中央大学大学院理工学研究科情報工学専攻

修士論文

煙シミュレーションのための部分空間法の高速化

Accelerated Subspace Method for 3D Smoke Simulation

須之内　俊樹
Toshiki SUNOUCHI
学籍番号　23N8100018B

指導教員　　森口　昌樹 准教授

2025年3月

概　要

キーワード: 流体シミュレーション，部分空間法，Snapshot 固有直交分解，Cubature

目 次

第 1 章　序論 **1**

第 2 章　関連研究 **2**

2.1　流体シミュレーション 2

2.2　部分空間法 4

2.2.1　部分空間への射影 5

2.2.2　Snapshot 固有直交分解 5

2.2.3　特異値分解を用いる方法 7

2.2.4　非線形項の計算 7

第 3 章　煙の流体シミュレーション **9**

3.1　格子法による離散化 9

3.1.1　ディリクレ境界条件 11

3.1.2　外力項計算 11

3.1.3　移流項計算 12

3.1.4　粘性項計算 13

3.1.5　圧力項計算 13

3.2　ボリュームデータの可視化手法 17

3.2.1　ボクセルの透明度計算 18

3.2.2　アルファブレンディング 18

3.2.3　スライス方向の比較 18

第 4 章　部分空間法 **20**

4.1　基底空間構築手法 20

4.2　フラクショナルステップ法への部分空間法の適用 21

4.3　部分空間法の課題 22

第 5 章　部分空間法の高速化手法の提案 **23**
5.1　Snapshot の分割による高速化 23
5.2　計算機実験 23
5.2.1　計算条件 23
5.2.2　計算結果 23
5.3　考察 23

第 6 章　結論 **25**

謝辞 **26**

参考文献 **27**

第1章　序論

流体シミュレーションとは，コンピュータを用いて，流体の位置，速度，圧力などの物理量を計算するものである．表現できる流体は，空気，ガスなどの気体や，水，土砂の混ざった水，蜂蜜のような粘性がある液体，砂粒などの固体の粒など多岐にわたる．また，2次元と3次元を扱うことができる．2次元では，水面に広がる波紋や，異なる流体が作り出すマーブル模様などがシミュレーションでき，3次元では私たちの身の回りの現象を広くシミュレーションすることができる．計算した結果を人間が見やすいように可視化することで，流体力学の理論だけでは把握することが困難な，空間に広がる流体の物理量の分布を把握することができるようになる．

流体シミュレーションは，工学分野とコンピュータグラフィックの分野で活用されている．工学分野では，流体に触れる製品の設計・開発において，製品と流体の相互関係を事前にシミュレーションすることで，実験のコスト削減などに役立っている．コンピュータグラフィックスの分野では，水や煙のそれらしいアニメーションを生成し，映像作品やゲームの演出に役立っている．コンピュータグラフィックスの分野では，現実の物理現象に忠実な計算よりも，それらしい振る舞いを高速に計算することや，ユーザーが表現を操作しやすいことが重要視されている．

近代的な高解像度の流体表現は計算負荷が高く，高速化の研究が盛んに行われている．コンピュータグラフィックスの分野における高速化において，大規模並列計算手法の適用性が重要視されている．流体シミュレーションの計算はGPUによる高速化と相性が良く，高速化手法もGPUに適用できるものが望まれている．また，前処理によってシミュレーション計算に利用する行列の次元を削減し計算負荷を削減する，部分空間法が存在する．これらの手法はシミュレーション計算は高速に行うことができるが，前処理の計算負荷や空間計算量が大きいことに注意が必要である．

本研究では，部分空間法の計算負荷削減により，前処理にかかる時間の短縮を目指す．

第2章　関連研究

2.1　流体シミュレーション

まず，流体シミュレーションの分野で一般的に使われている表記の説明をする．位置 $\boldsymbol{x}$ や時刻 t を指定した物理量を表現する際は，$\boldsymbol{u}\left(\boldsymbol{x}, t\right)$ のように表記する．

- Δx,Δt: 離散化する計算格子の格子幅，次の時刻までの時間幅.
- $\boldsymbol{u}$：シミュレーション空間全体の流体の速度ベクトル．$\boldsymbol{u}\left(\boldsymbol{x}, t\right)$ は三次元ベクトルとなる．
- $\boldsymbol{p}$：流体の圧力.
- ρ,ν: それぞれ，流体の密度，流体の粘性．ここでは位置や時刻によらない定数とする．
- $\boldsymbol{f}$: 位置 $\boldsymbol{x}$，時刻 t での流体にかかる外力ベクトル．重力などはここに含める．シミュレーションの手法によっては外力を位置 $\boldsymbol{x}$，時刻 t によって変化する外力を扱うことができるが，今回は簡単のため，重力のみを扱い，どの位置でも，どの時刻でも一定の値として考える．
- ∇: 空間微分演算子ナブラ．スカラー場に作用させると勾配を表し，ベクトル場に作用させると発散を表す．３次元空間なので，$\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z})$
- $\frac{\mathrm{D}}{\mathrm{Dt}} = \frac{\partial}{\partial t} + \boldsymbol{u}\left(\boldsymbol{x}, t\right) \cdot \nabla$ ：ラグランジュ微分, または物質微分. 流体力学のような，連続体を扱う力学で用いられており，流れに沿って移動する物体と同じように移動する観測者から見た，物理量の時間変化率を表している．

下記の式 2.1 で表される式を，ナビエ・ストークス方程式とよび，これは流体力学の支配方程式である．非線形二階微分方程式となっており，代数的に一般解を求める事ができない．下記の式 2.2 は流体の連続の式と呼ばれる，質量保存や流量保存を表す式である．水や砂粒のように圧縮されない流体を扱うときは，流体の密度が常に一定，つまり密度の時間微分が 0 になり，それに伴い式 2.2 の左辺の第二項も 0 になる．これにより，式 2.3 と，

式 2.4 を得る．式 2.1 と式 2.4 または式 2.3 を連立することで非圧縮性流体の速度を求めることができる．ナビエ・ストークス方程式を扱う際は，コンピューターで近似解を解析的に求める手法が用いられる．

$$\frac{\partial}{\partial t}\boldsymbol{u}\left(t\right) = -\left(\boldsymbol{u}\left(t\right)\cdot\nabla\right)\boldsymbol{u}\left(t\right) - \frac{1}{\rho}\nabla\boldsymbol{p}\left(t\right) + \nu\nabla^2\boldsymbol{u}\left(t\right) + \boldsymbol{f} \tag{2.1}$$

$$\frac{\mathrm{D}}{\mathrm{Dt}}\rho + \nabla\cdot\boldsymbol{u}\left(t\right) = 0 \tag{2.2}$$

$$\frac{\mathrm{D}}{\mathrm{Dt}}\rho = 0 \tag{2.3}$$

$$\nabla\cdot\boldsymbol{u}\left(t\right) = 0 \tag{2.4}$$

一般の非線形微分方程式は厳密な解析的解法が存在せず，様々な条件を設定した上で，数値解法により離散化し，近似解を求めるのが一般的である．ナビエ・ストークス方程式の数値解法は，分離解法と連成解法が存在する．分離解法は微分方程式の各項ごとに計算を分割する手法であり，線形問題の収束性が良い利点がある一方，分割した項同士の相互作用を計算できないため，厳密な現象の再現が困難であるという欠点がある．特に圧力の計算を分離する Marker And Cell 法 （MAC 法） [1] と呼ばれる手法を Losasso らが提案し，MAC 法は現在では圧力を分離する解法のことを広く指す．全ての項を個別に分割して解く手法としてフラクショナルステップ法が存在する．式 2.1 の右辺の第一項を移流項，第二項を圧力項，第三項を粘性項，第四項を外力項と呼ぶ．連成解法は分離せずに解くため，より正確に現象の再現ができる一方．計算負荷が高いことや収束性が悪いという欠点がある．分割法はコンピュータグラフィックスや流体解析の分野で広く用いられており，扱う手法によって詳細に計算したい相互作用を選択する．

以降はフラクショナルステップ法によるナビエ・ストークス方程式の分割方法を説明する．まず，上記の式 2.1 の時間微分に前進差分を適用し，以下を得る．

$$\boldsymbol{u}\left(t+\Delta t\right) = \boldsymbol{u}\left(t\right) - \Delta t\left(\boldsymbol{u}\left(t\right)\cdot\nabla\right)\boldsymbol{u}\left(t\right) - \frac{1}{\rho}\nabla\boldsymbol{p}\left(t\right) + \nu\nabla^2\boldsymbol{u}\left(t\right) + \boldsymbol{f}\right)$$

