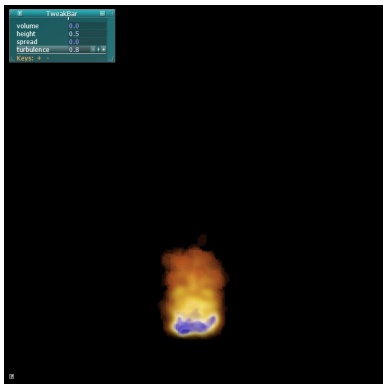


図 3.40: volume=0.8 / 9.75fps

図 3.41: turbulence=1.0 / 9.74fps

画像では，turbulance パラメータが大きいほど粒子が散らばっていることしか確認できないが，動画では粒子の乱れが大きくなることが確認できた.

3.6　パラメータの組み合わせ

ユーザが所望する様々な形状の炎を再現できることを示すため，ユーザが思い描くイメージ通りに 4 種類のパラメータを変えたシミュレーション結果を図 3.42〜図 3.45 に示す.

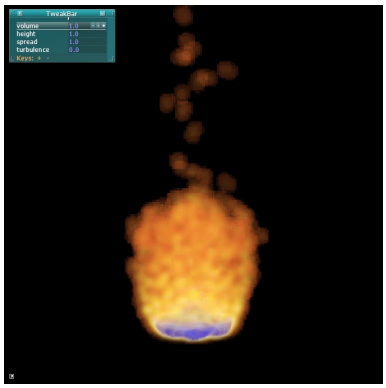


図 3.42: 横幅も縦幅も大きく，広がる炎

図 3.43: 中規模で縦に細長い炎

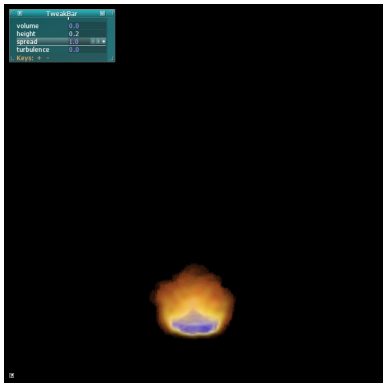


図 3.44: 小さく丸みのある炎

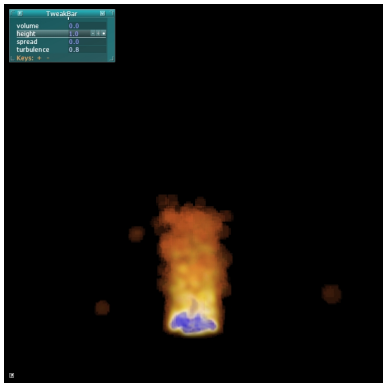


図 3.45: 細長く，乱れた炎

第4章　考察

前章の結果より，炎の規模は volume パラメータ，高さは height パラメータ，広がりは spread パラメータ，乱れは turbulence パラメータでそれぞれ変更できることが確認できた．パラメータ設定の際に条件とした，他の要素に影響しない，変化が線形という点もある程度満たされており，ユーザにとって使いやすいパラメータを設計できたといえる．これらのパラメータは物理的な現象の変化と 1 対 1 に対応するものではないが，例えば，spread パラメータについては浮力と温度低下を大きくしたことで広がりを大きくすることができたが，これは周囲の環境温度を下げたことと同等のパラメータ変更であると考えることができる．

また，特に volume パラメータによる結果からわかる通り，シミュレーション速度は粒子の個数によって変わる．本研究のデフォルトの設定では 10fps 近い速度でのシミュレーションが CPU でできているため，GPU を利用して更に高速化すれば，リアルタイムでのシミュレーションも期待できる．本研究ではリアルタイムでのシミュレーションは実現できなかったものの，パラメータを変更してから形状が変化するまでのタイムラグは少ないため，ユーザが炎の形状を確認しながらパラメータを調節することが可能である．しかし，炎の規模の拡大やレンダリングの精細化のためにも，GPU を用いた高速なシミュレーションが望ましい．

今後，更に厳密な形状制御のため，フィードバック制御を用いるのも有用であると考えられる．height パラメータや spread パラメータは，パラメータの数値とシミュレーション結果の炎の大きさが厳密に線形に変化しているとはいえない．そこで，シミュレーション結果の炎の大きさから，ΔT や β を変更するフィードバック制御を行うことで，ユーザが望む大きさにより近づけることができる．

