多相流体シミュレーションを可能とする
非圧縮性SPH法の開発

筑波大学
図書館情報メディア研究科
2019年3月
渡辺　拓希

目 次

第 1 章

序論

1.1 研究背景

流体の運動をシミュレートする技術として，Smoothed Particle Hydrodynamics (SPH) 法は長きにわたり研究されてきた [8, 17, 18, 13]．SPH 法は粒子法と呼ばれるシミュレーション手法の一種であり，図 1.1 のようにシミュレーションの対象物を粒子群として近似的に表現し，各粒子で運動計算を行うことで対象物全体の運動を表現する．オリジナルの手法は 1977 年に Gingold と Monaghan[8] および Lucy[16] によって提案され，当時は宇宙物理学における銀河の形成をシミュレートするための手法であった．90 年代以降には流体シミュレーションに応用され始め，現在では炎，雷，雲，爆発といった様々な自然現象シミュレーションが SPH 法によってなされている．用途としては物理学に基づいた解析が主であったが，近年ではコンピュータ性能の向上により，映像作品といったメディアコンテンツ制作にも活用されている．

シミュレーションの対象となる現象の一つに，複数の流体が互いに影響を及ぼし合う多相流体流れがある．現実世界では，水と油が混じり合わず流れたり，液体内で上昇する気体の泡といったものがこの現象の例として挙げられる．そしてこれらのような現象をシミュレートする場合，異なる性質を持つ流体の間における正確な相互作用計算が必要となる．相互作用計算で重要となる要素として，流体境界における正確な密度計算と非圧縮性がある．密度は流体の種類によって異なり，密度の違いは浮力の発生などの要因となるため，特に相互作用計算への影響が大きい．非圧縮性は体積保存性とも呼ばれ，流体がその密度を一定に保とうとする性質であり，水などの液体は非圧縮性流体である．この性質は，流体力学の分野では一般的に圧力のポアソン方程式によって表現される．しかしながら，従来の SPH 法による計算はいずれの要素も考慮されていないため，多相流体シミュレーションを行うことが困難であった．そのため多くの研究者によって各要素を考慮した SPH 法が開発されてきたが，両者を満たす手法は未だ提案されていない．

また，液体をシミュレートする際に表面張力計算を行うことで，より幅広い現象の表現が可能となる．表面張力は，流体の表面積を最小にするように働く力であり，流体同士が混ざらず分離する性質や，液体内または液体表面における泡の形状を保つ要因となる．表面張力は本来液体を構成する分子の動きによって生じる力であるが，SPH 法では粒子一つ一つを分子として見立て，粒子間力として表面張力を求める方法が計算としても安定しているため，よく用いられる．SPH 法における表面張力計算の手法はいくつか提案されているが，薄い液膜で覆われた泡の維持や破裂を再現した研究は存在しない．

図 1.1: SPH 法による液体シミュレーション

1.2 研究目的

本研究では，より安定した多相流体シミュレーションを行うための非圧縮性 SPH 法を提案し，異なる液体同士や液体と気体が相互作用するような様々なシーンを再現する．提案手法では，物理量パラメータである密度とは別に，粒子の分布に依存する粒子密度というパラメータを基準とした計算方法を適用する．これにより，境界付近に異なる流体粒子が存在する場合でも，それらの性質に影響されずに正確な密度計算ができる．また，粒子密度を用いた圧力のポアソン方程式を新たに定義することで，非圧縮性を維持した状態で複数の流体を同時にシミュレートすることが可能となる．表面張力計算に関しては従来法をそのまま用いるが，提案手法に適用することで，従来法では実現できなかった薄い液膜を含む泡の形状維持や破裂の再現が可能となる．

1.3 本論文の構成

本論文は，全 5 章から構成される．第 1 章では，序論として研究背景と本研究の目的を示した．第 2 章では，これまで行われてきた多相流体シミュレーションに関する研究および非圧縮性 SPH 法に関する研究について述べる．第 3 章では，SPH 法シミュレーションについての詳細を示した上で，本研究で提案する手法について述べる．第 4 章では，提案法を用いて様々なシーンの実験を行った結果を示す．最終章では，結論として本研究のまとめと今後の展望について述べる．

第 2 章

関連研究

2.1 多相流体シミュレーションに関する研究

2.1.1 SPH 法によるシミュレーション

Müller ら [20] は従来の SPH 法をそのまま用いて，性質の異なる複数の流体を同時にシミュレートした．図 2.1 では，密度の大きい赤色の流体と密度の小さい透明な流体を用いて，密度の大きい流体は下の方へ，密度の小さい流体は上の方に向かう様子が再現できている．しかし，1.1 節でも述べたように，従来の SPH 法は流体の境界付近において正確な密度計算ができないため，この研究では流体間の密度比を大きく設定できない．

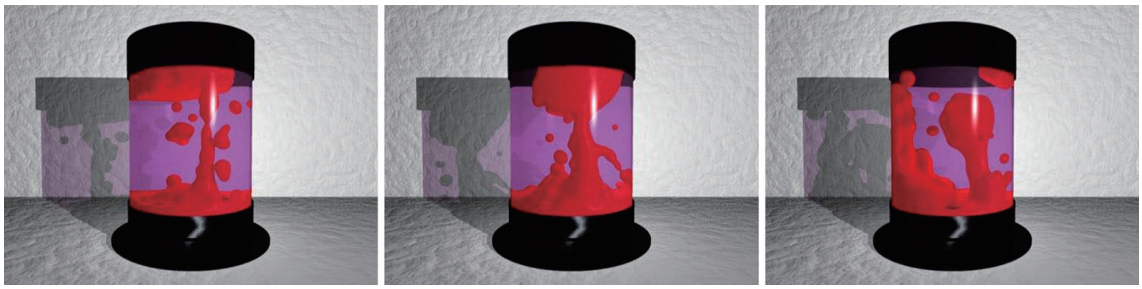


図 2.1: 2 種類の流体を用いたシミュレーション ([20] より引用)

Ihmsen ら [12] は液体粒子と気体粒子の 2 つを用いて，図 2.2 のような泡の発生から消滅までのシミュレーションを行った．手法としては，複数の気体粒子からなる泡を液体シミュレーション中に発生させることで，液体内を泡が動く様子を再現している．この手法では，液体と気体それぞれの粒子が重なることを許容しており，特に気体粒子はシミュレーション中に突然発生したり消滅したりするという非物理的な操作を行っているため，相互作用計算が正確にできない．そのため，水面に浮かぶ泡が割れる様子などは再現できない．

Solenthaler と Pajarola[23] は密度計算の際に，近傍の粒子分布から粒子密度を求め，それを用いた密度計算式を新たに定義することで，流体の界面付近における正確な密度計算を実現した．図 2.3 において，左の図は従来の SPH 法によるシミュレーション結果であり，密度計算を正確に行えないため，不正確な相互作用計算により境界付近で不自然な隙間が発生している．一方，右の図は [23] の粒子密度を用いた密度計算によるシミュレーション結果であり，境界付近で自然な粒子分布が生成できている．後に Szewc ら [24] は粒子密度を用いた密度計算を基に，単体の泡シミュレーションを行った．これらの手法は圧力計算に陽解法を用いているため，非圧縮性が満たされていない．また陽解法を用いると，シミュレーションを行う際のタイムステップ幅に上限を設けなければならず，最終的なシミュレーション結果が得られるまでにかなりの時間を要してしまう．

図 2.2: 水面周辺の空気が巻き込まれることで発生する泡のシミュレーション ([12] より引用)

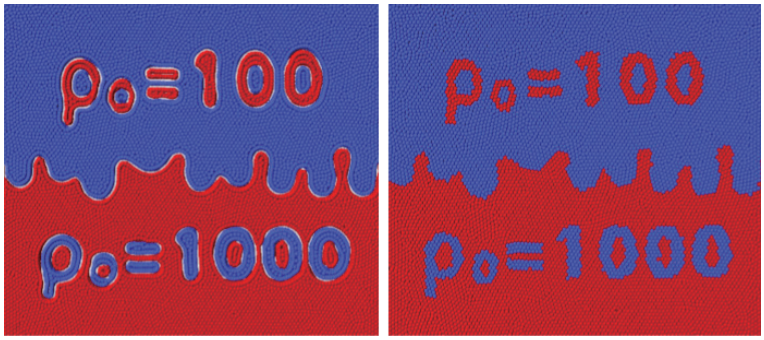


図 2.3: 流体境界における正確な密度計算を考慮したシミュレーション ([23] より引用)

2.1.2 格子法によるシミュレーション

粒子法とは別に，格子法と呼ばれるシミュレーション手法がある．格子法は，シミュレーション領域を格子状に分割し，固定された格子位置における運動を計算することで，シミュレーション対象の動きを表現する手法である．各格子の運動は，それに隣接する格子からの影響により計算される．Hong と Kim[9] は，レベルセット法で表された自由表面境界において，異なる流体を含む格子からの影響を計算する際，相手の値をそのまま用いるのではなく，ゴースト値を用いることでより安定した圧力場および粘性場の生成を実現した．この手法は，シミュレーション空間内の液体部分のみ運動計算を行っており，気体の動きから受ける影響を考慮していない．Zheng ら [29] はレベルセット法を拡張し，図 2.4 のように気泡ごとに領域を定義する手法を提案した．状況に応じて領域番号を変更することで，気泡の結合や破裂を表現した．この手法は領域ごとに独立して運動計算を行っており，別の領域からの影響を直接考慮していないため，正確な相互作用計算を行っていない．

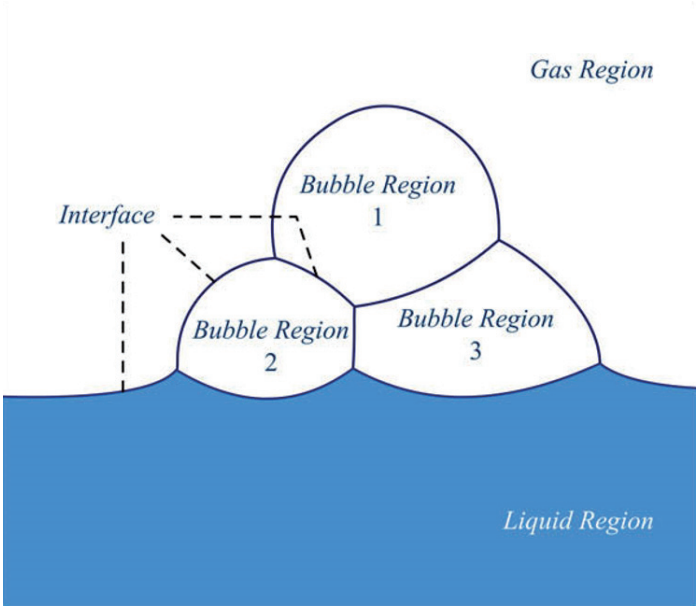
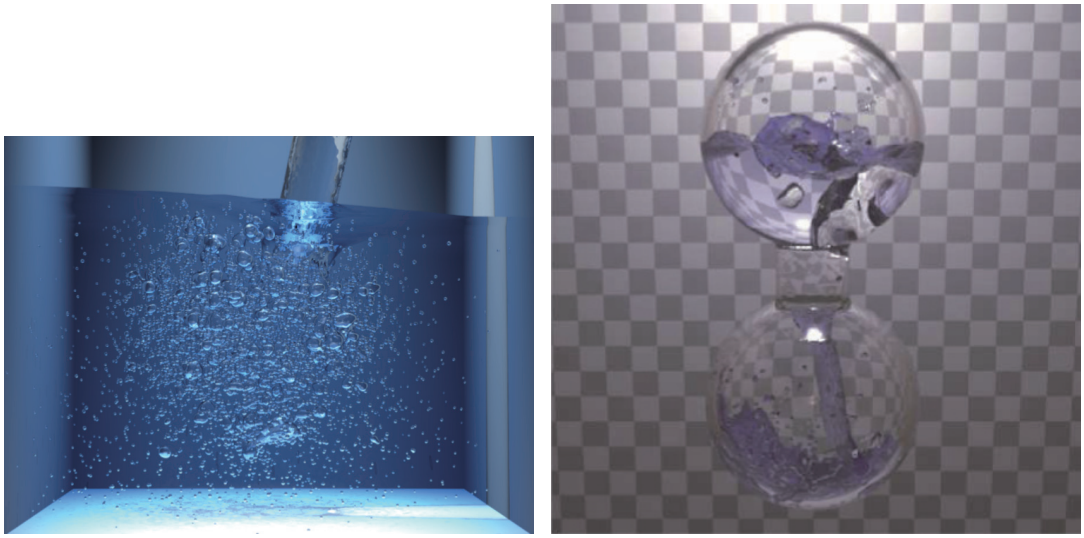


図 2.4: 領域分割による気泡シミュレーション ([29] より引用)

2.1.3 粒子法と格子法を用いたハイブリッド法によるシミュレーション

気泡シミュレーションに関する研究のうち，液体の運動を格子法，気泡の運動を粒子法により計算する手法がある [10, 7]．図 2.5(a) では，気泡への液体格子からの影響と，液体への気泡粒子からの影響を考えることで，相互作用計算を実現している．これらは，気泡を単体の粒子または粒子の塊で表現しているため，[12] と同様の理由により水面まで到達した後の気泡のふるまいを表現できない．別の手法として Boyd ら [6] は，Fluid Implicit Particle(FLIP) と呼ばれるシミュレーション手法を用いて液体と気体の運動を同時にシミュレートした．FLIP は，運動計算を格子上で行い，計算結果をもとに粒子の位置を更新することで，格子法で問題となる移流項計算を安定して行える手法である．図 2.5(b) に示すように，格子上に液体と気体それぞれの速度場を定義し，粒子も同様に液体と気体の 2 種類を用意することで，双方の影響を考慮したシミュレーションを可能とした．この手法は 2 種類の情報を互いに考慮した上での相互作用計算を行うため，とても複雑な運動計算を行わなければならない．また流体の種類が増えると，それに伴い格子に格納するパラメータが増え，粒子数も増加するため，多くのメモリと手間がかかる．

本論文におけるシミュレーション手法は粒子法のみをベースとしている．粒子法では，各相の粒子を用意するだけで簡単に相の区別ができ，運動計算の際には移流計算を数値的に解く必要が無く，他の計算に関しても容易に実装することができる．一方，格子法では，一つの格子領域内に流体境界が存在する場合，正確に相を区別することは難しくなり，正確に行わない場合，体積保存が失われてしまう．また高精度な移流計算が必要なため，粒子法と比較して運動計算の実装が複雑となる．これらの理由から，新しい多相流体シミュレーション手法を提案する際，幅広い現象を容易に計算できるように粒子法をベースとした．



(a) [10] より引用　　(b) [6] より引用

図 2.5: ハイブリッド法による気泡シミュレーション

2.2　非圧縮性 SPH 法に関する研究

非圧縮性はリアルでかつ安定したシミュレーションを行うにあたって重要な要素であり，これが満たされない場合，シミュレーション全体が潰れたような見た目となってしまい，密度計算が正確に行われなくなる．一般に流体シミュレーションでは，圧力のポアソン方程式を解くことで流体の非圧縮性を確保するが，陰解法により直接解くと膨大な計算量となってしまうため，代替法によって圧力計算を行うことが多い．Müller ら [19] および Becker と Teschner[4] の手法では，SPH 法における圧力計算を陽的に解いている．これらは圧力値を一回の計算で求めることができる代わりに，ある程度またはかなりの圧縮を許容してしまう．別の手法として，密度がほぼ一定に保たれるような圧力値を，反復法により求めるものがある [22, 5]．これらは流体の運動計算において，圧力勾配項以外の計算項を最初に解き，それを用いて次ステップの仮速度および仮位置を求める．そして仮位置から求めた仮密度が，基準値となる初期密度とほぼ等しくなるように，圧力勾配項を用いて仮速度・位置を修正することで，非圧縮性を確保する．また，同じ反復法で，SPH 法により離散化された圧力のポアソン方程式を解く手法がある [11]．反復法は非圧縮性を確保する上で非常に有効な手法であるが，これまでに CG 分野で多相流体シミュレーションに適用された例はなく，本研究が初めての試みである．