この式を，中間子 $\boldsymbol{u_0}$ から $\boldsymbol{u_3}$ を用いて，以下のように各項ごとに分割して計算する．

$$\boldsymbol{u_0} = \boldsymbol{u}\left(t\right) - \Delta t\boldsymbol{f} \tag{2.5}$$

$$\boldsymbol{u_1}(\boldsymbol{x}) = \boldsymbol{u_0}(\boldsymbol{x}) - \Delta t\left(\boldsymbol{u_0}(\boldsymbol{x})\cdot\nabla\right)\boldsymbol{u_0}(\boldsymbol{x}) \tag{2.6}$$

$$\boldsymbol{u_2} = \boldsymbol{u_1} - \Delta t\nu\nabla^2\boldsymbol{u_1} \tag{2.7}$$

$$\boldsymbol{u}\left(t+\Delta t\right) = \boldsymbol{u_3} = \boldsymbol{u_2} - \Delta t\frac{1}{\rho}\nabla\boldsymbol{p} \tag{2.8}$$

ここで，式 2.5 は外力項，式 3.6 は移流項，式 2.7 は粘性項，式 2.7 は粘性項，式 2.8 は圧力項を計算する式である．

ナビエ・ストークス方程式の空間の離散化については，大きく分けて格子法（Eulerian）と粒子法（Lagrangian）がある．格子法とは，流体を扱う空間を正方形や立方体の格子に区切り，格子の中心や辺，頂点などに物理量を配置して計算する方法である．区切る格子の数が多いほど，流体の詳細な動きが計算できるが，計算負荷は増加する．格子法の利点は，規則正しく並んだ格子で空間を離散化することで，微分演算を差分法などの方法で近似することができることや，境界条件の設定が容易であることが挙げられる．一方で，格子法の欠点は，格子の大きさよりも細かい表現が正確に行えないことや，格子内の流体の粒子の運動を平均化することにより，個々の粒子の詳細な運動を追跡することができないことが挙げられる．これらの欠点により，薄く広がった流体の運動や，水飛沫などを表現することが困難である．

粒子法は，流体の粒子の運動を追跡する手法であり，薄く広がった流体の運動や水飛沫などを表現することが容易である利点がある一方で，計算精度の保証が不十分であることや，境界条件の設定が煩雑になるという欠点がある．

格子法と粒子法は固体にも適用することが可能であり，流体とは異なる支配方程式を用いて変形や移動を計算することができる．流体と固体の相互関係をシミュレーションする際は，格子法と粒子法をそれぞれ個別に適用することが可能である．また，流体の移流項のみを粒子法を用いて計算し，そのほかを格子法で計算するなど，各項の計算について個別に適用する手法が存在する．

2.2 部分空間法

部分空間法は，ベクトル空間をより低次元の部分空間に射影し，数値シミュレーションにおける計算負荷を軽減する手法である．流体シミュレーションにおける部分空間法は，モデル縮約やモデル縮退とも呼ばれている．扱うベクトルデータやシミュレーションの計算を低次元の部分空間に射影して行い計算負荷を軽減する．部分空間上におけるシミュレーションでは，元のシミュレーションよりも Δt を小さい値に設定することで，シミュレーションの時間変化をより詳細に追跡することが可能である．以下では，部分空間法の理論の概要を説明する．

2.2.1 部分空間への射影

シミュレーションの前処理として，n 次元ベクトル $\boldsymbol{x}$ を r 次元ベクトル $\tilde{\boldsymbol{x}}$ に変換する，$n \times r$ 直交行列 $\mathbf{A}$ を考える．以下の 2 式が成り立つ．

$$\tilde{\boldsymbol{x}} = \mathbf{A}^{\mathbf{T}}\boldsymbol{x}$$

$$\boldsymbol{x} = \mathbf{A}\tilde{\boldsymbol{x}}$$

また，$\tilde{\boldsymbol{x'}} = \mathbf{A}^{\mathbf{T}}\boldsymbol{x'}$ を満たす n 次元ベクトル $\boldsymbol{x'}$，r 次元ベクトル $\tilde{\boldsymbol{x'}}$ について，以下の 2 式が成り立つような，任意の $m \times n$ 行列 $\mathbf{F}$，任意の $m \times r$ 行列 $\tilde{\mathbf{F}}$ を考える．

$$\boldsymbol{x'} = \mathbf{F}\boldsymbol{x}$$

$$\tilde{\boldsymbol{x'}} = \tilde{\mathbf{F}}\tilde{\boldsymbol{x}}$$

$\tilde{\boldsymbol{x'}} = \mathbf{A}^{\mathbf{T}}\boldsymbol{x'}$ に，$\boldsymbol{x'} = \mathbf{F}\boldsymbol{x}$ と，$\boldsymbol{x} = \mathbf{A}\tilde{\boldsymbol{x}}$ を代入することで，

$$\tilde{\boldsymbol{x'}} = (\mathbf{A}^{\mathbf{T}}\mathbf{F}\mathbf{A})\tilde{\boldsymbol{x}}$$

が得られる．$\tilde{\boldsymbol{x'}} = \tilde{\mathbf{F}}\tilde{\boldsymbol{x}}$ と比較すると，

$$\tilde{\mathbf{F}} = \mathbf{A}^{\mathbf{T}}\mathbf{F}\mathbf{A}$$

が得られる．

以上により，任意のベクトルに対する行列積を，$n \times r$ 直交行列 $\mathbf{A}$ を用いて部分空間に射影することができる．

2.2.2 Snapshot 固有直交分解

流体シミュレーションにおいて，部分空間に射影したベクトルを用いてシミュレーション計算をするためには，射影後のベクトルも流体の非圧縮性条件等の性質を満たさなければならない．そのような射影を達成する行列 $\mathbf{A}$ を得る手法として，Snapshot 固有直交分解がある．行列 $\mathbf{A}$ を既存の流体のデータを主成分分析を用いて計算することで，流体の性質を満たした射影が可能である．以降は Snapshot 固有直交分解の具体的な計算方法について説明する．

時刻 $t\{t \in N, 0 \leq t \leq m\}$ の n 次元ベクトルデータ $\boldsymbol{u_t}$ を用いて，$n \times m$ 行列 $\mathbf{S}$ を定義する．

$$\mathbf{S} = \begin{bmatrix} | & | & & | \\ \boldsymbol{u}_0 & \boldsymbol{u}_1 & \cdots & \boldsymbol{u}_{m-1} \\ | & | & & | \end{bmatrix}$$

次に，行列 $\mathbf{S}$ 主成分分析を行うため，$n \times r$ 行列 $\mathbf{A}$ について以下の最小化問題を考える．ここで r は抽出したい主成分の数である．

$$min||\mathbf{S} - \mathbf{A}\mathbf{A}^{\mathbf{T}}\mathbf{S}||_F^2$$

この最小化問題の解となる行列 $\mathbf{A}$ は，$\mathbf{S}\mathbf{S}^{\mathbf{T}}$ の固有ベクトル $\boldsymbol{v_i}(0 \leq i \leq r-1)$ を用いて，以下の行列になる．

$$\mathbf{A} = \begin{bmatrix} | & | & & | \\ \boldsymbol{v}_0 & \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_{r-1} \\ | & | & & | \end{bmatrix}$$

ここで，固有ベクトル $\boldsymbol{v_i}$ に対応する固有値を λ_i は，$\lambda_0 \geq \lambda_1 \cdots \geq \lambda_i \geq \lambda_{n-1}$ を満たすとする．$\mathbf{S}\mathbf{S}^{\mathbf{T}}$ は対称行列になっているため，固有ベクトルは互いに直交し，$\mathbf{A}^{-\mathbf{1}} = \mathbf{A}^{\mathbf{T}}$ となる．