第5章　結論

本論文では，位置ベース法を用いた安定した炎のシミュレーション手法と，視覚的にわかりやすいパラメータを提案した．従来のSPH法を用いた炎のシミュレーションを，PBFによって位置計算を安定化し，陰解法によって熱計算を安定化したことで，パラメータを変更しても不安定な挙動が起きなくなった．更に，物理学やCGに精通していないユーザも簡単に炎の形状が制御できるよう，規模・高さ・広がり・乱れの4つの視覚的なパラメータを提案した．提案したパラメータの算出式をシミュレーション実験した結果から求めることで，1つのパラメータを変えたときに他の形状要素に影響せず，例えば，高さを変えると幅まで変わってしまうといった制御しにくいものにならず，ユーザにとって使いやすいパラメータを設計した．また，パラメータの数値と形状で変化する結果がほぼ線形となるように設計することで，より直感的な制御を可能とした．

しかし，パラメータの数値と炎の大きさが正確には線形に変化していないパラメータもあったり，シミュレーションがリアルタイムでなかったりといった問題点も残されている．今後の研究としては，フィードバック制御によるパラメータの変更や，GPUによる高速化が挙げられる．

謝辞

はじめに，本研究を進めるにあたって終始丁寧なご指導を頂いた筑波大学図書館情報メディア系の藤澤誠助教に心より感謝申し上げます．また，合同ゼミにおいて多くのご意見・ご助言を賜りました筑波大学図書館情報メディア系の三河正彦准教授に深く御礼申し上げるとともに，物理ベースコンピュータグラフィクス研究室のメンバー，ソーシャルロボット研究室のメンバーに感謝致します．

参考文献

[1] Jos Stam and Eugene Fiume. Depicting fire and other gaseous phenomena using diffusion processes. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, pp.129–136, 1995.

[2] 稲垣 智, 井村 誠孝, 池田 聖, 眞鍋 佳嗣, and 千原 國宏. 粒子ベース流体シミュレーションを用いた炎のリアルタイムレンダリング. **システム制御情報学会　研究発表講演会講演論文集**, SCI09:pp.553–553, 2009.

[3] 佐々木 浩幸. 燃焼過程を考慮した炎のシミュレーション. **筑波大学卒業論文**, 2018.

[4] Miles Macklin and Matthias Müller. Position based fluids. *ACM Transactions on Graphics*, Vol.32(No.4):pp.1–12, 2013.

[5] Hector Barreiro, Ignacio Garcia-Fernandez, Ivan Alduan, and Miguel A. Otaduy. Conformation constraints for efficient viscoelastic fluid simulation. *ACM Transactions on Graphics*, Vol.36(No.6):pp.221:1–11, 2017.

[6] Duc Quang Nguyen, Ronald Fedkiw, and Henrik Wann Jensen. Physically based modeling and animation of fire. *ACM Transactions on Graphics*, Vol.21(No.3):pp.721–728, 2002.

[7] Yubo Zhang, Carlos D. Correa, and Kwan-Liu Ma. Graph-based fire synthesis. In *Proceedings of Eurographics / ACM SIGGRAPH Symposium on Computer Animation*, pp.187–194, 2011.

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

[9] Jan Bender, Matthias Müller, Miguel A. Otaduy, Matthias Teschner, and Miles Macklin. A survey on position-based simulation methods in computer graphics. *Comput. Graph. Forum*, Vol.33(No.6):pp.228–251, 2014.

[10] Hagit Schechter and Robert Bridson. Ghost sph for animating water. *ACM Transactions on Graphics*, Vol.31(No.4):pp.61:1–8, 2012.

[11] Ronald Fedkiw, Jos Stam, and Henrik Wann Jensen. Visual simulation of smoke. *ACM Transactions on Graphics*, Vol.20(No.3):pp.15–22, 2001.

[12] Matthias Müller, David Charypar, and Markus Gross. Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA '03, pp.154–159, 2003.

[13] Nadir Akinci, Markus Ihmsen, Gizem Akinci, Barbara Solenthaler, and Matthias Teschner. Versatile rigid-fluid coupling for incompressible sph. *ACM Transactions on Graphics*, Vol.31(No.4):pp.62:1–62:8, 2012.

[14] Hendrik Hochstetter and Andreas Kolb. Evaporation and condensation of sph-based fluids. In *Proceedings of Eurographics / ACM SIGGRAPH Symposium on Computer Animation*, pp.3:1–7, 2017.

[15] 越塚誠一, 柴田和也, and 室谷浩平. **粒子法入門**. 丸善出版, 2014.

[16] Brian Cabral, Nancy Cam, and Jim Foran. Accelerated volume rendering and tomographic reconstruction using texture mapping hardware. In *Proceedings of VVS '94: Proceedings of the 1994 symposium on Volume visualization*, pp.91–98, 1994.