第3章

提案手法

3.1 シミュレーションの流れ

本研究での流体シミュレーションは，流体の支配方程式であるナビエ・ストークス方程式に基づく．粘性流体のナビエ・ストークス方程式は，左辺が流体速度時間微分で，加速度を表す．右辺は第一項目からそれぞれ移流項，圧力勾配項，粘性拡散項，外力項で構成されている．

$$\frac{\partial \boldsymbol{v}}{\partial t} = -(\boldsymbol{v}\cdot\nabla)\boldsymbol{v} - \frac{\nabla p}{\rho} + \frac{\mu}{\rho}\nabla^2\boldsymbol{v} + \frac{\boldsymbol{f}}{m} \tag{3.1}$$

ここで，$\boldsymbol{v}$ は速度，ρ は密度，p は圧力，m は質量，t は時間，μ は粘性係数，$\boldsymbol{f}$ は外力である．移流項は，速度勾配に従って流体が自身の速度を移動させる作用を表す．SPH 法における移流項の扱いについては，粒子の移動自体が流体の移流を表現しているため，数値的に解く必要が無く計算から除くことができる．圧力勾配項は，圧力の高いところから低いところに向かって流体が移動するような作用を加える項であり，粘性拡散項は，流体の粘性に従って速度を周囲の流体に拡散させるような作用を加える項である．外力には重力や表面張力などが含まれる．また，非圧縮性を表す方程式として，連続方程式を式 (3.2) に示す．

$$\frac{\partial \rho}{\partial t} = -\rho\nabla\cdot\boldsymbol{v} \tag{3.2}$$

これらの式をもとに，Algorithm 1 に示した手順で非圧縮性 SPH 法の多相流体シミュレーションを行う．ある時刻 t において，始めに密度計算を行い，次にナビエ・ストークス方程式の圧力勾配項以外を計算し，圧力勾配項以外からの影響による微小時間 (シミュレーションのタイムステップ幅)Δt 秒後の中間速度を求める．中間速度は Δt 秒後における粒子の仮速度であり，それを用いて提案法より圧力値を求め，圧力勾配項を計算する．そして圧力勾配に基づいて，非圧縮性を満たす時刻 $t+\Delta t$ での粒子の速度と位置を求め，時刻 t における運動計算を終了する．以降，本研究におけるナビエ・ストークス方程式は以下のように簡略化した形で表す．

$$\boldsymbol{a}_i = \boldsymbol{a}_i^p + \boldsymbol{a}_i^v + \boldsymbol{a}_i^s - \boldsymbol{g} \tag{3.3}$$

$\boldsymbol{a}_i, \boldsymbol{a}_i^p, \boldsymbol{a}_i^v, \boldsymbol{a}_i^s$ はそれぞれ粒子 i の加速度，圧力勾配項，粘性拡散項，表面張力項を表す．$\boldsymbol{g}$ は重力加速度である．

Algorithm 1 シミュレーションの流れ

while Simulation **do**
  for all fluid particle i **do**
    密度計算：式 (3.26)
    粘性拡散項の計算：式 (3.28)
    表面張力計算 (3.5 節)
    中間速度の計算：式 (3.31)
  end for
  提案手法による圧力計算 (3.4 節)
  for all fluid particle i **do**
    速度更新：式 (3.38)
    位置更新：式 (3.14)
  end for
  $t \leftarrow t + \Delta t$
end while

3.2 SPH 法

3.2.1 SPH 法によるナビエ・ストークス方程式の離散化

SPH 法において，流体を構成する各粒子の運動は，図 3.1 のような粒子分布において，計算対象の粒子の近傍にある粒子との相互作用によって決まる．各粒子はそれぞれ位置，速度，質量，密度，圧力といったパラメータを保持しており，それらの情報を用いて，3.1 節に示したナビエ・ストークス方程式に基づいて相互作用計算を行う．しかし，ナビエ・ストークス方程式はそのままの形では計算ができないので，コンピュータが計算できるような形に離散化する必要がある．SPH 法の場合，計算点の周囲にある粒子が保持しているパラメータの重み付き和をとることで，コンピュータが計算できる形で物理量の連続場を定義する．任意の位置 $\boldsymbol{x}$ におけるパラメータ ϕ は

$$\phi(\boldsymbol{x}) = \sum_j \frac{m_j}{\rho_j} \phi(\boldsymbol{x}_j) W(\boldsymbol{x} - \boldsymbol{x}_j, h) \tag{3.4}$$

のように離散化される．ここで，添え字 j は近傍の粒子，h は有効半径，W は粒子間距離に応じた重み関数であり，カーネル関数と呼ぶ．本研究で用いるカーネル関数については 3.2 節で詳しく述べる．粒子 i の位置を $\boldsymbol{x}_i$ とすると，ϕ の勾配 $\nabla\phi$ およびラプラシアン $\nabla^2\phi$ は

$$\nabla\phi(\boldsymbol{x}_i) = \sum_j \frac{m_j}{\rho_j} \phi(\boldsymbol{x}_j) \nabla W(\boldsymbol{x}_{ij}, h) \tag{3.5}$$

$$\nabla^2\phi(\boldsymbol{x}_i) = \sum_j \frac{m_j}{\rho_j} \phi(\boldsymbol{x}_j) \nabla^2 W(\boldsymbol{x}_{ij}, h) \tag{3.6}$$

のように離散化される．ここで，$\boldsymbol{x}_{ij} = \boldsymbol{x}_i - \boldsymbol{x}_j$ である．これらをもとにナビエ・ストークス方程式を離散化することで，粒子の運動が求まる．文献 [19] では，密度，圧力勾配項，粘性拡散項はそれぞれ，

$$\rho_i = \sum_j m_j W(\boldsymbol{x}_{ij}, h) \tag{3.7}$$

$$\boldsymbol{a}_i^p = -\frac{1}{\rho_i} \sum_j m_j \frac{p_i + p_j}{2\rho_j} \nabla W(\boldsymbol{x}_{ij}, h) \tag{3.8}$$

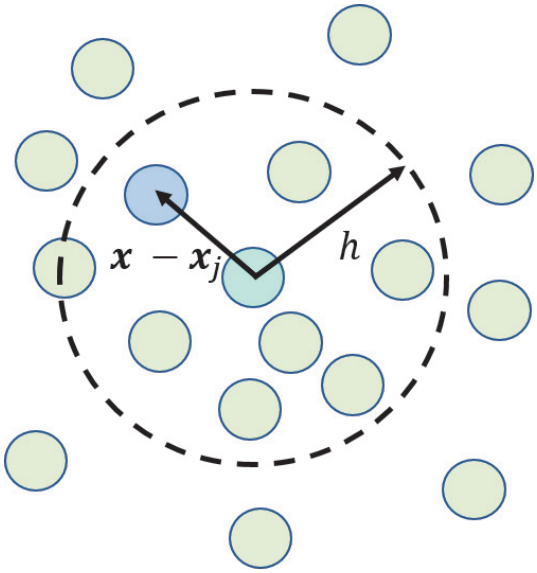


図 3.1: SPH 法における近傍粒子との相互作用計算

$$\boldsymbol{a}_i^v = \frac{\mu}{\rho_i} \sum_j m_j \frac{\boldsymbol{v}_j - \boldsymbol{v}_i}{\rho_j} \nabla^2 W(\boldsymbol{x}_{ij}, h) \tag{3.9}$$

のように離散化している．圧力勾配項と粘性拡散項の離散化の形が式 (3.5) および式 (3.6) と異なっているが，これは近傍粒子間での作用の対称性を考慮しているためである．他の離散化式として，文献 [20] では，粘性の異なる流体からの影響を考慮するため，粘性拡散項を

$$\boldsymbol{a}_i^v = \frac{1}{\rho_i} \sum_j \frac{\mu_i + \mu_j}{2} m_j \frac{\boldsymbol{v}_j - \boldsymbol{v}_i}{\rho_j} \nabla^2 W(\boldsymbol{x}_{ij}, h) \tag{3.10}$$

のように，粘性係数の平均を取ることで対称性を保持している．また，文献 [17] では作用の対称性と運動量保存をより正確にするため，圧力勾配項を商の法則より

$$\frac{\nabla p}{\rho} = \nabla \left(\frac{p}{\rho} \right) + \frac{p}{\rho^2} \nabla \rho \tag{3.11}$$

と書き換えてから離散化し，

$$\boldsymbol{a}_i^p = -\sum_j m_j \left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) \nabla W(\boldsymbol{x}_{ij}, h) \tag{3.12}$$

という形で計算している．

ナビエ・ストークス方程式の各項を計算したのち，現在時刻の Δt 秒後における粒子の速度と位置を求める．実際には前進オイラー法を用いて，

$$\boldsymbol{v}_i(t + \Delta t) = \boldsymbol{v}_i(t) + \boldsymbol{a}_i \Delta t \tag{3.13}$$

$$\boldsymbol{x}_i(t + \Delta t) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t + \Delta t) \Delta t \tag{3.14}$$

のように更新する．

3.2.2 圧力計算

2.2 節でも触れたように，流体シミュレーションにおける圧力計算は，圧力のポアソン方程式を直接解いて非圧縮性を保つような圧力を求めるという方法が望ましい．圧力のポアソン方程式は式 (3.1) の両辺に ∇ を掛けて，式 (3.2) を代入して整理することで得られ，式 (3.15) のような形で表される．

$$\nabla^2 p = \frac{\rho}{\Delta t} \nabla \cdot \boldsymbol{v} \tag{3.15}$$

圧力のポアソン方程式は離散化することで線形システムとなるが，これを陰解法により直接解くと膨大な計算量となってしまう．そのような課題もあって，SPH 法では代替法により圧力計算を行う場合がほとんどである．

SPH 法によるシミュレーションでは，文献 [19] で提案された

$$p_i = k(\rho_i - \rho_{0i}) \tag{3.16}$$

がよく用いられる．ここで，ρ_0 はシミュレーション対象となる流体の初期密度であり，単一流体のシミュレーションは全体で一つの ρ_0 を用いるが，本研究では多相流体を扱うため，粒子ごとの初期密度 ρ_{0i} を用いている．k はガス定数である．この式は，圧力と密度の関係を表す理想気体の状態方程式が元となっており，一回の計算で圧力を求めることができる．とても簡単な方法である代わりに非圧縮性が保てず，粒子分布が不均一となり，重力がかかるシーンでは，シミュレーション対象が全体的に潰れたような見た目となってしまう．文献 [4] では，陽解法でありながらもある程度の非圧縮性が保てるように，以下の Tait 方程式を用いた．

$$p_i = \frac{\rho_{0i} c_s^2}{\gamma} \left(\left(\frac{\rho_i}{\rho_{0i}} \right)^{\gamma} - 1 \right) \tag{3.17}$$

ここで，c_s は流体内の音速，$\gamma = 7$ が一般的に用いられる．Tait 方程式も状態方程式であるが，式 (3.16) よりも圧縮性を抑えることができる．音速 c_s は現実世界のそれと同じ値 (空気の場合で約 340 [m/s]) を用いてもよいが，安定したシミュレーションを行うためのタイムステップ幅がかなり制限されるため，実際にはもう一回り小さな値を設定する．タイムステップ幅の上限は Courant-Friedrichs-Lewy(CFL) 条件より，

$$\Delta t = \min_i \left(0.25 \min_i \left(\frac{h_i}{|\boldsymbol{f}_i|} \right), 0.4 \frac{h_i}{c_s(1 + 0.6\alpha)} \right) \tag{3.18}$$

で求まる．α は定数で 0.05〜0.8 の間に設定する．本研究では 3.3.2 節で述べる方法で圧力計算を行うため，式 (3.18) の制限は受けない．

3.2.3 カーネル関数

式 (3.4) から式 (3.12) で用いられているカーネル関数は，近傍粒子からどの程度影響を受けるかを決める重みの役割を果たす．関数本体は粒子間距離と有効半径に依存し，粒子間距離が離れるほど影響が弱くなる，ガウス関数のような形となっている．また，式 (3.4) を導出するときに，積分すると 1 となるように正規化されていることが前提となっている．カーネル関数は多くの研究者によって様々な形のものが提案されてきたが，本研究では，文献 [17] の cubic spline kernel と，文献 [19] の viscosity kernel を用いる．それぞれの文献におけるカーネル関数の用途に則り，前者は通常の重みおよび勾配のときに適用し，後者はラプラシアンのときに適用する．以下に cubic spline kernel およびその勾配を示す．

$$W(\boldsymbol{x}_{ij},h)=\alpha_1\begin{cases}1-\frac{3}{2}\left(\frac{|\boldsymbol{x}_{ij}|}{h}\right)^2+\frac{3}{4}\left(\frac{|\boldsymbol{x}_{ij}|}{h}\right)^3 & 0\le\frac{|\boldsymbol{x}_{ij}|}{h}<1\\ \frac{1}{4}\left(2-\frac{|\boldsymbol{x}_{ij}|}{h}\right)^3 & 1\le\frac{|\boldsymbol{x}_{ij}|}{h}<2\\ 0 & othewise\end{cases}\tag{3.19}$$

$$\nabla W(\boldsymbol{x}_{ij},h)=\alpha_2\begin{cases}\left(\frac{|\boldsymbol{x}_{ij}|}{h}-\frac{4}{3}\right)\boldsymbol{x}_{ij} & 0\le\frac{|\boldsymbol{x}_{ij}|}{h}<1\\ -\frac{1}{3}\left(2-\frac{|\boldsymbol{x}_{ij}|}{h}\right)^2\frac{h}{|\boldsymbol{x}_{ij}|}\boldsymbol{x}_{ij} & 1\le\frac{|\boldsymbol{x}_{ij}|}{h}<2\\ 0 & othewise\end{cases}\tag{3.20}$$

ここで，α_1 は 2 次元シミュレーションのとき $\dfrac{10}{7\pi h^2}$，3 次元シミュレーションのとき $\dfrac{1}{\pi h^3}$，α_2 は 2 次元シミュレーションのとき $\dfrac{45}{14\pi h^4}$，3 次元シミュレーションのとき $\dfrac{9}{4\pi h^5}$ である．

viscosity kernel のラプラシアンは，

$$\nabla^2 W(\boldsymbol{x}_{ij},h)=\alpha_3\begin{cases}(h-|\boldsymbol{x}_{ij}|) & 0\le\frac{|\boldsymbol{x}_{ij}|}{h}<1\\ 0 & othewise\end{cases}\tag{3.21}$$

で表され，α_3 は 2 次元シミュレーションのとき $\dfrac{20}{3\pi h^5}$，3 次元シミュレーションのとき $\dfrac{45}{\pi h^6}$ である．

また，式 (3.19) - 式 (3.21) からわかるように，有効半径 h はどこまでの範囲にある粒子と相互作用するかを決めるのに用いられている．有効半径はユーザが自由に設定しても良いが，計算に用いたい近傍粒子の数を用いて，

$$h_i=\begin{cases}\left(\frac{N_H}{\pi}\frac{m_i}{\rho_i}\right)^{\frac{1}{2}} & for\,2D\\ \left(\frac{3N_H}{4\pi}\frac{m_i}{\rho_i}\right)^{\frac{1}{3}} & for\,3D\end{cases}\tag{3.22}$$

で求めることもできる．N_H は相互作用計算を行う近傍粒子の個数であり，N_H に応じて有効半径の大きさが決まる．h_i が大きいほど計算精度，安定性は高くなるが，探索すべき近傍粒子数が増えるため，計算量が大きくなる．本研究では $N_H=26$ を用いている．