次に，行列 $\mathbf{A}$ を用いて $\boldsymbol{u}$ を $\tilde{\boldsymbol{u}}$ に射影するとき，$\tilde{\boldsymbol{u}}$ が流体の非圧縮性条件 $\nabla \cdot \tilde{\boldsymbol{u}} = 0$ を満たしていることを確かめる．行列 $\mathbf{S}$ の各列ベクトル $\boldsymbol{u_i}$ は既存のシミュレーションの速度データであるため，流体の非圧縮性条件 $\nabla \cdot \boldsymbol{u_i} = 0$ を満たす．このことから，

$$\nabla \cdot \mathbf{S} \ = \sum_{0 \leq i \leq n-1} \nabla \cdot \boldsymbol{u_i} = 0$$

が成り立つ．

また，行列 $\mathbf{S}\mathbf{S}^{\mathbf{T}}$ の固有ベクトル $\boldsymbol{v_i}$ と，それに対応する固有値を λ_i とすると，固有値と固有ベクトルの定義から以下が成り立つ．

$$\mathbf{S}\mathbf{S}^{\mathbf{T}}\boldsymbol{v_i} = \lambda_i \boldsymbol{v_i}$$

両辺に左から $\nabla\cdot$ を取ることにより，

$$(\nabla \cdot \mathbf{S})\, \mathbf{S}^{\mathbf{T}}\boldsymbol{v_i} = \lambda_i \nabla \cdot \boldsymbol{v_i}$$

$\nabla \cdot \mathbf{S} \ = 0$ より，0 でない λ_i について，$\nabla \cdot \boldsymbol{v_i} = 0$ が成り立つことがわかる．

$n \times n$ 行列 $\mathbf{S}\mathbf{S}^{\mathbf{T}}$ の主成分分析を行う代わりに，$\mathbf{S}^{\mathbf{T}}\mathbf{S}$ の主成分分析を行い，$\mathbf{S}\mathbf{S}^{\mathbf{T}}$ の固有値固有ベクトルを求めることができる．$\mathbf{S}\mathbf{S}^{\mathbf{T}}\boldsymbol{v_i} = \lambda_i \boldsymbol{v_i}$ の両辺に左から $\mathbf{S}^{\mathbf{T}}$ をかけることにより，

$$(\mathbf{S}^{\mathbf{T}}\mathbf{S})\, \mathbf{S}^{\mathbf{T}}\boldsymbol{v_i} = \lambda_i \mathbf{S}^{\mathbf{T}}\boldsymbol{v_i}$$

を得る．行列 $\mathbf{S}^{\mathbf{T}}\mathbf{S}$ の固有値は，$\mathbf{S}\mathbf{S}^{\mathbf{T}}$ の固有値と一致し，固有値 λ_i 対応する固有ベクトル $\boldsymbol{u_i}$ について，$\boldsymbol{u_i} = \mathbf{S}^{\mathbf{T}}\boldsymbol{v_i}$ であることがわかる．よって，$\mathbf{S}^{\mathbf{T}}\mathbf{S}$ の主成分分析により得られた固有ベクトルと $\mathbf{S}^{\mathbf{T}}$ の逆行列を用いて，$\mathbf{S}\mathbf{S}^{\mathbf{T}}$ の主成分分析を行うことができる．

2.2.3 特異値分解を用いる方法

$n \times r$ 行列 $\mathbf{S}$ の特異値分解は，以下のように定義される．

$$\mathbf{S} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$

ここで，$\mathbf{U}$ は $n \times n$ のユニタリ行列であり，左特異ベクトルを列ベクトルとして持つ．$\mathbf{V}$ は $r \times r$ のユニタリ行列であり，右特異ベクトルを列ベクトルとして持つ．$\mathbf{\Sigma}$ は $n \times r$ の対角行列であり，対角成分には特異値 $\sigma_1, \sigma_2, \ldots, \sigma_r$ が非負の降順で並ぶ．

以下に示す，行列 $\mathbf{S^T S}$ の固有値や固有ベクトルと行列 $\mathbf{S}$ の特異値分解の関係により，$\mathbf{SS^T}$ の主成分分析を行う代わりに，行列 $\mathbf{S}$ の特異値分解を用いることができる．

行列 $\mathbf{S^T S}$ の対角化を，以下のように表す．

$$\mathbf{S^T S} = \mathbf{PAP^{-1}} \tag{2.9}$$

ここで，$\mathbf{A}$ は $n \times n$ の対角行列であり，対角成分には行列 $\mathbf{S^T S}$ の固有値が降順に並ぶ．$\mathbf{P}$ の i 列は固有値 $\mathbf{A_{i,\,i}}$ に対応する固有ベクトルである．

行列 $\mathbf{S}$ の特異値分解より，以下が成り立つ．

$$\mathbf{S^T S} = (\mathbf{V}\mathbf{\Sigma}\mathbf{U}^T)(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T)$$

行列 $\mathbf{U}$ はユニタリ行列，$\mathbf{\Sigma}$ は対角行列であるから，

$$\mathbf{S^T S} = \mathbf{V}\mathbf{\Sigma^2}\mathbf{V}^T$$

式 2.9 と右辺を比較することによって，$\mathbf{P} = \mathbf{V}$，$\mathbf{A} = \mathbf{\Sigma^2}$ であることがわかる．

2.2.4 非線形項の計算

非線形項については，Cubature 法を用いた積分計算の計算量削減手法を用いて次元削減を行う．非線形関数 $\mathcal{F}$ について，$\boldsymbol{f} = \mathcal{F}(\boldsymbol{x})$ とする．ここで，$\boldsymbol{f}$,（$\boldsymbol{x}$）は n 次元ベクトルである．削減前の $\boldsymbol{f}$ は，$\mathcal{F}$ をシミュレーション空間 Ω の全ての計算点 $\boldsymbol{x_p}$ において積分することで求められる．

$$\boldsymbol{f} = \int_{\Omega} \mathcal{F}_p(\boldsymbol{x_p})$$

この積分計算を Cubature 法を用いて次元削減する．Cubature 法はシミュレーション空間全域から適切に計算点を有限個サンプリングし，重み付き和をとることで積分計算を離散

化する．サンプリングした点集合を P，サンプリング点 p に対応する重みを w_p とすると，Cubature 法は以下のように表せる．

$$\boldsymbol{f} = \sum_{p=1}^{P} w_p \mathcal{F}_p(\boldsymbol{x_p})$$

この式は速度に関する基底を用いて以下のように部分空間へ射影することが可能である．

$$\mathbf{U_1} \boldsymbol{f} = \sum_{p=1}^{P} w_p (\mathbf{U^P})^T \mathcal{F}_p(\mathbf{U^P} \boldsymbol{x})$$

ここで，$\mathbf{U^P}$ は $3 \times n$ 基底行列 $\mathbf{U}$ のサンプリング点 p に関する行を抜粋して作成した，$3 \times r$ 行列である．ここで，$\tilde{\boldsymbol{f}}_p = (\mathbf{U^P})^T \mathcal{F}_p(\tilde{\boldsymbol{u}})$ とする．

以降はサンプリング点と重みを評価する手法について説明する．サンプリングした点集合を P とする．以下の式を非負の $\boldsymbol{w}$ に関する線形方程式として解くことで，適切な重みを求める．

$$\begin{bmatrix} \tilde{\boldsymbol{f}}_0^0 & \cdots & \tilde{\boldsymbol{f}}_0^p & \cdots & \tilde{\boldsymbol{f}}_0^P \\ \vdots & \ddots & \vdots & & \vdots \\ \tilde{\boldsymbol{f}}_t^0 & \cdots & \tilde{\boldsymbol{f}}_t^p & \cdots & \tilde{\boldsymbol{f}}_t^P \\ \vdots & & \vdots & \ddots & \vdots \\ \tilde{\boldsymbol{f}}_T^0 & \cdots & \tilde{\boldsymbol{f}}_T^p & \cdots & \tilde{\boldsymbol{f}}_T^P \end{bmatrix} \begin{bmatrix} w_0 \\ \vdots \\ w_p \\ \vdots \\ w_P \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{f}}_0 \\ \vdots \\ \tilde{\boldsymbol{f}}_t \\ \vdots \\ \tilde{\boldsymbol{f}}_T \end{bmatrix} \tag{2.10}$$

ここで，$\tilde{\boldsymbol{f}}_t$ はシミュレーションの Snapshot 行列 $\mathbf{S}$ から抜粋した t 番目の速度 $\boldsymbol{u}_t$ を用いて $\tilde{\boldsymbol{f}}_t = \mathbf{U}\boldsymbol{u}_t$ とする．$\tilde{\boldsymbol{f}}_t^p$ はサンプリング点に関する値 $\mathbf{U^P}\boldsymbol{u}_t^{\boldsymbol{p}}$ である．