3.2.4 近傍粒子探索

相互作用計算を行う近傍粒子は，シミュレーション空間に存在する全ての粒子との粒子間距離を求め，その距離が有効半径以下かどうかで判別できる．しかし，このような粒子を全探索する方法は，全粒子数の 2 乗オーダーの計算量となり，計算効率が非常に悪くなる．そのため，近傍粒子を効率的に探索する手法が提案されてきた．本研究では文献 [30] を基に，図 3.2 のようにシミュレーション空間を格子で等分割し，自身が存在する格子内およびそれに隣接する格子内の粒子のみを探索する手法を用いることで，探索の効率化を図る．

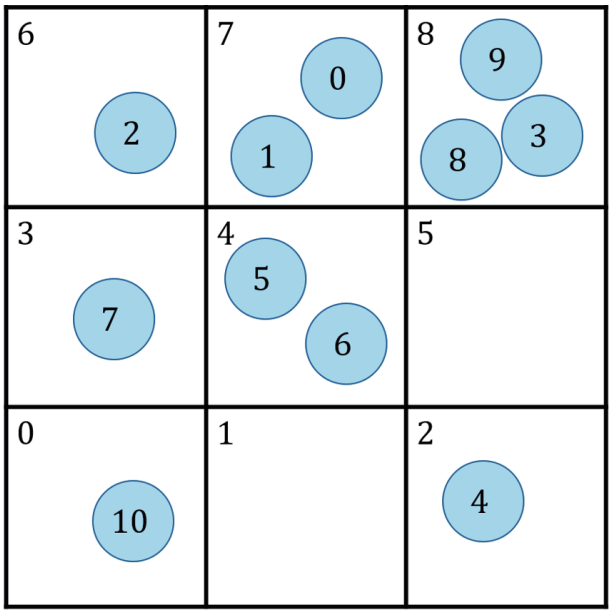


図 3.2: 等間隔格子を用いた近傍粒子探索法

表 3.1: 図 3.2 における近傍粒子登録の結果

head	0	1	2	3	4	5	6	7	8
値	10	-1	4	7	5	-1	2	0	3

last	0	1	2	3	4	5	6	7	8
値	10	-1	4	7	6	-1	2	1	9

next	0	1	2	3	4	5	6	7	8	9	10
値	1	-1	-1	8	-1	6	-1	-1	9	-1	-1

効率的な探索を行うための準備として，まずは Algorithm 2 に示す近傍粒子登録を行う．近傍粒子登録は，相互作用計算を行う前に，各格子内に存在する粒子のリストを事前に作成する作業である．各粒子がどの格子に存在するかは，粒子位置から求めたインデックス値 id が各格子に割り振られた格子番号と一致するかによって判定される．インデックス値は，x 方向の格子分割数 N_x および y 方向の格子分割数 N_y を用いて，

$$id = iz * N_y * N_x + iy * N_x + ix \tag{3.23}$$

Algorithm 2 近傍粒子登録

```
for all particle i do
  id の計算：式 (3.23) および式 (3.24)
  tmp = last[id]
  last[id] = i
  if tmp == −1 then
    head[id] = i
  else
    next[tmp] = i
  end if
end for
```

で求める．インデックス値 id は格子ごとにユニークなハッシュ値にもなっている．ix，iy，iz は，粒子位置 $\boldsymbol{x}_i = (x, y, z)$ とシミュレーション空間における座標の最小値 $\boldsymbol{x}_{min}= (x_{min}, y_{min}, z_{min})$ を用いて，

$$\begin{aligned} ix &= \frac{(x - x_{min})}{dx} \\ iy &= \frac{(y - y_{min})}{dx} \\ iz &= \frac{(z - z_{min})}{dx} \end{aligned} \tag{3.24}$$

で求める．ここで，dx は格子幅である．これを用いて，各格子に存在する粒子のリストを作成する．リストの作成には，各格子における先頭の粒子番号を格納する配列 $head$，最後の粒子番号を格納する配列 $last$，同じ格子内の次の粒子番号を格納する配列 $next$ を用いる．$head$ と $last$ は全格子数，$next$ は全粒子数の分だけメモリを確保し，全ての要素を -1 で初期化する．リストを作成する手順として，まずは粒子 i の id を求め，一時的に値を格納するための適当な変数 tmp に $last[id]$ を代入 $(tmp = last[id])$ し，$last[id]$ に粒子番号 i を代入する $(last[id] = i)$．その後，tmp の値を確認し，$tmp == -1$ の場合は $head[id] = i$ とし，$tmp \neq -1$ であれば $next[tmp] = i$ とする．これを全粒子で行うことで，近傍粒子登録が完了する．例として，図 3.2 のシーンにおける近傍粒子登録の結果を表 3.1 に示す．

作成されたリストを用いた近傍探索探索は，Algorithm 3 に示す手順で行う．始めに自身の存在する格子のインデックスおよび隣接する格子のインデックスを求める．探索する格子 id において，先頭の粒子番号 $head[id]$ を近傍粒子 j とし，距離計算によって相互作用計算を行うか判定する．近傍粒子番号は $next[j]$ で更新し，$j == -1$ ならばその格子内での探索を終了する．

粒子位置は毎ステップ更新されるため，近傍粒子登録も同様に毎ステップ行わなければならないが，その計算量は全粒子数の 1 乗オーダーとなるため，全探索よりも計算効率が大幅に改善される．そのため，シミュレーションに用いられる粒子数が多いほどその効果が発揮される．

Algorithm 3 近傍粒子探索

```
for all particle i do
  for idx = ix − 1 to ix + 1 do
    for idy = iy − 1 to iy + 1 do
      for idz = iz − 1 to iz + 1 do
        id = idz ∗ N_y ∗ N_x + idy ∗ N_x + idx
        j = head[id]
        while j ≠ −1 do
          if |x_ij| < h then
            粒子 j との相互作用計算
          end if
          j = next[j]
        end while
      end for
    end for
  end for
end for
```

3.3 SPH 法の拡張

3.3.1 MultiSPH

多相流体シミュレーションでは，シミュレートする流体によって密度，質量，粘性係数などのパラメータを粒子によって異なる値に設定する場合がほとんどである．このような状況下で従来の SPH 法の式を用いてシミュレーションを行った場合，2 つの流体の境界付近で性質の異なる流体からの影響も考慮してしまうため，密度計算を正確に行うことができない．このときの境界付近の密度分布を，境界に垂直な直線上の一次元分布として表すと，図 3.3 の中段のように滑らかに変化してしまう．現実には，図 3.3 上段のように分布しており，正確な相互作用計算を行うためには，密度が滑らかに変化せず，境界付近で不連続に変化することが望ましい．そこで，文献 [23] では粒子密度 δ というパラメータを導入することでこの問題に対応した．粒子密度は，式 (3.25) に示すように近傍粒子との重み計算の総和で表され，近傍の粒子分布と粒子の大きさのみに依存し，その他の質量などに依存しないパラメータである．この粒子密度を用いて，新たに式 (3.26) のように密度計算式を定義した．

$$\delta_i = \sum_j W(\boldsymbol{x}_{ij}, h) \tag{3.25}$$

$$\rho_i = m_i \delta_i \tag{3.26}$$

式 (3.7) と比較して、m_j が m_i に置き換わっていることがわかる．このように，近傍の粒子は自身と同じ性質であるとみなすことで，図 3.3 下段のように境界付近で不連続な密度分布を生成することができる．この粒子密度を用いた SPH 法を本研究では MultiSPH と呼ぶ．圧力勾配項および粘性拡散項も同様に，粒子密度を用いて

$$\boldsymbol{a}_i^p = -\frac{1}{m_i} \sum_j \left(\frac{p_i}{\delta_i^2} + \frac{p_j}{\delta_j^2} \right) \nabla W(\boldsymbol{x}_{ij}, h) \tag{3.27}$$

$$\boldsymbol{a}_i^v = \frac{1}{m_i \delta_i} \sum_j \frac{\mu_i + \mu_j}{2} \frac{\boldsymbol{v}_j - \boldsymbol{v}_i}{\delta_j} \nabla^2 W(\boldsymbol{x}_{ij}, h) \tag{3.28}$$

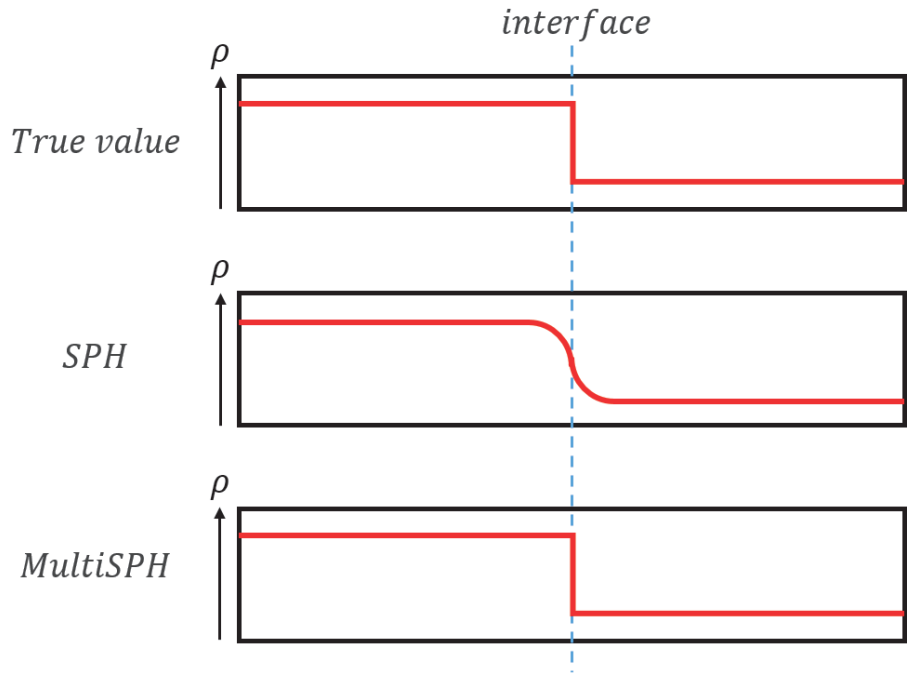


図 3.3: 各手法における密度分布の比較

と定義される．式 (3.12) および式 (3.10) と比較すると，近傍粒子の密度および質量が式から取り除かれており，近傍粒子の性質に依存しない形となっている．また，文献 [23] において圧力計算は式 (3.17) の Tait 方程式を用いている．MultiSPH を用いた多相流体シミュレーションの結果が図 2.3 の右であり，従来の SPH 法により得られた左の結果と比較すると，流体境界付近の粒子分布が均一となっている．これは，性質の異なる流体が複数存在するような状況においても，境界付近で正確な相互作用計算ができていることを表している．

3.3.2 IISPH

1.1 節で述べたように，従来の SPH 法は非圧縮性を考慮していないため，非圧縮性流体をシミュレーションするための SPH 法が提案されてきた．本研究ではそれらの中で Ihmsen らの Implicit Incompressible SPH(IISPH) [11] を採用した．

IISPH は，式 (3.2) に示した連続方程式を用いて圧力のポアソン方程式を新たに定義し，反復法により各粒子の圧力値を求める手法である．式 (3.2) の左辺を前進差分，右辺を SPH 法で離散化すると，

$$\frac{\rho_i(t+\Delta t)-\rho_i(t)}{\Delta t}=\sum_j m_j(\boldsymbol{v}_i(t+\Delta t)-\boldsymbol{v}_j(t+\Delta t))\cdot\nabla W(\boldsymbol{x}_{ij},h) \tag{3.29}$$

のような式が得られる．ここで，式 (3.2) の $\nabla\cdot\boldsymbol{v}$ は，

$$\nabla\cdot\boldsymbol{v}_i=-\frac{1}{\rho_i}\sum_j m_j(\boldsymbol{v}_i-\boldsymbol{v}_j)\cdot\nabla W(\boldsymbol{x}_{ij},h)$$

と離散化している．次に，圧力勾配項以外からの影響を考慮した Δt 秒後の密度 (中間密度)ρ^* を求めるため，式 (3.29) を

$$\rho_i^*=\rho_i(t)+\Delta t\sum_j m_j(\boldsymbol{v}_i^*-\boldsymbol{v}_j^*)\cdot\nabla W(\boldsymbol{x}_{ij},h) \tag{3.30}$$

のように書き換える．ここで，$\boldsymbol{v}^*$ は中間速度であり，

$$\boldsymbol{v}_i^* = \boldsymbol{v}_i(t) + (\boldsymbol{a}_i^v + \boldsymbol{a}_i^s - \boldsymbol{g})\Delta t$$

で求める．式 (3.30) で得られた中間密度にさらに圧力勾配項の影響を加える場合，式 (3.29) は

$$\rho_i(t+\Delta t) = \rho_i^* + \Delta t \sum_j m_j(\boldsymbol{v}_i^p - \boldsymbol{v}_j^p)\cdot\nabla W(\boldsymbol{x}_{ij}, h) \tag{3.31}$$

のように書き換えられる．ここで，Δt 秒後も密度が初期密度 ρ_{0i} に保たれるとすると，$\rho_i(t+\Delta t) = \rho_{0i}$ とでき，また $\boldsymbol{v}^p = \boldsymbol{a}^p\Delta t$ とおくと，最終的に以下のような圧力のポアソン方程式が得られる．

$$\Delta t^2 \sum_j m_j(\boldsymbol{a}_i^p - \boldsymbol{a}_j^p)\cdot\nabla W(\boldsymbol{x}_{ij}, h) = \rho_{0i} - \rho_i^* \tag{3.32}$$

ここから反復計算によって粒子の圧力を求めるために，式 (3.32) を展開して圧力 p_i を求める形に変形する．$\Delta t^2\boldsymbol{a}_i^p$ は式 (3.12) を用いて，

$$\begin{aligned}
\Delta t^2\boldsymbol{a}_i^p &= -\Delta t^2 \sum_j m_j\left(\frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2}\right)\nabla W(\boldsymbol{x}_{ij}, h) \\
&= \underbrace{\left(-\Delta t^2 \sum_j \frac{m_j}{\rho_i^2}\nabla W(\boldsymbol{x}_{ij}, h)\right)}_{\boldsymbol{d}_{ii}} p_i + \sum_j \underbrace{\left(-\Delta t^2 \frac{m_j}{\rho_j^2}\nabla W(\boldsymbol{x}_{ij}, h)\right)}_{\boldsymbol{d}_{ij}} p_j
\end{aligned} \tag{3.33}$$

のように展開される．ここで式中にもあるように，右辺第一項，第二項をそれぞれ $\boldsymbol{d}_{ii}$，$\boldsymbol{d}_{ij}$ とする．$\Delta t^2\boldsymbol{a}_j^p$ も同様の手順で展開することで，

$$\sum_j m_j(\boldsymbol{d}_{ii}p_i + \sum_j \boldsymbol{d}_{ij}p_j - \boldsymbol{d}_{jj}p_j - \sum_k \boldsymbol{d}_{jk}p_k)\cdot\nabla W(\boldsymbol{x}_{ij}, h) = \rho_{0i} - \rho_i^* \tag{3.34}$$

という式が得られる．ここで，$\sum_k \boldsymbol{d}_{jk}p_k$ には p_i に関する項が含まれているため，

$$\sum_k \boldsymbol{d}_{jk}p_k = \sum_{k\neq i} \boldsymbol{d}_{jk}p_k^l + \boldsymbol{d}_{ji}p_i \tag{3.35}$$

のように分解し，p_i が含まれる項を取り出す．これより式 (3.34) の p_i に関する項を左辺に，それ以外を右辺にまとめると

$$\begin{aligned}
&\underbrace{\left(\sum_j m_j(\boldsymbol{d}_{ii} - \boldsymbol{d}_{ji})\cdot\nabla W(\boldsymbol{x}_{ij}, h)\right)}_{a_{ii}} p_i = \\
&\rho_{0i} - \rho_i^* - \sum_j m_j\left(\sum_j \boldsymbol{d}_{ij}p_j^l - \boldsymbol{d}_{jj}p_j^l - \sum_{k\neq i}\boldsymbol{d}_{jk}p_k^l\right)\cdot\nabla W(\boldsymbol{x}_{ij}, h)
\end{aligned} \tag{3.36}$$

となる．ここで，p_i に掛かっている係数を a_{ii} とする．これより，最終的に圧力を求める式は，

$$p_i^{l+1} = (1-\omega)p_i^l + \omega\frac{1}{a_{ii}}\left(\rho_{0i} - \rho_i^* - \sum_j m_j\left(\sum_j \boldsymbol{d}_{ij}p_j^l - \boldsymbol{d}_{jj}p_j^l - \sum_{k\neq i}\boldsymbol{d}_{jk}p_k^l\right)\cdot\nabla W(\boldsymbol{x}_{ij}, h)\right) \tag{3.37}$$

となる．ここで，l は現在の反復回数，ω は緩和係数 $(= 0.5)$ である．反復計算には緩和ヤコビ法を用いており，一つ前の反復計算結果からの影響も考慮することで，収束が早くなる．反復計算は，式 (3.32) の左辺に中間密度を移項して求めた値と，初期密度との誤差を求め，全粒子の平均密度誤差が閾値 ϵ 内に収まった場合に終了させる．最終的に求めた圧力値を用いて圧力勾配項の計算を行い，粒子の速度を式 (3.38) で更新する．

$$\boldsymbol{v}_i(t + \Delta t) = \boldsymbol{v}_i^* + \boldsymbol{a}_i^p \Delta t \tag{3.38}$$

IISPH を用いるメリットとしては，実装が容易であること，収束が早いこと，タイムステップ幅を大きく設定できること，近傍粒子登録を再度行う必要が無いことなどが挙げられる．文献 [22, 5] では，反復毎に Δt 秒後 (1 ステップ後) の予測位置を求め，再び近傍粒子登録を行って 1 ステップ後の密度を求めてから終了判定を行うため，多くの手順を要する．一方，IISPH はポアソン方程式から密度誤差を求める過程において粒子の移動を伴わないため，そのような手順を踏まずに効率よく圧力計算を行うことができる．

3.4 提案法

3.3 節では，多相流体シミュレーション向けの MultiSPH および非圧縮性を確保する IISPH について説明した．それぞれ多相流体シミュレーションにおける正確な相互作用と非圧縮性という性質を満たしてはいるが，両方を満たす粒子法のための手法はこれまで存在していない．本研究では，MultiSPH および IISPH の考え方をもとに，粒子法で多相流体シミュレーションを可能とする非圧縮性 SPH 法を新たに提案し，提案法を MultiIISPH と呼ぶ．手法としては，IISPH のポアソン方程式を，MultiSPH で定義された粒子密度をベースとした式に修正することで，両方の性質を満たすようにしている．提案法による圧力計算の手順を Algorithm 4 に示す．

3.4.1 MultiIISPH による圧力計算

まずは，式 (3.30) の中間密度を求める式に関して，$m_j = m_i$ と置き換える．

$$\rho_i^* = \rho_i(t) + m_i \Delta t \sum_j (\boldsymbol{v}_i^* - \boldsymbol{v}_j^*) \cdot \nabla W(\boldsymbol{x}_{ij}, h) \tag{3.39}$$

この操作は，MultiSPH における密度計算のときと同様に，近傍の粒子が自分と同じ性質の流体であるとみなすことを示している．粒子密度は式 (3.26) より，$\delta = \frac{\rho}{m}$ で求められることから，式 (3.39) の両辺を m_i で割ることにより，

$$\delta_i^* = \delta_i(t) + \Delta t \sum_j (\boldsymbol{v}_i^* - \boldsymbol{v}_j^*) \cdot \nabla W(\boldsymbol{x}_{ij}, h) \tag{3.40}$$

が得られる．ここで，δ_i^* は中間粒子密度である．式 (3.32) のポアソン方程式も，同様の操作を行うことで，

$$\Delta t^2 \sum_j (\boldsymbol{a}_i^p - \boldsymbol{a}_j^p) \cdot \nabla W(\boldsymbol{x}_{ij}, h) = \delta_{0i} - \delta_i^* \tag{3.41}$$

Algorithm 4 提案法による圧力計算

-反復計算の前準備-
for all fluid particle i **do**
$\tilde{\boldsymbol{d}}_{ii}$ の計算：式 (3.42)
中間粒子密度 δ^* の計算：式 (3.49)
初期化 ($p_i^0 = 0.5 p_i(t - \Delta t)$)
$\hat{a}_{ii}$ の計算：式 (3.52)
end for
$l = 0$
-反復計算-
while average density error $< \epsilon$ **or** $l < 2$ **do**
for all fluid particle i **do**
$\sum_j \tilde{\boldsymbol{d}}_{ij} p_j^l$ の計算：式 (3.42)
end for
for all fluid particle i **do**
p_i^{l+1} の計算：式 (3.51)
$p_i = p_i^{l+1}$
end for
$l \leftarrow l + 1$
end while

と置き換えられる．ここで，δ_0 は初期粒子密度である．$\Delta t^2 \boldsymbol{a}^p$ は式 (3.27) を用いて，

$$\begin{aligned} \Delta t^2 \boldsymbol{a}_i^p &= -\frac{\Delta t^2}{m_i} \sum_j \left(\frac{p_i}{\delta_i^2} + \frac{p_j}{\delta_j^2} \right) \nabla W(\boldsymbol{x}_{ij}, h) \\ &= \underbrace{\left(-\frac{\Delta t^2}{m_i \delta_i^2} \sum_j \nabla W(\boldsymbol{x}_{ij}, h) \right)}_{\tilde{\boldsymbol{d}}_{ii}} p_i + \sum_j \underbrace{\left(-\frac{\Delta t^2}{m_i \delta_j^2} \nabla W(\boldsymbol{x}_{ij}, h) \right)}_{\tilde{\boldsymbol{d}}_{ij}} p_j \end{aligned} \tag{3.42}$$

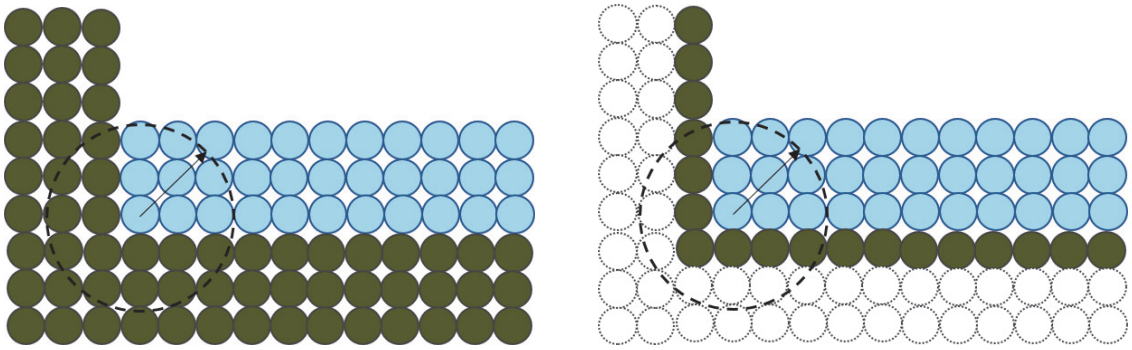
のように展開できる．式 (3.42) では $\boldsymbol{d}_{ii}$，$\boldsymbol{d}_{ij}$ の代わりに $\tilde{\boldsymbol{d}}_{ii}$，$\tilde{\boldsymbol{d}}_{ij}$ と表記する．最終的に圧力 p_i に関する式は，

$$p_i^{l+1} = (1 - \omega) p_i^l + \omega \frac{1}{\tilde{a}_{ii}} \left(\delta_{0i} - \delta_i^* - \sum_j \left(\sum_j \tilde{\boldsymbol{d}}_{ij} p_j^l - \tilde{\boldsymbol{d}}_{jj} p_j^l - \sum_{k \neq i} \tilde{\boldsymbol{d}}_{jk} p_k^l \right) \cdot \nabla W(\boldsymbol{x}_{ij}, h) \right) \tag{3.43}$$

となる．ここで，

$$\tilde{a}_{ii} = \sum_j (\tilde{\boldsymbol{d}}_{ii} - \tilde{\boldsymbol{d}}_{ji}) \cdot \nabla W(\boldsymbol{x}_{ij}, h) \tag{3.44}$$

である．終了判定は IISPH と同様に，式 (3.41) の左辺に中間粒子密度を足したものと，初期粒子密度との誤差を求め，全粒子の平均密度誤差が閾値 ϵ 内に収まった場合とする．いずれの式においても，近傍粒子の密度や質量といったパラメータが除かれ，近傍粒子の性質に依存しない粒子密度ベースとなっていることがわかる．以上の修正式より MultiIISPH を用いることで，性質の異なる流体が複数存在するような状況においても，正確な密度計算により全ての流体を同時にシミュレートすることが可能となり，かつ非圧縮性を確保することができる．



(a) 複数層配置した場合　(b) 仮想体積により複数層配置を 1 層で近似した場合

図 3.4: 固体境界粒子の配置

3.4.2　固体境界粒子との相互作用

ここまでは流体同士の相互作用計算について説明してきたが，様々なシーンを再現するためには，固体との相互作用計算も考える必要がある．粒子法では固体との相互作用を考慮する際，固体表面に境界粒子を配置する方法が一般的である．この場合，流体粒子の計算を正確に行うために，図 3.4(a) のように境界粒子を複数層配置する必要があるため，粒子数が増えてしまい計算量の増加につながってしまう．これを改善するために，Akinci ら [3] は境界粒子に対して仮想体積を導入し，図 3.4(b) のように境界粒子 1 層でも複数層と変わらないシミュレーションを可能とする手法を提案した．仮想体積 V_b は近傍の境界粒子を用いて，

$$V_b = \frac{1}{\sum_b W(\boldsymbol{x}_{ib}, h)} = \frac{1}{\delta_b} \tag{3.45}$$

で表される．添え字 b は境界粒子を表している．式からわかるように，仮想体積は粒子密度の逆数となっているため，本研究における粒子密度ベースの手法と相性が良い．相互作用計算において境界粒子からの影響を考慮する場合，仮想体積に ρ_0 をかけたパラメータ

$$\Psi_b(\rho_{0i}) = \rho_{0i} V_b \tag{3.46}$$

を用いる．これにより，仮想体積の大きさによって境界粒子からの影響の大きさを変えることができる．このパラメータを考慮した密度計算は，

$$\rho_i = m_i \sum_j W(\boldsymbol{x}_{ij}, h) + \sum_b \Psi_b(\rho_{0i}) W(\boldsymbol{x}_{ib}, h) \tag{3.47}$$

と表される．また，MultiIISPH の計算式において，固体境界粒子が動かないものとして考えた場合 ($p_b = 0, \boldsymbol{v}_b^* = 0, \boldsymbol{a}_b^p = 0$)，式 (3.39) は

$$\begin{aligned} \rho_i^* = \rho_i(t) \quad &+ \quad m_i \Delta t \sum_j (\boldsymbol{v}_i^* - \boldsymbol{v}_j^*) \cdot \nabla W(\boldsymbol{x}_{ij}, h) \\ &+ \quad \Delta t \sum_b \Psi_b(\rho_{0i}) \boldsymbol{v}_i^* \cdot \nabla W(\boldsymbol{x}_{ib}, h) \end{aligned} \tag{3.48}$$

のように書き換えられ，結果として中間粒子密度の式は

$$\begin{aligned}\delta_i^* = \delta_i(t) \quad &+ \quad \Delta t \sum_j (\boldsymbol{v}_i^* - \boldsymbol{v}_j^*) \cdot \nabla W(\boldsymbol{x}_{ij}, h) \\ &+ \quad \Delta t \sum_b \frac{\Psi_b(\rho_{0i})}{m_i} \boldsymbol{v}_i^* \cdot \nabla W(\boldsymbol{x}_{ib}, h)\end{aligned} \tag{3.49}$$

となる．式 (3.41) のポアソン方程式も同様に，

$$\Delta t^2 \sum_j (\boldsymbol{a}_i^p - \boldsymbol{a}_j^p) \cdot \nabla W(\boldsymbol{x}_{ij}, h) + \Delta t^2 \sum_b \frac{\Psi_b(\rho_{0i})}{m_i} \boldsymbol{a}_i^p \cdot \nabla W(\boldsymbol{x}_{ib}, h) = \delta_{0i} - \delta_i^* \tag{3.50}$$

と書き換えられる．最終的に圧力 p_i に関する式は，

$$\begin{aligned}p_i^{l+1} =& (1-\omega) p_i^l + \omega \frac{1}{\hat{a}_{ii}} \left(\delta_{0i} - \delta_i^* - \sum_j \left(\sum_j \tilde{\boldsymbol{d}}_{ij} p_j^l - \tilde{\boldsymbol{d}}_{jj} p_j^l - \sum_{k \neq i} \tilde{\boldsymbol{d}}_{jk} p_k^l \right) \cdot \nabla W(\boldsymbol{x}_{ij}, h) \right. \\ &\left. - \sum_b \frac{\Psi_b(\rho_{0i})}{m_i} \left(\sum_j \tilde{\boldsymbol{d}}_{ij} p_j^l \right) \cdot \nabla W(\boldsymbol{x}_{ib}, h) \right)\end{aligned} \tag{3.51}$$

となり，ここで，

$$\begin{aligned}\hat{a}_{ii} =& \sum_j (\tilde{\boldsymbol{d}}_{ii} - \tilde{\boldsymbol{d}}_{ji}) \cdot \nabla W(\boldsymbol{x}_{ij}, h) \\ &+ \sum_b \tilde{\boldsymbol{d}}_{ii} \cdot \nabla W(\boldsymbol{x}_{ib}, h)\end{aligned} \tag{3.52}$$

である．

3.5 表面張力計算

表面張力は，流体を構成する分子同士に働く力により，その表面積を最小化しようとする力のことを指す．SPH 法では最初，粒子分布から曲率を求めて計算する手法 [19] が用いられていたが，近年では粒子一つ一つを分子として見立て，粒子間の相互作用として表面張力を求める計算法 [25, 4, 2] が，安定した計算が可能であることからよく用いられる．本研究では，Akinci ら [2] の表面張力モデルをシミュレーションに導入する．なお，表面張力計算では同じ性質を持つ流体粒子間でのみ相互作用計算を行う．

文献 [2] の表面張力モデルは，分子間力項と表面積最小化項の 2 つで構成されている．分子間力項は，

$$\boldsymbol{f}_{ij}^c = -\gamma m_i m_j C(\boldsymbol{x}_{ij}, h) \frac{\boldsymbol{x}_{ij}}{|\boldsymbol{x}_{ij}|} \tag{3.53}$$

で表される．ここで，γ は表面張力係数，$C(\boldsymbol{x}_{ij}, h)$ はこのモデルにおけるスプライン関数であり，

$$C(\boldsymbol{x}_{ij}, h) = \alpha_c \begin{cases} (h - |\boldsymbol{x}_{ij}|)^3 |\boldsymbol{x}_{ij}|^3 & 0.5 \leq \frac{|\boldsymbol{x}_{ij}|}{h} < 1 \\ 2(h - |\boldsymbol{x}_{ij}|)^3 |\boldsymbol{x}_{ij}|^3 - \frac{h^6}{64} & 0 \leq \frac{|\boldsymbol{x}_{ij}|}{h} < 0.5 \\ 0 & otherwise \end{cases} \tag{3.54}$$