サンプリングした点集合は，空集合の状態からランダムに点集合を選び追加する．式 2.10 を線形方程式 $\mathbf{A}\boldsymbol{w} = \boldsymbol{b}$ とし，残差 $\boldsymbol{r} = |\boldsymbol{b} - \mathbf{A}\boldsymbol{w}|$ が閾値以下になるまでサンプリング点を追加する．サンプリングされた点を点集合に追加する際に，以下の式によって求められる確率関数を用いて追加するモンテカルロ法により，更なる効率化を達成できる．

$$f\left(\boldsymbol{x_p}\right) = R\left(\frac{\boldsymbol{a_p} \cdot \boldsymbol{r}}{\boldsymbol{r} \cdot \boldsymbol{r}}\right)$$

第3章　煙の流体シミュレーション

流体シミュレーションを行う際，境界条件や外力項を適切に設定することで流体の振る舞いを操作することができる．この章では文献 [2] を参考に，フラクショナルステップ法を格子法により離散化した煙の流体シミュレーションを実装する方法と，スライスベースのボリュームレンダリングによる可視化手法について説明する．

3.1　格子法による離散化

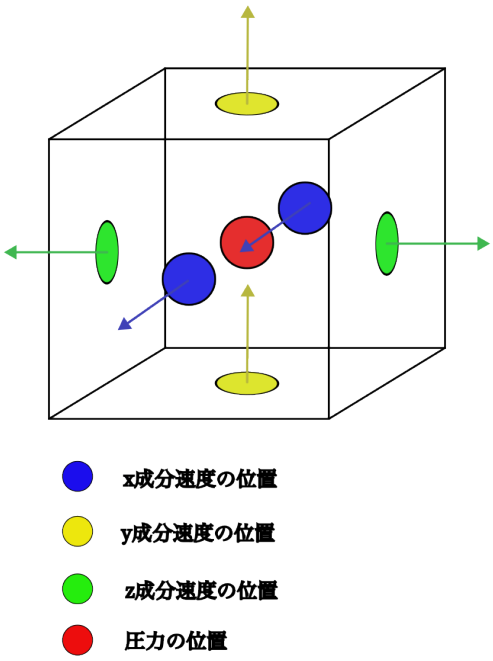

図 3.1: スタッガード格子

格子法による空間の離散化は，扱う格子によって様々な手法が存在する．例えば物理量を格子の中心に配置するコロケート格子や，速度成分を格子の面の中心に配置するスタッガード格子が存在する．コロケート格子は流体の格子の形状によらず適用できるが，境界

条件の設定が煩雑である．スタッガード格子は適用が立方体格子などに限定される一方，境界条件の設定が容易である．スタッガード格子は上の図 3.1 のように，圧力の定義位置を格子の中心とし，格子面の中心にその格子面と垂直な流速の成分の位置を定義する．例えば xy 平面に並行な格子面には，その地点での流速の z 成分を配置する．

シミュレーション空間の x, y, z 軸方向の分割数を nx, ny, nz とする．このとき空間解像度は nx · ny · nz となり，速度の x, y, z 成分の空間解像度はそれぞれ $(nx+1)\cdot ny\cdot nz$, $nx\cdot(ny+1)\cdot nz$，$nx\cdot ny\cdot(nz+1)$ となる．本論文では，$nx=ny=nz$ とし，速度 $\boldsymbol{v}$ と圧力 $\boldsymbol{p}$ は以下のベクトルに離散化する．

- $\boldsymbol{v_x}$：$(nx+1)\times ny\times nz$ 次元ベクトル．
- $\boldsymbol{v_y}$：$nx\times(ny+1)\times nz$ 次元ベクトル．
- $\boldsymbol{v_z}$：$nx\times ny\times(nz+1)$ 次元ベクトル．
- $\boldsymbol{v}$：$3\times(nx+1)\cdot ny\cdot nz$ 次元ベクトル．
- $\boldsymbol{p}$：$nx\cdot ny\cdot nz$ 次元ベクトル．

ここで $\boldsymbol{v}$ の v_0 から $v_{(nx+1)\cdot ny\cdot nz-1}$ には速度の x 成分，$v_{(nx+1)\cdot ny\cdot nz}$ から $v_{2\times(nx+1)\cdot ny\cdot nz-1}$ には速度の y 成分，$v_{2\times(nx+1)\dot{n}y\cdot nz-1}$ から $v_{3\times(nx+1)\cdot ny\cdot nz-1}$ には速度の z 成分を保存する．また，位置 $\boldsymbol{x}=(i,j,k)$ における速度 $\boldsymbol{v}(\boldsymbol{x})$ を，以下のように表す．

$$\boldsymbol{v}(\boldsymbol{x})=\begin{bmatrix}\boldsymbol{v_x}(i,j,k)\\ \boldsymbol{v_y}(i,j,k)\\ \boldsymbol{v_z}(i,j,k)\end{bmatrix}$$

前進差分を用いた空間の偏微分の離散化の例として，スタッガード格子を用いた以下の偏微分の離散化を上記のベクトルを用いて表現することを考える．

$$\frac{\partial}{\partial\boldsymbol{x}}\boldsymbol{u}(\boldsymbol{x})=\frac{\boldsymbol{u}(\boldsymbol{x}+\Delta\boldsymbol{x})-\boldsymbol{u}(\boldsymbol{x})}{\Delta\boldsymbol{x}}$$

ここで，$\boldsymbol{u}(\boldsymbol{x})$ は位置 $\boldsymbol{x}$ での速度を表す三次元ベクトルである．スタッガード格子の一辺の長さを $\Delta L=\Delta\boldsymbol{x}$，流体の位置ベクトル $\boldsymbol{x}$ を自然数 $i\,(0\le i\le nx-1)$, $j\,(0\le j\le ny-1)$, $k\,(0\le k\le nz-1)$ を用いて,$\boldsymbol{x}=(i\Delta L, j\Delta L, k\Delta L)$ とすると，以下のようになる．

$$\frac{\boldsymbol{u}(\boldsymbol{x}+\Delta\boldsymbol{x})-\boldsymbol{u}(\boldsymbol{x})}{\Delta\boldsymbol{x}}=\begin{bmatrix}\frac{\boldsymbol{v_x}(i+1,\,j,\,k)-\boldsymbol{v_x}(i,\,j,\,k)}{\Delta L}\\ \frac{\boldsymbol{v_y}(i,\,j+1,\,k)-\boldsymbol{v_y}(i,\,j,\,k)}{\Delta L}\\ \frac{\boldsymbol{v_z}(i,\,j,\,k+1)-\boldsymbol{v_z}(i,\,j,\,k)}{\Delta L}\end{bmatrix}$$

3.1.1 ディリクレ境界条件

流体シミュレーションにおいて，流体と流体以外の境界面の物理量の条件を設定する必要がある場面がある．これを境界条件と呼び，文献 [2] では境界における物理量を事前に一定の値に設定するディリクレ境界条件が用いられており，境界における物理量は 0 と設定する．ディリクレ境界条件は速度成分を都度初期化することで適用できるほか，境界成分に対応する成分を 0 とした単位行列を用いた表現も可能である．ディリクレ境界条件を表現する行列を $\mathbf{D}$ とすると，$\mathbf{D}$ の i, j 成分は以下のように定義される．

$$\mathbf{D}_{i,\,j} = \begin{cases} 1 & i = j \text{ かつ } i \text{ は境界成分でない} \\ 0 & \text{その他} \end{cases} \tag{3.1}$$

この行列表現は通常の流体シミュレーションでは不要であるが，部分空間法を用いた流体計算では境界成分に対応する値を変更することが困難なため，境界条件を行列積を用いて表現する手法が用いられている．