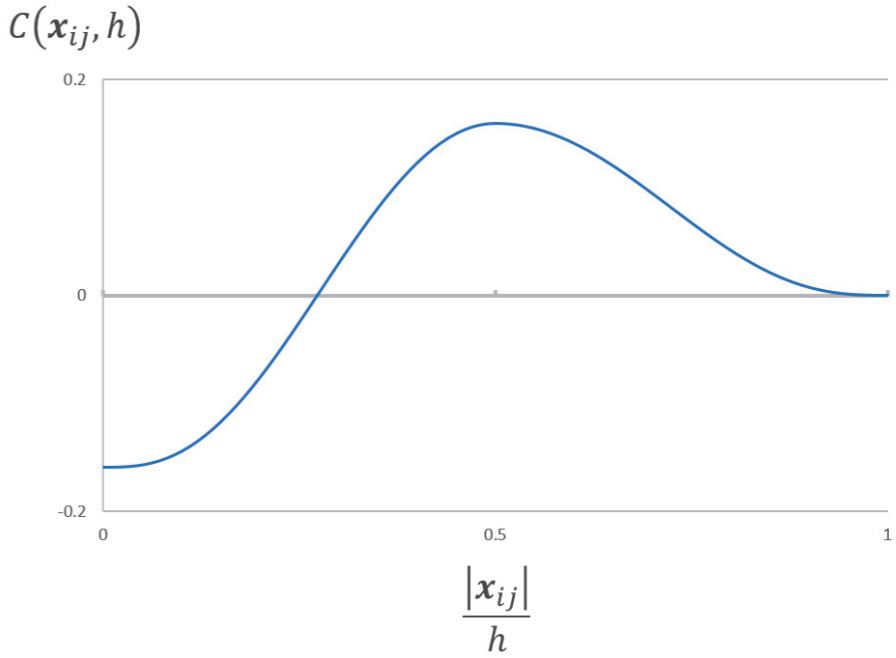


図 3.5: スプライン関数 $C(\boldsymbol{x}_{ij}, h)$ の分布 $(h = 1)$

という形となっている．α_c は，2 次元シミュレーションの場合 $\dfrac{35840}{209\pi h^8}$，3 次元シミュレーションの場合 $\dfrac{32}{\pi h^9}$ である．このスプライン関数 $C(\boldsymbol{x}_{ij}, h)$ は，図 3.5 に示すように粒子間距離が近い場合にマイナスの値が得られるように設計されている．これは，表面積を最小化させる計算の過程で，表面付近で粒子が密集してしまう現象を防ぐために，距離が近い粒子に斥力を働かせることで，一定の粒子間距離を保つという役割のためである．また表面積最小化項は，曲率を直接求める代わりに法線ベクトル $\boldsymbol{n}$ を用いて，

$$\boldsymbol{f}_{ij}^{a} = -\gamma m_i(\boldsymbol{n}_i - \boldsymbol{n}_j) \tag{3.55}$$

という形で表面積を最小化する力を計算する．粒子の法線ベクトルは，

$$\boldsymbol{n}_i = h_i \sum_j \frac{m_j}{\rho_j} \nabla W(\boldsymbol{x}_{ij}, h) \tag{3.56}$$

で計算される．式 (3.55) では，法線ベクトルの大きさと曲率が比例関係にあることを利用しており，本来複雑な曲率計算が必要なところ，簡単な式で表すことができている．

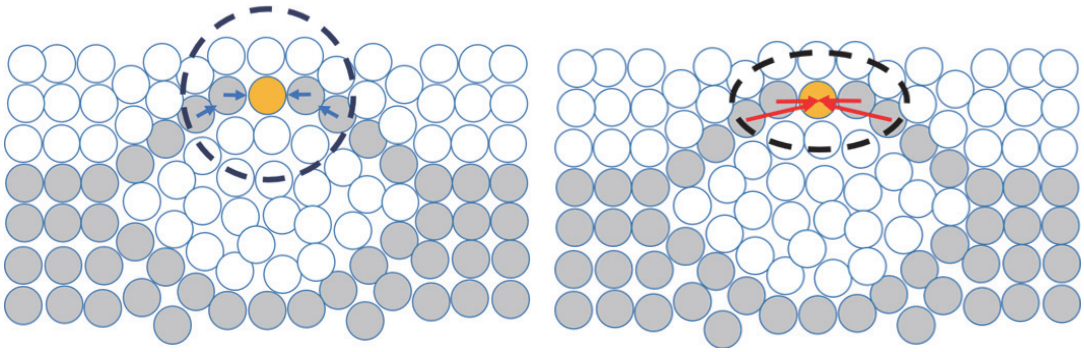
これら 2 つの項を計算したのち，作用の対称性を確保するための係数 $K_{ij} = \dfrac{2\rho_{0i}}{\rho_i + \rho_j}$ を用いて，近傍粒子からの粒子間力

$$\boldsymbol{f}_{ij}^{s} = K_{ij}(\boldsymbol{f}_{ij}^{c} + \boldsymbol{f}_{ij}^{a}) \tag{3.57}$$

を計算する．これにより，最終的な表面張力項は，

$$\boldsymbol{a}_i^{s} = \frac{1}{m_i} \sum_j \boldsymbol{f}_{ij}^{s} \tag{3.58}$$

で計算される．



(a) 異方性フィルタ未使用　　(b) 異方性フィルタ適用後

図 3.6: 異方性フィルタによる近傍粒子からの影響力の補正．矢印の長さは影響力の強さを示している．

Akinci らの表面張力モデルは，液体表面の粒子において高いパフォーマンスを発揮するが，薄い液膜のような，極端に粒子が少ない領域では正確に表面張力を計算することが難しくなる (図 3.6(a))．本研究では，Akinci らの表面張力モデルを直接修正せずに，計算結果に補正をかけることで，その課題に対応する．補正をかける方法として，Yang ら [27] の異方性フィルタを用いた手法を適用する．異方性フィルタは，近傍の粒子分布から求まる共分散行列から得られる，ベクトルを引き延ばすような変形行列である．これを用いることで，近傍粒子数が少ない分を，個々の影響力を強くすることで補うことができ，より正確に表面張力を表現することができる (図 3.6(b))．手法としては重み付き主成分分析 (WPCA) をベースにしており，以下の式のように近傍の粒子分布から粒子 i の重み付き平均位置 $\boldsymbol{x}_i^w$ を求め，それをもとに重み付き共分散行列 $\boldsymbol{C}_i$ を求める．

$$\boldsymbol{x}_i^w = \frac{\sum_j \boldsymbol{x}_j W^s(\boldsymbol{x}_{ij}, h)}{\sum_j W^s(\boldsymbol{x}_{ij}, h)} \tag{3.59}$$

$$\boldsymbol{C}_i = \frac{\sum_j (\boldsymbol{x}_j - \boldsymbol{x}_i^w)(\boldsymbol{x}_j - \boldsymbol{x}_i^w)^T W^s(\boldsymbol{x}_{ij}, h)}{\sum_j W^s(\boldsymbol{x}_{ij}, h)} \tag{3.60}$$

ここで，

$$W^s(\boldsymbol{x}_{ij}, h) = \begin{cases} 1 - \left(\frac{|\boldsymbol{x}_{ij}|}{h}\right)^3 & |\boldsymbol{x}_{ij}| < h \\ 0 & otherwise \end{cases} \tag{3.61}$$

である．ここから $\boldsymbol{C}_i$ を特異値分解することで，変形の方向を得る．

$$\boldsymbol{C}_i = \boldsymbol{R}\boldsymbol{\Sigma}\boldsymbol{R}^T \tag{3.62}$$

ここで，$\boldsymbol{R}$ は各列が固有ベクトルで構成された回転行列，$\boldsymbol{\Sigma}$ は対角成分に固有値 σ が格納された対角行列 $(= \mathrm{diag}(\sigma_1, \ldots, \sigma_d))$，$d$ は 2 次元シミュレーションなら 2，3 次元なら 3 である．ここからさらに $\boldsymbol{\Sigma}$ を修正し，

$$\tilde{\boldsymbol{\Sigma}} = \begin{cases} k_s \mathrm{diag}(\tilde{\sigma}_1, \ldots, \tilde{\sigma}_d) & N > N_\epsilon \\ k_n \boldsymbol{I} & othewise \end{cases} \tag{3.63}$$

を計算する．ここで，$\tilde{\sigma}_k = \max(\sigma_k, \frac{\sigma_1}{k_r})$，$N$ は近傍粒子数，N_ϵ は閾値であり，N がこれに満たない場合，対角行列を単位行列に設定する．k_s，k_n，k_r はスケーリング係数である．こ

れらのパラメータに関して，本研究では $N_\epsilon = 10$，$k_s = 4500.0$，$k_n = 0.5$，$k_r = 4.0$ としている．これより，修正された共分散行列

$$\tilde{\boldsymbol{C}}_i = \boldsymbol{R}\tilde{\boldsymbol{\Sigma}}\boldsymbol{R}^T \tag{3.64}$$

を求め，$\tilde{\boldsymbol{C}}$ の逆行列として異方性行列 $\boldsymbol{G}$ を求める．

$$\boldsymbol{G}_i = \frac{1}{h_i}\boldsymbol{R}\tilde{\boldsymbol{\Sigma}}^{-1}\boldsymbol{R}^T \tag{3.65}$$

最終的に，異方性フィルタ $\boldsymbol{T}$ は $\boldsymbol{G}$ を用いて，

$$\boldsymbol{T}_i = (1-\eta)\boldsymbol{I} + \eta\frac{\boldsymbol{G}_i}{|\boldsymbol{G}_i|} \tag{3.66}$$

と計算される．ここで，η は異方性行列の影響度を決定する係数であり，本研究では $\eta = 0.02$ としている．実際に異方性フィルタを表面張力に適用する場合，式 (3.58) の計算式に $\boldsymbol{T}$ を掛けるだけでよい．

$$\boldsymbol{a}_i^s = \frac{1}{m_i}\boldsymbol{T}_i\sum_j \boldsymbol{f}_{ij}^s \tag{3.67}$$

3.6 表面形状の生成

本研究において，一部シミュレーションでは粒子群を直接描画する代わりに，粒子分布から液体表面形状をポリゴンメッシュとして抽出してレンダリングを行っている．粒子分布から表面形状を抽出するために，本研究では，Yu ら [28] の異方性カーネルを用いた手法を導入する．従来の表面抽出法では，シミュレーション空間を格子分割し，格子上に陰関数場を生成することで表面形状を定義する．陰関数場 ϕ^s は，以下に示す関数を各格子位置において計算することで得られる．

$$\phi^s(\boldsymbol{x}) = \sum_j \frac{m_j}{\rho_j}W(\boldsymbol{x}-\boldsymbol{x}_j, h) \tag{3.68}$$

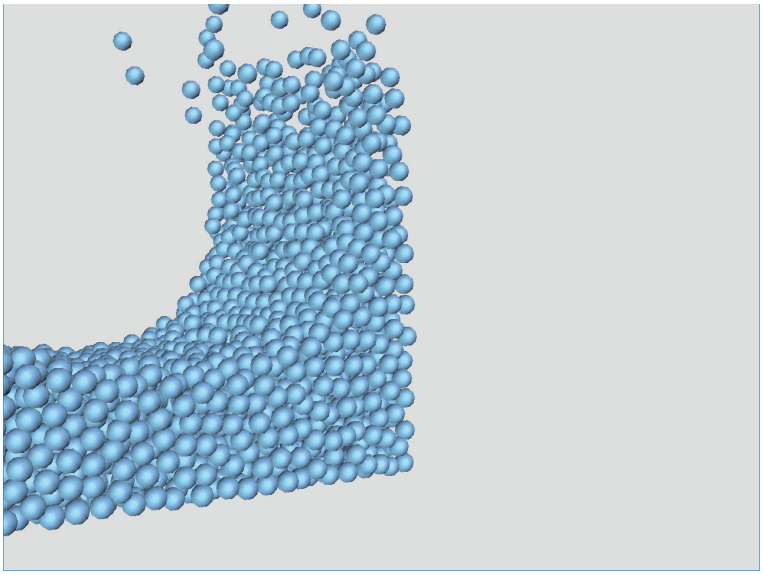
式 (3.68) で用いているカーネル関数は，式 (3.19) - 式 (3.21) のものと同様，有効半径 h 内までの粒子と相互作用する等方性のカーネルである．等方性カーネルは，図 3.7(a) に示すような不規則な粒子分布ではきれいな平面を抽出することが困難なため，ほとんどの場合表面形状は粒子の形状に沿ったものとなってしまい，図 3.7(b) のようにでこぼこした見た目となってしまう．文献 [28] では，この問題を改善するために，カーネル関数の探索範囲を引き延ばし，探索方向によって相互作用の範囲が異なる異方性のカーネルを用いることを提案した．異方性カーネルは，3.5 節の式 (3.65) で導出した異方性行列 $\boldsymbol{G}$ を用いて，

$$\phi_{new}^s(\boldsymbol{x}) = \sum_j \frac{m_j}{\rho_j}W(\boldsymbol{x}-\bar{\boldsymbol{x}}_j, |\boldsymbol{G}\boldsymbol{r}|) \tag{3.69}$$

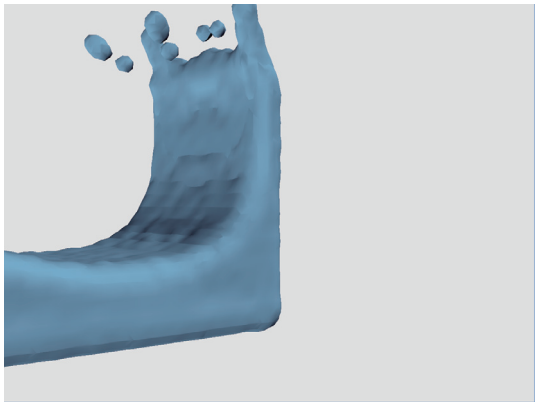
と表される．ここで，$\boldsymbol{r} = \boldsymbol{x} - \bar{\boldsymbol{x}}_j$ である．また，異方性カーネルに用いる $\boldsymbol{G}$ は $k_s = 1400.0$ で計算している．$\bar{\boldsymbol{x}}_j$ は，不規則な配置によるノイズを軽減した粒子の修正位置であり，

$$\bar{\boldsymbol{x}}_i = (1-\lambda)\boldsymbol{x}_i + \lambda\frac{\sum_j \boldsymbol{x}_j W^s(\boldsymbol{x}_{ij}, h)}{\sum_j W^s(\boldsymbol{x}_{ij}, h)} \tag{3.70}$$

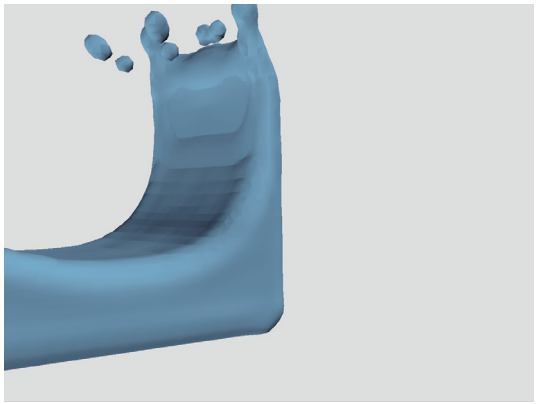
で計算される．λ は $0 \leq \lambda \leq 1$ の値を取るが，文献 [28] では $0.9 \leq \lambda \leq 1$ の範囲でより良い結果が得られると述べている．本研究では $\lambda = 0.9$ としている．実際に異方性カーネルを用いて表面形状を抽出すると，図 3.7(c) のようにきれいな結果を得ることができる．表面を描画するために，本研究では Marching Cubes 法 [15] を用いて，陰関数場がユーザが決めた閾値となる等値面を三角形メッシュとして抽出した．



(a) 表面抽出を行う粒子の分布



(b) 等方性カーネルを用いた表面生成



(c) 異方性カーネルを用いた表面生成

図 3.7: 表面形状の生成

第4章

結果

本章では，提案手法を用いて様々なシーンをシミュレートした結果を示す．4.1 節では，提案法を 3.3.1 節で説明した従来法と比較し，提案法の安定性を示す．4.2 節では，提案法を用いて性質の異なる複数の液体を同時にシミュレートした結果を示す．4.3 節では，液体と気体の相互作用として，様々な気泡の動きをシミュレートした結果を示す．本研究における全てのシミュレーションの実装は，表 4.1 に示す条件下で行った．表面メッシュを用いた結果については，POV-Ray[1] でレンダリングを行った．

表 4.1: シミュレーションの実装環境

CPU	Intel(R) Core(TM) i7-4770 3.40 GHz
メモリ	16.0 GB
OS	Microsoft Windows 10
使用言語	C++
グラフィックス用ライブラリ	OpenGL
数値計算ライブラリ	GSL (GNU Scientific Library)
並列計算ライブラリ	OpenMP

4.1 従来法との比較

実験内容

提案法の安定性を確かめるために，レイリー・テイラー不安定性の実験を行う．レイリー・テイラー不安定性は，密度の異なる流体の境界付近において，密度の大きい流体から密度の小さい流体に力が働くことで，運動が不安定になる現象である．代表的なシーンとして，空間の上部に密度の大きい流体，下部に密度の小さい流体を用意し，密度の小さい流体が空間上部に流れ込む現象を再現する．実験では 3.3.1 節で説明した Solenthaler と Pajarola[23] の手法を従来法とし，タイムステップ幅を変えて実験を行った上で提案法との安定性を比較する．実験で用いたパラメータを表 4.2 に示す．

実験結果

タイムステップ幅 $\Delta t = 0.3\,[\mathrm{ms}]$ としたときの，従来法による実験結果を図 4.1，提案法による実験結果を図 4.2 に示す．また，タイムステップ幅 $\Delta t = 3.0\,[\mathrm{ms}]$ としたときの，従来法による実験結果を図 4.3，提案法による実験結果を図 4.4 に示す．シミュレーションは $t = 3.0[s]$ まで行い，1 ステップ当たりの平均計算時間は表 4.3 の通りである．なお，シミュレーション空間の周囲は 4,704 個の固体境界粒子で囲まれており，結果の画像では全て非表示としている．

表 4.2: レイリー・テイラー不安定性の実験における各パラメータ

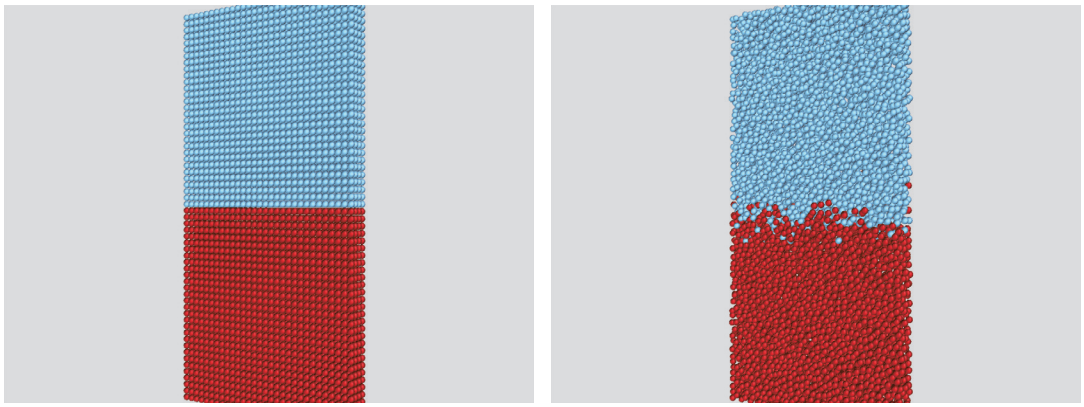
流体の色	水色	赤
流体粒子数	3600	3600
初期密度 $\rho_0\,[\mathrm{kg/m^3}]$	1000.0	250.0
表面張力係数 $\gamma\,[\mathrm{N/m}]$	1.0	5.0
粘性係数 $\mu\,[\mathrm{Pa\cdot s}]$	0.001	0.001
音速 $c_s\,[\mathrm{m/s}]$ $(\Delta t = 0.3\,[\mathrm{ms}])$	100.0	100.0
音速 $c_s\,[\mathrm{m/s}]$ $(\Delta t = 3.0\,[\mathrm{ms}])$	20.0	20.0

表 4.3: 1 ステップ当たりの平均計算時間

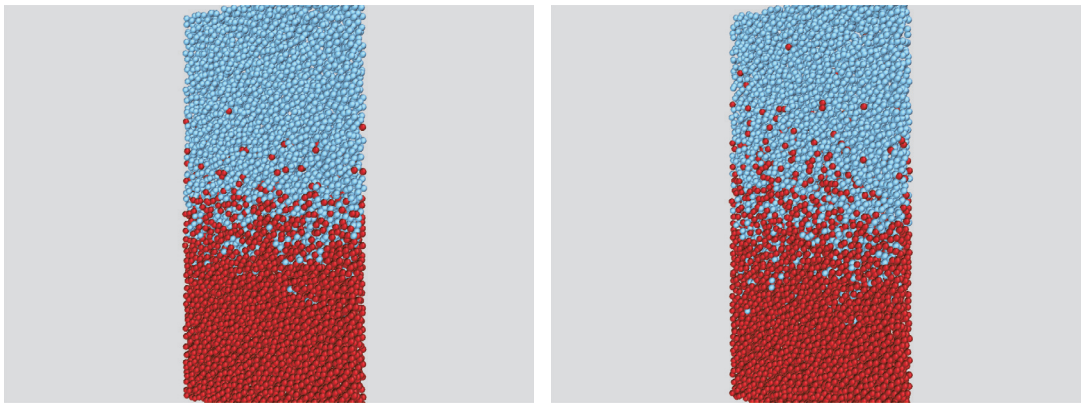
実験	計算時間 [s/step]
従来法 $\Delta t = 0.3\,[\mathrm{ms}]$	0.19
従来法 $\Delta t = 3.0\,[\mathrm{ms}]$	0.25
提案法 $\Delta t = 0.3\,[\mathrm{ms}]$	0.27
提案法 $\Delta t = 3.0\,[\mathrm{ms}]$	0.40

全ての実験結果において，密度の小さい流体が空間上部へ向かう挙動を確認した．一つ一つを細かく見ていくと，図 4.1 の従来法によるシミュレーションでは，見た目としては非圧縮性を保つことができているように見える．しかし，図 4.3 のようにタイムステップ幅を大きくした場合，特に図 4.3(b) に示す $t = 0.6\,[\mathrm{s}]$ のタイミングで全体が圧縮してしまっていることがわかる．表 4.2 に示すように，タイムステップ幅によって音速 c_s の値を変えており，c_s を小さくすると，式 (3.18) の CFL 条件によりある程度大きなタイムステップ幅を設定できる代わりに，圧縮性が高くなってしまう．c_s を変えた理由は，$\Delta t = 3.0\,[\mathrm{ms}]$ のときに $c_s = 100.0\,[\mathrm{m/s}]$ とした場合，粒子速度が大きな圧力を受けて発散してしまい，シミュレーション空間外に飛び出す現象を事前に確認したからである．一方で，図 4.2 および図 4.4 の提案法によるシミュレーションでは，タイムステップ幅の大きさに依らず，非圧縮性を保つことができているとわかる．

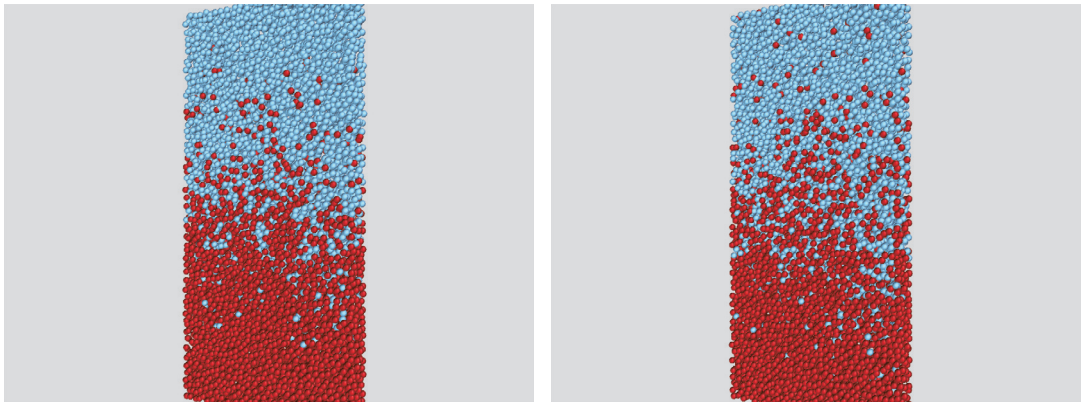
従来法と提案法で粒子分布に違いが見られる原因として，境界付近における圧力と表面張力との力のつり合いが異なるためであると考えられる．$\Delta t = 0.3\,[\mathrm{ms}]$ の場合，従来法では非圧縮性を保つための圧力による力が表面張力に勝っており，表面張力の影響がほとんどなくなり不規則な粒子分布となっている．反復法を用いた提案法ではそのような分布は見られず，同じ流体同士である程度まとまっていることから，表面張力の影響が確認できる．$\Delta t = 3.0\,[\mathrm{ms}]$ では，どちらの手法も表面張力により 2 つの流体がそれぞれできれいにまとまっていることが確認できる．結果の違いの原因としては，圧縮による近傍粒子数の増加が挙げられる．表 4.3 で示した Δt による計算時間の違いについては，従来法では圧縮による近傍粒子数の増加，提案法では反復回数の増加が関与していると考えられる．このように，タイムステップ幅の大きさに依らず非圧縮性を保ち，表面張力の影響も確認できるという結果から，提案法の安定性が確認できる．



(a) $t = 0\,[\mathrm{s}]$ (b) $t = 0.6\,[\mathrm{s}]$

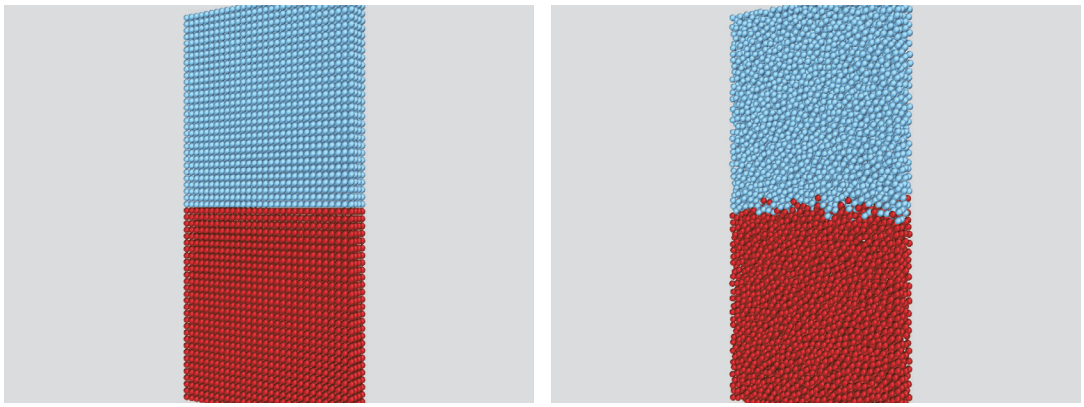


(c) $t = 1.2\,[\mathrm{s}]$ (d) $t = 1.8\,[\mathrm{s}]$

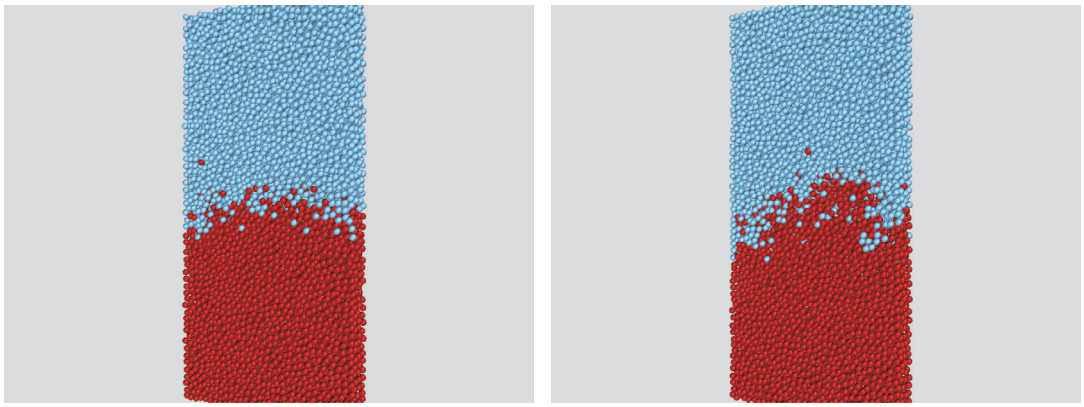


(e) $t = 2.4\,[\mathrm{s}]$ (f) $t = 3.0\,[\mathrm{s}]$

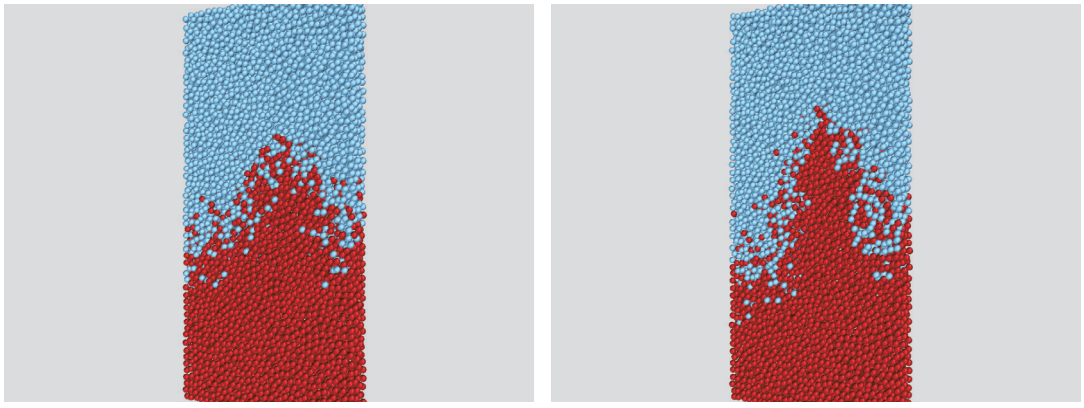
図 4.1: 従来法によるレイリー・テイラー不安定性の実験 ($\Delta t = 0.3\,[\mathrm{ms}]$)