3.1.2 外力項計算

外力項では，再現したい現象になるような外力を与えなければならない．コンピュータグラフィックスにおける気体の外力は，気体にかかる重力と，温度差による対流を発生させる力，気体が渦巻くように与える力などをユーザーがパラメータによって調整できるようにする計算方法が一般的である．本実験では重力と，対流を発生させる力を加える．よりそれらしい振る舞いを再現するため，他の計算では定数として扱う密度や温度を，ここでは位置と時刻によって変化するデータとして計算する．位置 $\boldsymbol{x}$ における気体の密度，温度をそれぞれ $\rho(\boldsymbol{x})$，$T\,(\boldsymbol{x})$，環境温度を $T_{amb}(\boldsymbol{x})$，重力の方向ベクトルを $\boldsymbol{d}$ とすると，気体に働く外力は以下のようになる．

$$\boldsymbol{f_{buoy}}(\boldsymbol{x}) = \alpha\rho(\boldsymbol{x})\,\boldsymbol{d} + \beta(T\,(\boldsymbol{x}) - T_{amb}(\boldsymbol{x}))\,\boldsymbol{d} \tag{3.2}$$

ここで，α，β はユーザーが設定するパラメータであり，それぞれ重力と対流の力を調整する．最終的な外力項計算は以下のようになる．

$$\boldsymbol{u_0} = \boldsymbol{u} - \Delta t \boldsymbol{f_{buoy}} \tag{3.3}$$

3.1.3 移流項計算

移流項は流体の物理量が流体の速度場によって移動する様を計算することができる．移流項は非線形方程式であり，様々な計算方法が存在する．文献 [2] の Semi-Lagrangian 法を紹介する．Semi-Lagrangian 法は計算負荷が小さく，陰解法により安定した計算結果を得ることができるため，コンピュータグラフィックスの分野で広く用いられている．

Semi-Lagrangian 法は，格子に配置されている計算点上の物理量を計算する格子法（Eulerian）の特徴と，移動する計算点を追跡し計算を行う粒子法（Lagrangian）の特徴を両方持っている計算手法である．移流項の計算

$$\boldsymbol{u_1} = \boldsymbol{u_0} - \Delta t \left(\boldsymbol{u_0} \cdot \nabla\right) \boldsymbol{u_0}$$

を変形し，時刻の偏微分に対し前進差分 $\frac{\partial \boldsymbol{u}}{\partial t} = \frac{\boldsymbol{u_1} - \boldsymbol{u_0}}{\Delta t}$ すると，以下の式が得られる．

$$\frac{\partial \boldsymbol{u}}{\partial t} + \left(\boldsymbol{u_0} \cdot \nabla\right) \boldsymbol{u_0} = 0$$

この式は以下の輸送方程式と呼ばれる式の特殊な場合であり，輸送方程式は一般の物理量 ϕ に関して適用可能である．

$$\frac{\partial \phi}{\partial t} + (\boldsymbol{u} \cdot \nabla)\phi = 0 \tag{3.4}$$

この輸送方程式が成り立つとき，位置 $\boldsymbol{x}$，時刻 t における物理量 $\phi(\boldsymbol{x}, t)$ について，以下の式が成り立つ．

$$\phi(\boldsymbol{x}, t) = \phi(\boldsymbol{x} - \Delta t \boldsymbol{u}, t) \tag{3.5}$$

Semi-Lagrangian 法を以下のように流速に適用することで移流項の計算を行う．また，密度や温度などに対して適用することで，空間の速度ベクトル場に沿って気体が運動する様子を再現することができる．

$$\boldsymbol{u_1}(\boldsymbol{x}) = \boldsymbol{u_0}(\boldsymbol{x} - \Delta t \boldsymbol{u_0}) \tag{3.6}$$

以下では，輸送方程式 3.4 が成り立つとき，式 3.5 が成り立つことを確認する．$\boldsymbol{x}$ にある物理量は $\boldsymbol{u}\left(\boldsymbol{x}\right)$ に沿って $\Delta t \boldsymbol{u}\left(\boldsymbol{x}\right)$ 移動すると仮定すると，移動前の物理量は $\phi(\boldsymbol{x} - \Delta t \boldsymbol{u}\left(\boldsymbol{x}\right))$ となる．この間の物理量の変化 $\Delta\phi$ は以下のようになる．

$$\Delta\phi = \phi(\boldsymbol{u}\left(\boldsymbol{x}\right), t) - \phi(\boldsymbol{x} - \Delta t \boldsymbol{u}\left(\boldsymbol{x}\right), t)$$

これを一次のテイラー展開を用いて表すと,

$$\Delta\phi = \frac{\partial \boldsymbol{u}\left(\boldsymbol{x}\right)}{\partial \boldsymbol{x}} \phi(\boldsymbol{x}, t) \Delta t + \frac{\partial \phi(\boldsymbol{x}, t)}{\partial t} \Delta t$$

$\frac{\partial \boldsymbol{u}(\boldsymbol{x})}{\partial \boldsymbol{x}}\phi(\boldsymbol{x},t) = (\boldsymbol{u}(\boldsymbol{x})\cdot\nabla)\phi(\boldsymbol{x},t)$ より，

$$\frac{\Delta\phi}{\Delta t} = (\boldsymbol{u}(\boldsymbol{x})\cdot\nabla)\phi(\boldsymbol{x},t) + \frac{\partial\phi(\boldsymbol{x},t)}{\partial t}$$

右辺が輸送方程式3.4になっていることから，$\Delta\phi = 0$が成り立つ．$\Delta\phi = \phi(\boldsymbol{u}(\boldsymbol{x})) - \phi(\boldsymbol{x} - \Delta t\boldsymbol{u}(\boldsymbol{x})) = 0$ より，式3.5が成り立つことが確認できる．

3.1.4 粘性項計算

粘性項は流体が徐々に周囲に広がっていく様を計算することができる．粘性項計算は$\Delta t\nu\nabla^2$を行列を用いて表現することで，以下のように行列積を用いて表すことができる．

$$\boldsymbol{u_2} = \boldsymbol{u_1} - \Delta t\nu\nabla^2\boldsymbol{u_1} = (\mathbf{I} - \frac{\Delta t\nu}{(\Delta x)^2}\mathbf{L})\,\boldsymbol{u_1} = \mathbf{W}\boldsymbol{u_1} \tag{3.7}$$

ここで，∇^2の離散化を行列$\mathbf{L}$を用いて表すことを考える．行列$\mathbf{L}$をv_0からv_{n-1}のn個の頂点を持つ無向グラフに対して離散ラプラシアンと呼び，$n\times n$行列$\mathbf{L}$の成分$\mathbf{L}_{i,j}$は以下のように定義される．ただし，無向グラフは自己ループを持たないとする．

$$\mathbf{L}_{i,\,j} = \begin{cases} deg\,(v_i) & i = j \\ -1 & v_i\text{と }v_j\text{が接続している} \\ 0 & \text{その他} \end{cases} \tag{3.8}$$

スタッガード格子を用いた流体シミュレーションにおいて，無向グラフは流体のそれぞれの速度成分ごとの連結成分を持つとする．つまり，本論文のシミュレーションにおいては連結成分数は3であり，それぞれの頂点数は$(nx+1)\times ny\times nz$である．また，シミュレーション境界を除く速度に対応する次数は6であり，シミュレーション境界上の成分はディリクレ境界条件によって速度成分が0になるため，任意の値でよい．よって粘性項計算を表現する行列$\mathbf{V}$の成分$\mathbf{V}_{i,\,j}$は以下のように定義される．

$$\mathbf{V}_{i,\,j} = \begin{cases} 1 - 6\frac{\Delta t\nu}{(\Delta x)^2} & i = j \\ \frac{\Delta t\nu}{(\Delta x)^2} & v_i\text{と }v_j\text{が接続している} \\ 0 & \text{その他} \end{cases}$$

3.1.5 圧力項計算

圧力項計算は，流体の非圧縮性を満たすために流体に圧力勾配による外力を与える計算である．以下では，代表的な圧力項計算手法である，コリンの射影法を紹介する．圧力項

計算によって離散化するのは以下の方程式である．

$$\boldsymbol{u}\,(t+\Delta t) = \boldsymbol{u_2} - \Delta t\frac{1}{\rho}\nabla p$$

上記の式の両辺の発散をとると，

$$\nabla\cdot\boldsymbol{u}\,(t+\Delta t) = \nabla\cdot\boldsymbol{u_2} - \Delta t\frac{1}{\rho}\nabla^2 p$$