(a) $t = 0\,[\mathrm{s}]$ (b) $t = 0.6\,[\mathrm{s}]$

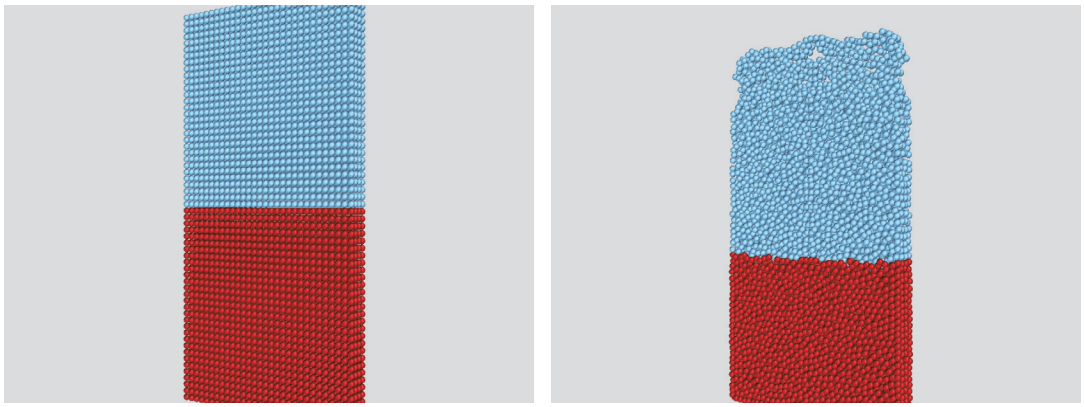


(c) $t = 1.2\,[\mathrm{s}]$ (d) $t = 1.8\,[\mathrm{s}]$



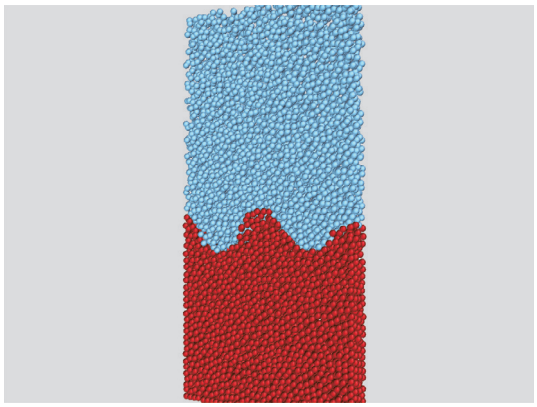
(e) $t = 2.4\,[\mathrm{s}]$ (f) $t = 3.0\,[\mathrm{s}]$

図 4.2: 提案法によるレイリー・テイラー不安定性の実験 ($\Delta t = 0.3\,[\mathrm{ms}]$)

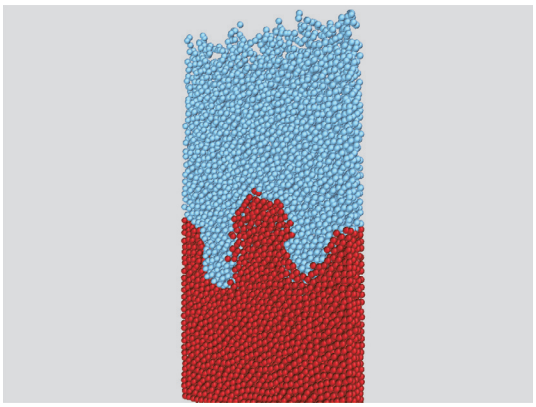


(a) $t = 0\,[\mathrm{s}]$

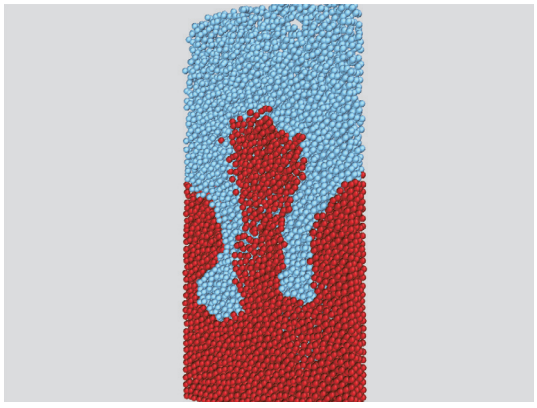
(b) $t = 0.6\,[\mathrm{s}]$



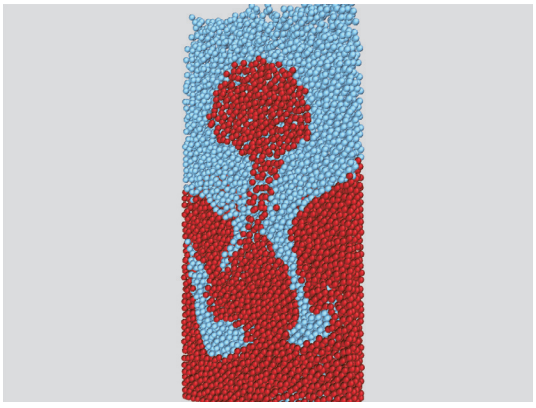
(c) $t = 1.2\,[\mathrm{s}]$



(d) $t = 1.8\,[\mathrm{s}]$

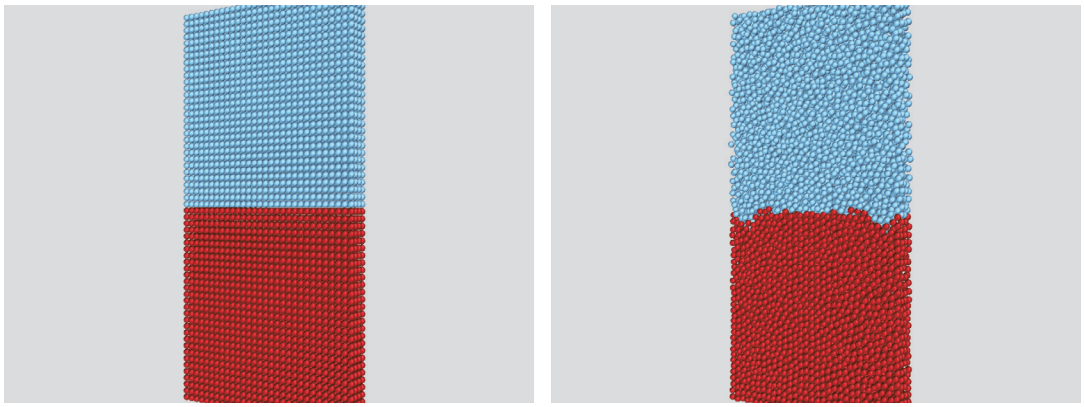


(e) $t = 2.4\,[\mathrm{s}]$

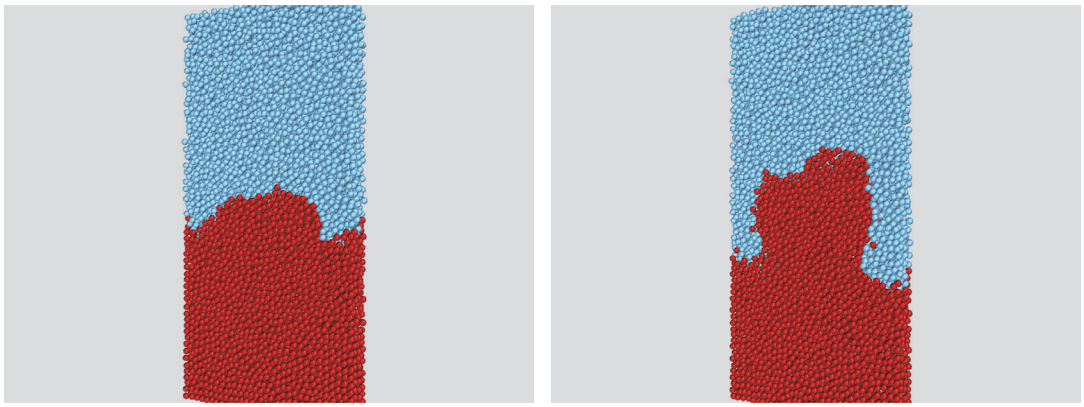


(f) $t = 3.0\,[\mathrm{s}]$

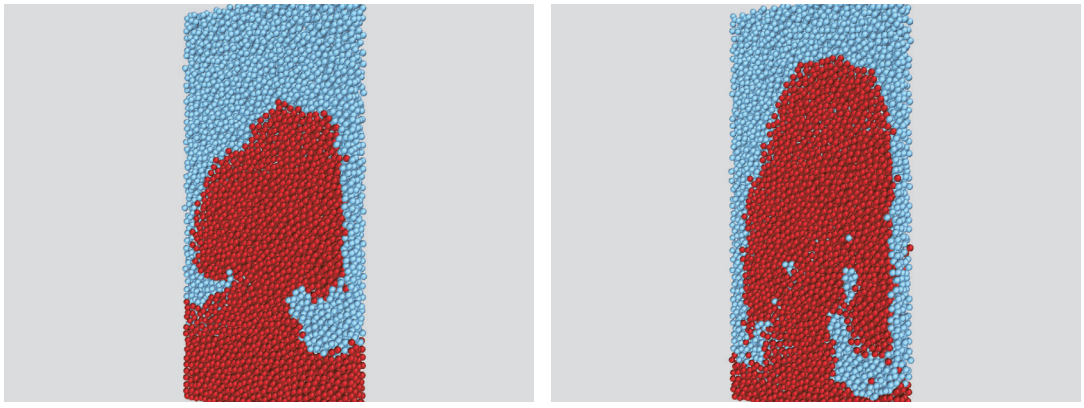
図 4.3: 従来法によるレイリー・テイラー不安定性の実験 ($\Delta t = 3.0\,[\mathrm{ms}]$)



(a) $t = 0\,[\mathrm{s}]$ (b) $t = 0.6\,[\mathrm{s}]$



(c) $t = 1.2\,[\mathrm{s}]$ (d) $t = 1.8\,[\mathrm{s}]$



(e) $t = 2.4\,[\mathrm{s}]$ (f) $t = 3.0\,[\mathrm{s}]$

図 4.4: 提案法によるレイリー・テイラー不安定性の実験 ($\Delta t = 3.0\,[\mathrm{ms}]$)

4.2　4 つの液体を用いた実験

実験内容

4.1 節では 2 つの流体におけるレイリー・テイラー不安定性の実験を行ったが，本節では密度の異なる 4 つの液体が存在する場合のシミュレーションを行う．シーンとしては，2 つの液体が既に存在している水槽の中に，もう 2 つの液体を注ぎ込み，最終的に密度の大きさによって液体が分離するような場面を想定している．これにより，流体の数に関わらず，提案法により安定した相互作用計算ができることを確認する．実験で用いたパラメータを表 4.4 に示す．

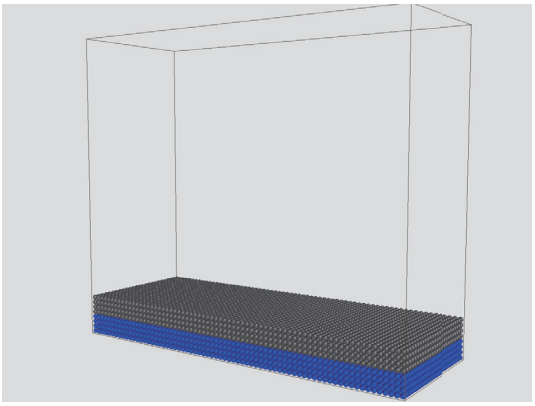
表 4.4: 4 つの液体を用いた実験における各パラメータ

流体の色	水色	赤	グレー	青
初期密度 $\rho_0\,[\mathrm{kg/m^3}]$	1000.0	500.0	100.0	10.0
表面張力係数 $\gamma\,[\mathrm{N/m}]$	1.0	1.0	1.0	1.0
粘性係数 $\mu\,[\mathrm{Pa\cdot s}]$	0.001	0.001	0.001	0.001

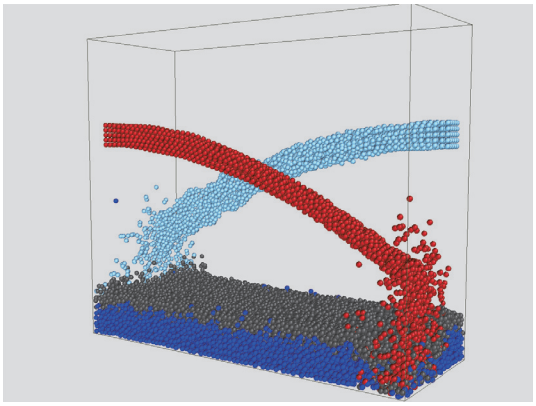
実験結果

結果の画像を図 4.5 に示す．シミュレーション内における最大粒子数は，4 つの液体を合計して約 24,200 個となった．4.1 節の実験と同様にシミュレーション空間の周囲は 15,792 個の固体境界粒子で囲まれており，結果の画像では全て非表示としている．タイムステップ幅は $\Delta t = 3.0\,[\mathrm{ms}]$ として $t = 9.0\,[\mathrm{s}]$ までシミュレーションを行い，1 ステップ当たりの平均計算時間は 0.93 [s/step] となった．

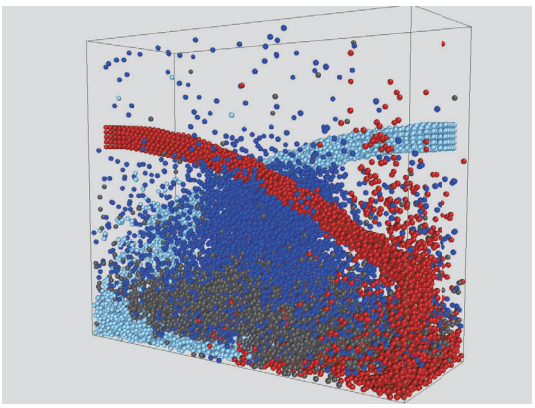
実験結果では，図 4.5(b) から (d) において，密度の最も小さい青色の粒子が，他の液体粒子からの作用により大きく移動していることがわかる．シミュレーションが進むにつれて粒子の動きが落ち着いていき，最終的には図 4.5(f) のように液体が分離した．同図からわかるように，液体が密度の小さい順に積み重なっている．これにより，流体の数が増えた場合でも，安定した相互作用計算が提案法によって実現できるといえる．また，表 4.4 で示したように相互作用する液体同士の密度比は最大で 100:1 となっている．従来法においても，実験で用いた最大の密度比は 100:1 なので，従来法と同程度の密度比によるシミュレーションが可能であることを確認した．しかし一部の粒子に関して，図 4.5(e) や図 4.5(f) のように，異なる液体の中で孤立しているものが確認できる．原因として，孤立粒子は自身と同じ性質の粒子が近傍に存在しないため，近傍の異なる液体粒子によりその動きが抑えられていることが考えられる．



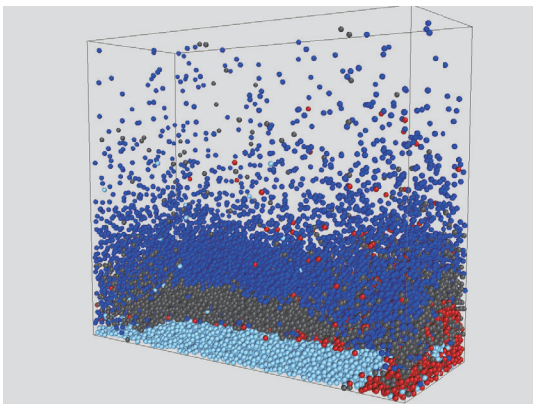
(a) $t = 0$ [s]



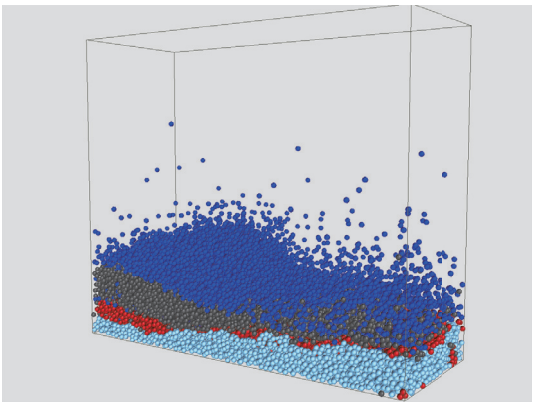
(b) $t = 0.9$ [s]