が得られる．流体の非圧縮性条件，$\nabla\cdot\boldsymbol{u}\,(t+\Delta t) = 0$ により，左辺は 0 になるため，

$$\nabla\cdot\boldsymbol{u_2} = \frac{\Delta t}{\rho}\nabla^2 p \tag{3.9}$$

が得られる．この式を圧力に関するポアソン方程式として解き，$\boldsymbol{u}\,(t+\Delta t) = \boldsymbol{u_2} - \Delta t\frac{1}{\rho}\nabla p$ に代入することで圧力項計算が完了する．ポアソンの境界条件にはディリクレ境界条件を適用する．

スタッガード格子では圧力と流速の配置されている位置が異なるため，空間微分に対して同じ差分は適応できない．そこで両辺をセルの領域で積分することを考える．

$$\int_V \nabla\cdot\boldsymbol{u_2}dV = \int_V \frac{\Delta t}{\rho}\nabla^2\boldsymbol{p}dV$$

両辺にガウスの発散定理を適用すると，周回積分の形で表せる．

$$\oint_V \boldsymbol{u_2}\cdot\boldsymbol{n}dV = \oint_V \frac{\Delta t}{\rho}\nabla\boldsymbol{p}\cdot\boldsymbol{n}dV$$

左辺から空間微分が消去できたため，右辺の空間微分に対して適切な微分スキームを適用することで，周回積分を離散化することができる．ここで離散化のため，x 軸方向に i 番目，y 軸方向に j 番目，z 軸方向に k 番目の格子を$g_{i,j,k}$，圧力を $p_{i,j,k}$ と定義し，図 3.3 のようなモデルを考える．

まず，整数 $n = 0, 1, 2, \cdots 5$ までを，それぞれ格子の各面の方向に割り振る．図 3.3 の場合，$n = 0$ は x 軸正の方向，$n = 1$ は y 軸正の方向，$n = 2$ は x 軸負の方向，$n = 3$ は y 軸負の方向，$n = 4$ は z 軸正の方向，$n = 5$ は z 軸負の方向に割り振る．ここで，この整数 n を用いて，以下の配列や変数を定義する．

- F_n 格子の面の各方向に隣接しているものが流体なら 1，壁面なら 0 を取るブーリアン値．

  図 3.3 の場合，$n = 0$ に対応する x 軸正の方向と，$n = 5$ に対応する z 軸負の方向は壁に隣接していて格子がないため，$F_0, F_5 = 0, F_1, F_2, F_3, F_4 = 1$ となる．

- D_n 格子の面の各方向に垂直で，格子の外側を向く単位ベクトルを表す係数．向いている方向が座標軸の正の向きなら$+1$， 負の向きなら -1 を取る整数値．

  図 3.3 の場合，座標軸の正の方向に向いているのは $n = 0, n = 1, n = 4$ に対応する方向で，それ以外は負の方向に向いているため，$F_0, F_1, F_4 = 1, F_2, F_3, F_4 = -1$ となる．

- p_n 割り振られた方向に隣り合っている格子の圧力値．図 3.3 の場合，$p_0 = p_{i+1,j,k}$，$p_1 = p_{i,j+1,k}$，$p_2 = p_{i-1,j,k}$，$p_3 = p_{i,j-1,k}$，$p_4 = p_{i,j,k+1}$，$p_5 = p_{i,j,k-1}$ となるが，境界条件を考えると，壁に隣接している格子との圧力値の勾配を 0 にするため，$p_0 = p_{i,j,k}$，$p_5 = p_{i,j,k}$ である．

- ∇p_n 割り振られた方向の圧力勾配. 前方差分によって，$\nabla p_n(\boldsymbol{x}, t) = \frac{p_n - p_{i,j,k}}{\Delta x}$ と近似できる．

- u_n $g_{i,j,k}$ において，割り振られた方向の格子面に配置されている流速値．壁に隣接している方向は流速を 0 にするため，$u_0 = 0$，$u_5 = 0$ である．

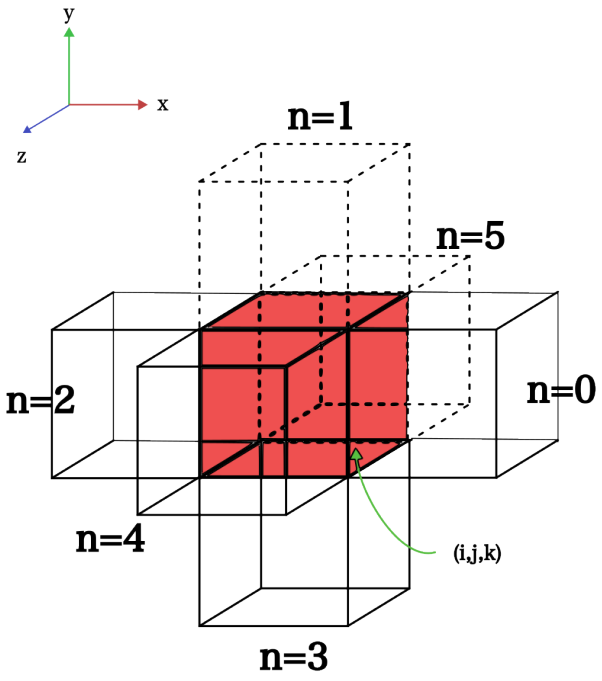

図 3.2: 圧力計算のモデル

定義した配列や変数を用いて，周回積分の形で表された式を離散化することで，下記の式を得る．

$$\sum_n \frac{\Delta t}{\rho} \nabla p_n(\boldsymbol{x}, t)\ F_n = \sum_n u_n D_n F_n$$

圧力項は圧力を変数とした，上記の式を $\boldsymbol{Ax} = \boldsymbol{b}$ に対応させたものを解く．空間解像度を n とすると，$\boldsymbol{A}$ は $n \times n$ 行列であり，$\boldsymbol{b}$ は n 次元ベクトルである．行列 $\boldsymbol{A}$ は規模が大きいため，直接法ではなく反復法を用いて必要な精度で計算を打ち切って計算する．連立方程式の解法としては，前処理付き共役勾配法がよく用いられている．また，行列 $\mathbf{A}$ は離散ラプラシアンを用いて $\mathbf{A} = \frac{\Delta t}{\rho \Delta x}\mathbf{L}$ と計算できる．よって，$\mathbf{A}$ の (i, j) 成分 $\mathbf{A_{i,\,j}}$ の各成分は以下のように計算する．

$$\mathbf{A}_{i,\,j} = \begin{cases} 6\frac{\Delta t}{\rho \Delta x} & i = j \\ -\frac{\Delta t}{\rho \Delta x} & v_i \text{と } v_j \text{が接続している} \\ 0 & \text{その他} \end{cases}$$

ベクトル $\boldsymbol{b}$ は速度ベクトル $\boldsymbol{u_2}$ と行列 $\mathbf{W}$ を用いて，$\boldsymbol{b} = \mathbf{W}\boldsymbol{u_2}$ と表せる．したがって，実際に解く線形方程式は $\mathbf{A}\boldsymbol{p} = \mathbf{W}\boldsymbol{u_2}$ となる．

上記の線型方程式の解法は，行列 $\mathbf{A}$ が正定値対称行列になるため，前処理付き共役勾配法が広く用いられている．また，$\mathbf{A}$ のほとんどの成分は 0 であるため，疎行列の反復法ソルバーを用いることで空間計算量や時間計算量を削減できる．線形代数ライブラリには Eigen を用いる．反復法では，必要な精度に解が収束したら計算を打ち切る．ここでの精度は反復回数や視覚的に影響を与えるほか，低い精度では流体の非圧縮性条件を満たさなくなってしまう恐れがある．文献 [2] のフラクショナルステップ法における計算時間のボトルネックは圧力項計算であり，解の収束が早いほど反復回数が少なくなり，計算負荷が少なくなる．

次に，圧力の勾配ベクトル $\nabla \boldsymbol{p}$ が速度に与える影響を考える．$g_{i,j,k}$ の x 方向の速度成分 $v\,(i, j, k)$ は，

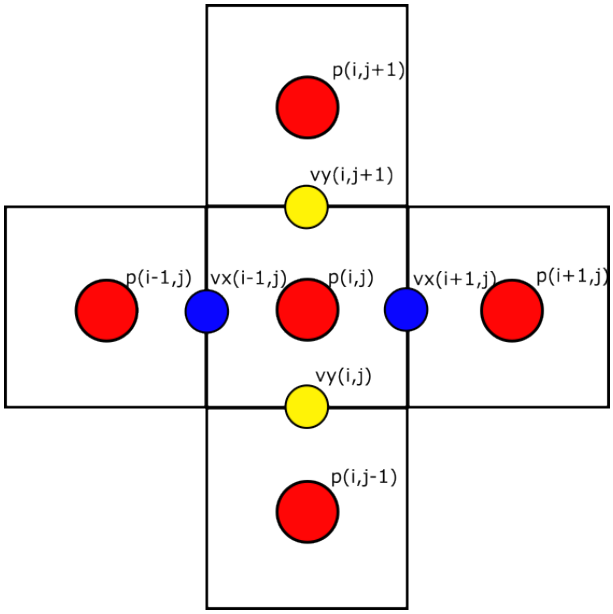

図 3.3: 圧力と速度の配置位置

$\mathbf{A}\boldsymbol{p} = \mathbf{W}\boldsymbol{b}$ によって求めた $\boldsymbol{p}$ を元の式に代入し，最終的な流速を得る．

$$\boldsymbol{u}\left(t + \Delta t\right) = \boldsymbol{u_2} - \Delta t \frac{1}{\rho} \nabla \boldsymbol{p}$$

ここで，$\Delta t \frac{1}{\rho} \nabla \boldsymbol{p}$ を行列 $\mathbf{Y}$ を用いて，$\Delta t \frac{1}{\rho} \nabla \boldsymbol{p} = \mathbf{Y}\boldsymbol{p}$ と表せる．

3.2 ボリュームデータの可視化手法

ボリュームデータを三次元的な広がりの様子を把握できるようにレンダリングする手法をボリュームレンダリングという．描画対象の表面のみをレンダリングするサーフェスレンダリングとは異なり，光の散乱や吸収などの光学的性質を利用して，描画対象の内側のデータも出力画像に反映させる．代表的な手法として，レイトレーシングを用いるボリュームレイキャスティング（Volume Ray Casting）や，サンプルデータをスライスごとに投影するスライスベースの手法が存在する．スライスベースの手法は計算負荷が低くリアルタイム処理に適している．スライスベースのボリュームレンダリングは，シミュレーション空間がボクセル空間として捉えられることを利用し，気体の密度を用いてボクセルの透明度や色を計算する．その後，シミュレーション空間をボクセルの透明度や色をテクスチャとしてマッピングした複数の平面（スライス）に分割し，それらを順次描画することで最終的な画像を生成する．

以下では，スライスベースのボリュームレンダリングのアルゴリズムと，各スライスのテクスチャの計算方法を紹介する．

3.2.1 ボクセルの透明度計算

視線方向に沿ったスライス間の距離を Δs とし，i 番目のスライスと視線の交点の位置を $i\Delta s$ とする．ボクセルの透明度 $t(i\Delta s)$ と不透明度 $\alpha(i\Delta)$ は，以下の式で計算する．

$$t(i\Delta s) = \exp(-\rho(\mathrm{i}\Delta\mathrm{s})\Delta\mathrm{s})$$

$$\alpha(i\Delta s) = 1 - t(i\Delta s)$$

$\rho(i\Delta s)$ は，流体シミュレーションによって計算した，位置 $i\Delta s$ の密度である．また，Δs は視線とスライスの法線がなす角 θ とすると以下のように変化するため，Δs はテクスチャ座標によって異なる．

$$\Delta s' = \Delta s / \cos\theta \tag{3.10}$$

3.2.2 アルファブレンディング

透明度を考慮したレンダリング手法の代表的なものとして，アルファブレンディングが存在する．アルファブレンディングは下記の式で計算する．

$$I_n(\boldsymbol{x}) = \alpha_n(\boldsymbol{x})c_n(\boldsymbol{x}) + (1 - \alpha_n(\boldsymbol{x}))I_{n-1}(\boldsymbol{x}) \tag{3.11}$$

- $I_n(\boldsymbol{x})$: n 番目のスライスの位置 $\boldsymbol{x}$ における累積輝度
- $c_n(\boldsymbol{x})$: n 番目のスライスの位置 $\boldsymbol{x}$ における輝度
- $\alpha_n(\boldsymbol{x})$: n 番目のスライスの位置 $\boldsymbol{x}$ における不透明度

式 3.11 の不透明度に $\alpha(i\Delta s')$ を用いることで，アルファブレンディングによってシミュレーション空間がボリュームレンダリングできる．

3.2.3 スライス方向の比較

シミュレーション空間をスライスによって分割する方向は，視線方向に垂直な方向と，x, y, z 軸の中で視線になるべく垂直な方向が考えられる．一般的に視線とスライスの交点

がボクセルの計算点上にないため，$\rho(i\Delta s)$ を補間して得る必要がある．前者は式 3.10 による Δs の変化量が透視投影の影響しか受けない．一方，後者は視線とスライスの法線がなす角は視点に依存し，より式 3.10 による Δs の変化量が大きい．また，x, y, z 軸の方向による分割は視線の方向によって視線と交差するスライスが少なくなり，視覚的に不自然なスライスの形状が現れることがある．CG の分野においては後者の手法でも見た目上問題ないことが多く，文献 [2] では後者の手法を用いている．

本研究では，コンピュータグラフィックスライブラリ OpenGL と c++を用いて，文献 [2] の煙の流体シミュレーションとスライスベースのボリュームレンダリングを行うソフトウェアを実装した．立方体形状のシミュレーション空間の底面中心から，煙が立ち上る様子をシミュレーションした．

計算時間

第4章　部分空間法

4.1　基底空間構築手法

スタッガード格子を用いた流体シミュレーションの，時刻 $t\{t \in N, 0 \leq t \leq m\}$ の流速データ $\boldsymbol{u_t}$ と圧力データ $\boldsymbol{p}$ を用いて，Snapshot 行列 $\mathbf{S_u}$ と $\mathbf{S_p}$ を定義する．x, y, z 軸方向の分割数を nx, ny, nz($nx = ny = nz$) とすると，$\boldsymbol{u_t}$ と $\boldsymbol{p}$ の次元はそれぞれ 3 ($nx + 1$) $\times nx \times nx$，$nx \times nx \times nx$ である．

$$\mathbf{S_u} = \begin{bmatrix} | & | & & | \\ \boldsymbol{u}_0 & \boldsymbol{u}_1 & \cdots & \boldsymbol{u}_{m-1} \\ | & | & & | \end{bmatrix}$$

$$\mathbf{S_p} = \begin{bmatrix} | & | & & | \\ \boldsymbol{p}_0 & \boldsymbol{p}_1 & \cdots & \boldsymbol{p}_{m-1} \\ | & | & & | \end{bmatrix}$$

流体シミュレーションにおいて，これらの Snapshot 行列はデータ間に相関があるため条件数が大きくなり，計算が不安定になることがある．これらを避けるため，QR 分解を利用する手法を説明する．

まず，$n \times m$ 行列 $\mathbf{S}$ を以下のように QR 分解する．

$$\mathbf{S} = \mathbf{QR}$$

ここで，行列 $\mathbf{Q}$ は $n \times m$ 直交行列であり，行列 $\mathbf{R}$ は $m \times m$ 上三角行列である．その後，行列 $\mathbf{R}$ を以下のように特異値分解する．

$$\mathbf{R} = \mathbf{U_R \Sigma_R {V_R}}^T$$

ここで，$\mathbf{S^T S}$ を計算する．

$$\mathbf{S^T S} = \mathbf{V_R {\Sigma_R}}^T \mathbf{{U_R}}^T \mathbf{Q}^T (\mathbf{Q U_R \Sigma_R {V_R}}^T)$$

$\mathbf{Q}$ と $\mathbf{U_R}$ は直交行列であることから，以下が得られる．

$$\mathbf{S^T S} = \mathbf{V_R \Sigma_R^2 V_R}^T$$

2.2 にて $\mathbf{S^T S} = \mathbf{V \Sigma^2 V}^T$ となることを利用して基底を求めることができることを紹介した．同様にして QR 分解後の特異値分解を利用して基底を求めることができる．この手法により，特異値分解する行列は各列に相関がある $n \times m$ 行列から QR 分解後の $m \times m$ 上三角行列になり，行列の特異性が改善される．

4.2 フラクショナルステップ法への部分空間法の適用

本節では，2.1 にて得た，フラクショナルステップ法による以下のシミュレーション計算を部分空間上に射影して計算する方法を説明する．

$$\boldsymbol{u_0} = \boldsymbol{u}\,(t) - \Delta t \boldsymbol{f}$$

$$\boldsymbol{u_1}(\boldsymbol{x}) = \boldsymbol{u_0}(\boldsymbol{x} - \Delta t \boldsymbol{u_0})$$

$$\boldsymbol{u_2} = \mathbf{V}\boldsymbol{u_1}$$

$$\boldsymbol{b} = \mathbf{W}\boldsymbol{u_2}$$

$$\boldsymbol{p} = \mathbf{A^{-1}}\boldsymbol{b}$$

$$\boldsymbol{u_3} = \boldsymbol{u_2} - \mathbf{Y}\boldsymbol{p}$$

フラクショナルステップ法では，中間子 $\boldsymbol{u_0}$ から $\boldsymbol{u_3}$ までの 4 種類の速度ベクトルを計算する．それぞれの速度ベクトルを部分空間に射影するため，対応した Snapshot 行列が 1 つずつ必要である．中間子 $\boldsymbol{u_i}$ から得た基底を $\mathbf{U_i}$，$\boldsymbol{p}$ から得られた基底を $\mathbf{P}$ とする．フラクショナルステップ法の線形項の計算を部分空間に射影すると，以下のようになる．

$$\begin{aligned}
\mathbf{U_2^T}\boldsymbol{u_2} &= \left(\mathbf{U_2^T V U_1}\right)\mathbf{U_1^T}\boldsymbol{u_1} & \tilde{\boldsymbol{u_2}} &= \tilde{\mathbf{V}}\tilde{\boldsymbol{u_1}} \\
\mathbf{P^T}\boldsymbol{b} &= \left(\mathbf{P^T W U_2}\right)\mathbf{U_2^T}\boldsymbol{u_2} & \tilde{\boldsymbol{b}} &= \tilde{\mathbf{W}}\tilde{\boldsymbol{u_2}} \\
\mathbf{P^T}\boldsymbol{p} &= \left(\mathbf{P^T A^{-1} P}\right)\mathbf{P^T}\boldsymbol{b} & \tilde{\boldsymbol{p}} &= \tilde{\mathbf{A}}\tilde{\boldsymbol{b}} \\
\mathbf{U_3^T}\boldsymbol{u_3} &= \mathbf{U_2^T}\boldsymbol{u_2} - \left(\mathbf{U_3^T Y P}\right)\mathbf{P^T}\boldsymbol{p} & \tilde{\boldsymbol{u_3}} &= \tilde{\boldsymbol{u_2}} - \tilde{\mathbf{Y}}\tilde{\boldsymbol{p}}
\end{aligned}$$

ただし，$\mathbf{P^T A^{-1} P} = \left(\mathbf{P^T A P}\right)^{-1}$ であり，圧力ベクトルの次元数を n として $\mathbf{A}$ は $n \times n$ 疎行列であり，$\left(\mathbf{P^T A P}\right)$ は $m \times m$ 密行列である．密行列であることに注意が必要だが，逆行列を高速に求められるため，$\tilde{\mathbf{A}} = \left(\mathbf{P^T A P}\right)^{-1}$ とする．

部分空間法によるシミュレーション計算は，前処理として基底と部分空間に射影した行列を計算する．基底を計算するステップの計算量のボトルネックは，QR 分解の $O\,(nm^2)$ である．部分空間に射影した行列を計算するステップについては，射影前のそれぞれの行列が非常に疎であるため，基底の計算ほど時間はかからない．

非線形項は 2.2.4 にて紹介した Cubature 法を用いて計算する．Cubature 法に用いる基底は，外力項は $\mathbf{U}_0$ であり，移流項は $\mathbf{U}_1$ である．例えば，移流項を最終的な部分空間に射影した移流項計算は以下のようになる．

$$\tilde{\boldsymbol{u}_2} = \sum_{p=1}^{P} w_p \tilde{\boldsymbol{f}}_p = \sum_{p=1}^{P} w_p (\mathbf{U}_1^{\mathrm{p}})^T \mathcal{F}_p(\tilde{\boldsymbol{u}_1})$$

4.3 部分空間法の課題

部分空間法はシミュレーションの次元を n 次元から r 次元に削減するため，各ステップの計算負荷を大幅に減らすことができる手法である．その一方で，前処理として基底の計算や Cubature 法のランダムサンプリングと重み計算，シミュレーションに用いる行列の部分空間への射影などを行う必要がある．これらの計算負荷は Snapshot の数に依存しているが，Snapshot の数以下の次元にしか次元削減できず，流体シミュレーションとして意味のある結果を得るためには Snapshot の数が十分でなければならない．また，Snapshot 行列の空間解像度が大きいという問題点がある．例えば空間解像度 $64 \times 64 \times 64$，Snapshot 数 50 とすると，流速の Snapshot 行列の空間解像度は 10GB 程度になる．GPU による高速化を併用する際，GPU が 1 度のデータ通信によって処理し切ることができる空間解像度の制限を超えてしまうことで計算時間が増加することが考えられる．

他にも，事前計算したシミュレーションと異なるシミュレーションを行うことが困難であり，境界条件や流体の初期化，与える外力を大幅に変更することはできず，別途シミュレーションデータを用意する必要がある．

第5章　部分空間法の高速化手法の提案

部分空間法の前処理において，基底の計算と行列の射影の計算量はSnapshotの2乗に比例し，前処理の大多数の時間を占めている．この章では部分空間法の前処理の高速化を達成するため，Snapshotをd個に分割する手法を提案する．

5.1　Snapshotの分割による高速化

基底の計算に用いるSnapshotをd分割することによって，基底1つの計算負荷$O(nm^2)$が$\frac{1}{d^2}$に削減される．アルゴリズム全体ではd回計算するため，計算負荷は$\frac{1}{d}$になることが期待できる．行列の射影計算も分割した基底ごとに行う．分割前の基底の削減した次元をrとすると，分割する基底の削減した次元は$\frac{r}{d}$とすることにより基底に用いる空間計算量は変わらない．射影した行列のサイズは$r \times r$から$\frac{r}{d} \times \frac{r}{d}$になる．

5.2　計算機実験

5.2.1　計算条件

5.2.2　計算結果

非線形項については追加の実験を行ったところ，サンプリングされる点によって重みの計算が発散する場合がある．計算時間はSnapshot数については線形時間であるが，サンプリングする点の数が減ることによりNNLSソルバの反復回数が減ることで，実質的に高速化が見込めることもあった．

5.3　考察

削減後の次元数を分割ごとに揃える必要がある．分割後の基底の条件数が異なるため，揃えた次元数では過剰であったり，不足する可能性がある．次元数が過剰になった基底で

は条件数が大きくなり，不安定な部分空間を構築する場合がある．次元数が不足した基底では，シミュレーション計算の精度が下がる．

第6章　結論

基底の計算に用いるSnapshotをd分割することによって，基底計算全体の計算負荷は$\frac{1}{d}$になることが期待できる．

結論には，研究の成果や意義その他を総括的に**過去形**で述べる．

謝辞

本研究を進めるにあたり，大変多くのご指導，ご助言を頂いた中央大学理工学部情報工学科の森口　昌樹 准教授に深く感謝いたします．また，多大なるご助言，ご協力を頂いた形状情報処理研究室の皆様には大変お世話になりました．心から感謝いたします．

参考文献

[1] F. H. Harlow and J. E. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with a free surface. *The Physics of Fluids*, 8（12）: 2182–2189, 1965.

[2] R. Fedkiew, J. stam, H. Jensen. Visual simulation of smoke. In *Proceedings of SIGGRAPH 01*, 15–22, 2001.

[3] J. Stam. Stable Fluids. In *SIGGRAPH 99 Conference Proceedings, Annual Conference Series*, pages 121–128, 1999.

[4] T. Kim, J. Delaney. Subspace Fluid Re-Simulation. *ACM Transactions on Graphics*,32（4）: 62:1–62:9, 2013.

[5] A. Jones, P. Sen, and T. Kim. Compressing fluid subspaces. *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 77–84, 2016.

[6] M. Wicke, M. Stanton, and A. Treuille, Modular Bases for Fluid Dynamics. *ACM Transactions on Graphics*, 39:1–39:8, 2009.