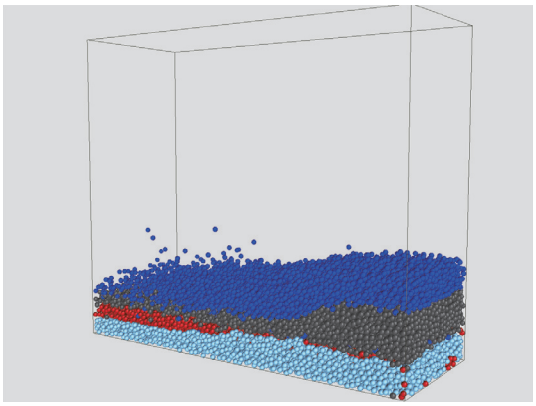
(c) $t = 1.8$ [s]



(d) $t = 3.6$ [s]



(e) $t = 5.4$ [s]



(f) $t = 9.0$ [s]

図 4.5: 4 つの液体を用いたシミュレーション

4.3 気泡シミュレーション

気泡の運動は，液体内に存在する気体との相互作用によって引き起こされ，現実世界ではそれによる多くの特徴的な挙動が確認される．よく見られる挙動として，液体内を気泡が上昇し，水面で薄い液膜を張った状態で漂う，または割れるといったものが挙げられる．本節では，これらの挙動を提案法より再現し，より幅広い現象の表現ができることを示す．実験で用いたパラメータは表 4.5 の通りである．また，シミュレーション空間は上部をオープンにする代わりに気体粒子の層を一定の高さで敷き詰め，それ以外は固体境界粒子 (結果の画像では非表示) で囲まれているとする．

表 4.5: 気泡シミュレーションにおける各パラメータ

粒子の種類	液体	気体
初期密度 $\rho_0\,[\mathrm{kg/m^3}]$	1000.0	50.0
表面張力係数 $\gamma\,[\mathrm{N/m}]$	1.0	0.0
粘性係数 $\mu\,[\mathrm{Pa\cdot s}]$	0.001	0.0
タイムステップ幅 $\Delta t\,[\mathrm{ms}]$	3.0	

4.3.1 液体内を上昇する気泡の変形および破裂

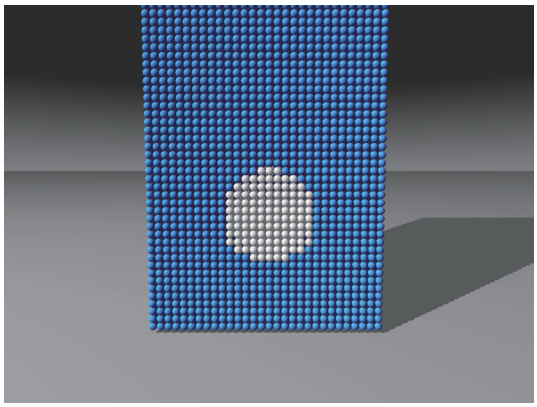
実験内容

現実世界では，気泡の径が大きいほどその形状も球形から変化する．一般的に，小さいものはほぼ完全な球形を保ち，径が大きくなるにつれて楕円形，キノコ傘状へと変化していくことが知られている．また，径が大きい気泡は，水面に到達するとすぐに割れてしまう．このような挙動を再現するために，まずは 1 つの気泡の水面までの上昇をシミュレートする．シミュレーションでは複数の液体粒子と気体粒子を用いて 1 つの大きな泡を構成し，提案法を用いて相互作用計算を行う．

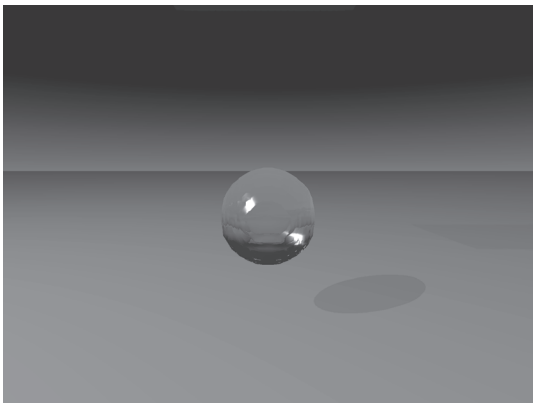
実験結果

結果の画像を図 4.6 および図 4.7 に示す．シミュレーションには液体粒子 39,600 個，気体粒子 5,400 個，固体境界粒子 7,224 個を使用し，$t = 3.0\,[\mathrm{s}]$ までシミュレーションを行った．1 ステップ当たりの計算時間は 2.1 [s/step] である．粒子による結果の画像 (図 4.6，図 4.7 の左列) は，奥行き $z = 0$ における断面を描画しており，青色の粒子を液体，白色の粒子を気体としている．また，それぞれの図の右列は 3.6 節の手法で液体表面を抽出してレンダリングした結果である．

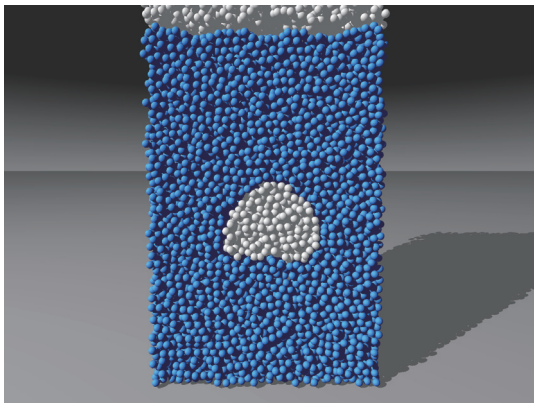
時系列に沿って見ていくと，図 4.6(a)，(b) から図 4.7(a)，(b) に至るまで，気泡の形状が球形からキノコ傘状へと変形している様子が再現できていることがわかる．その後，図 4.7(c) および (d) のように水面まで達した気泡は，図 4.7(e) および (f) のように液体粒子による液膜が破れ，崩壊していく様子を確認することができる．



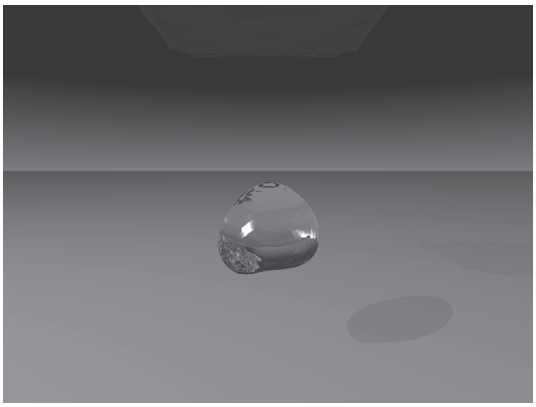
(a) $t = 0$ [s]



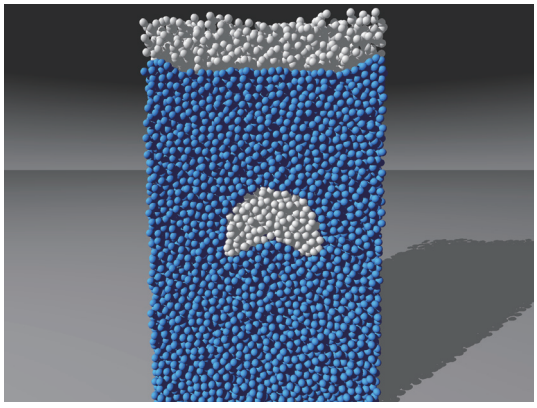
(b) $t = 0$ [s](レンダリング)



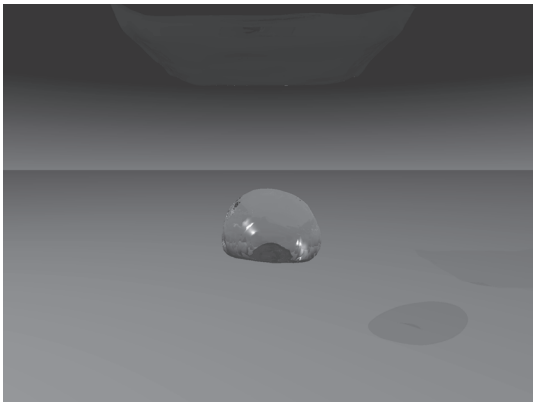
(c) $t = 0.4$ [s]



(d) $t = 0.4$ [s](レンダリング)

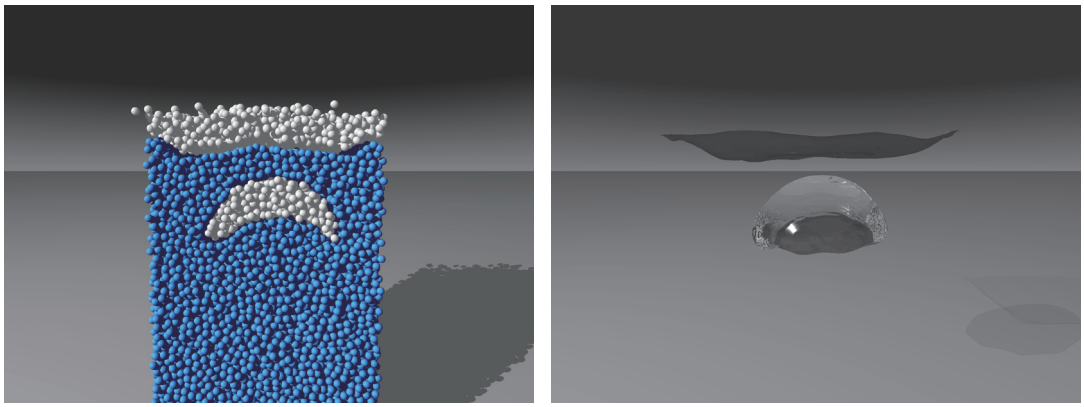


(e) $t = 0.6$ [s]

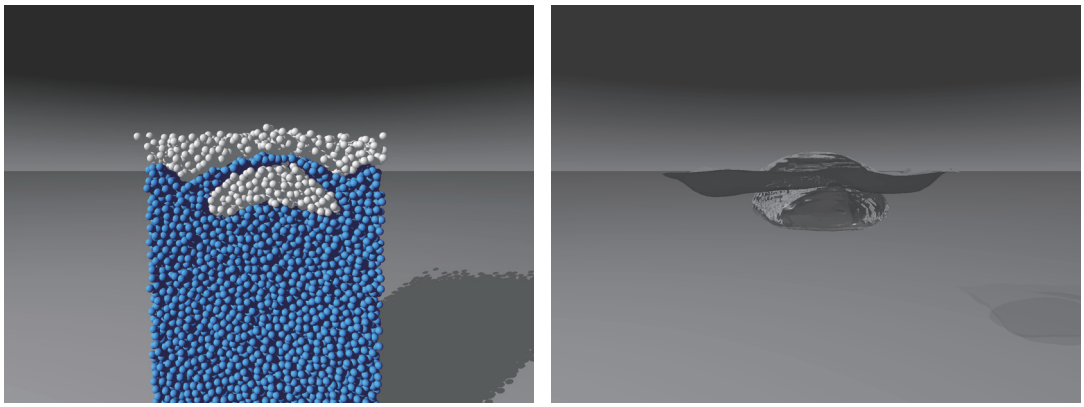


(f) $t = 0.6$ [s](レンダリング)

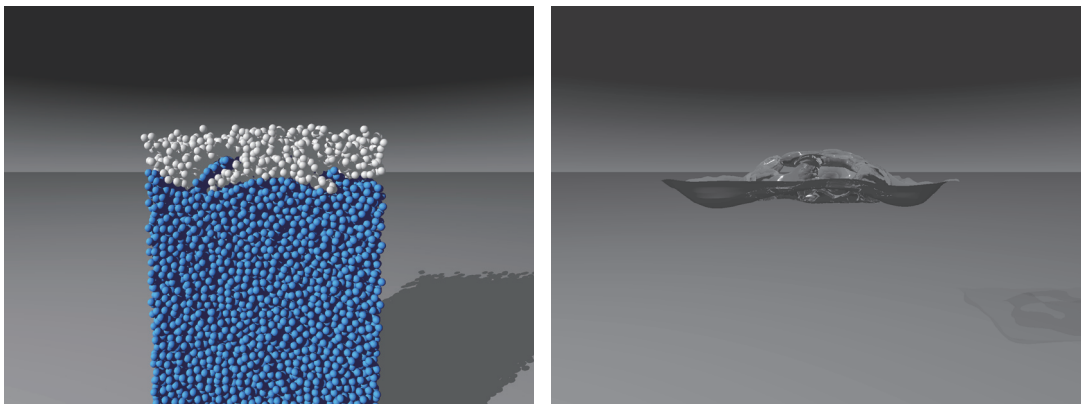
図 4.6: 気泡の変形から破裂までのシミュレーション結果 ($t = 0.6$ [s] まで)



(a) $t = 1.2$ [s]　(b) $t = 1.2$ [s](レンダリング)



(c) $t = 1.5$ [s]　(d) $t = 1.5$ [s](レンダリング)



(e) $t = 1.8$ [s]　(f) $t = 1.8$ [s](レンダリング)

図 4.7: 気泡の変形から破裂までのシミュレーション結果 ($t = 0.6$ [s] 以降)

4.3.2　水面で浮かぶ気泡

実験内容

液体内の気泡が水面まで上昇した際，破裂せずに薄い液膜を張った状態で浮かんでいる様子が確認されることがある (図 4.8)．このような水面で浮かぶ気泡は，表面張力や粘性，気泡内外の圧力差など，様々な要因によって破裂せずにその形状が維持される．粒子法による水面で浮かぶ気泡は文献 [12] や [7] でもシミュレーションされているが，これらは液体粒子による液膜を構成していない．本手法では，3.5 節で説明した表面張力計算を薄い液膜を保つ要因としている．これを用いて，液体粒子による液膜を構成し，水面に浮かぶ気泡をシミュレーションにより再現する．シーンとしては，液体内に 9 つの気泡を配置し，全ての気泡が水面で割れずに浮かんでいる場面を想定している．

実験結果

結果の画像を図 4.9 に示す．シミュレーションには液体粒子 76,823 個，気体粒子 19,977 個，固体境界粒子 11,116 個を使用し，$t = 3.0$ [s] までシミュレーションを行った．1 ステップ当たりの計算時間は 4.5 [s/step] である．

図 4.9(a) において，気泡は水面に近い場所に 4 つ，離れたところに 5 つ配置されている．図 4.9(c) におけるタイミングで水面に近い気泡が水面まで到達し，シミュレーションの終盤まで割れずに残っている様子が図 4.9(f) より確認できる．液膜は，本手法で用いている異方性フィルタによる表面張力の補正により，その状態を維持することができている．このような液膜を保った状態で水面に浮かぶ気泡を，粒子法により再現した研究はこれまでに存在しなかったため，本手法を用いることで既存手法よりも幅広い現象の表現が可能となる．

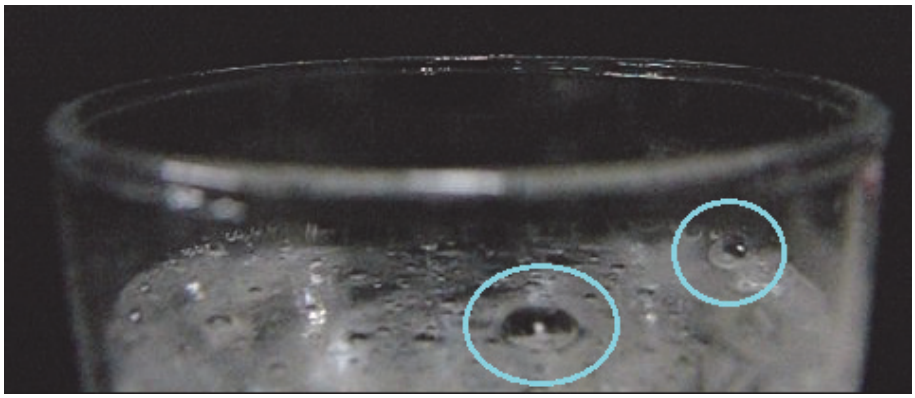
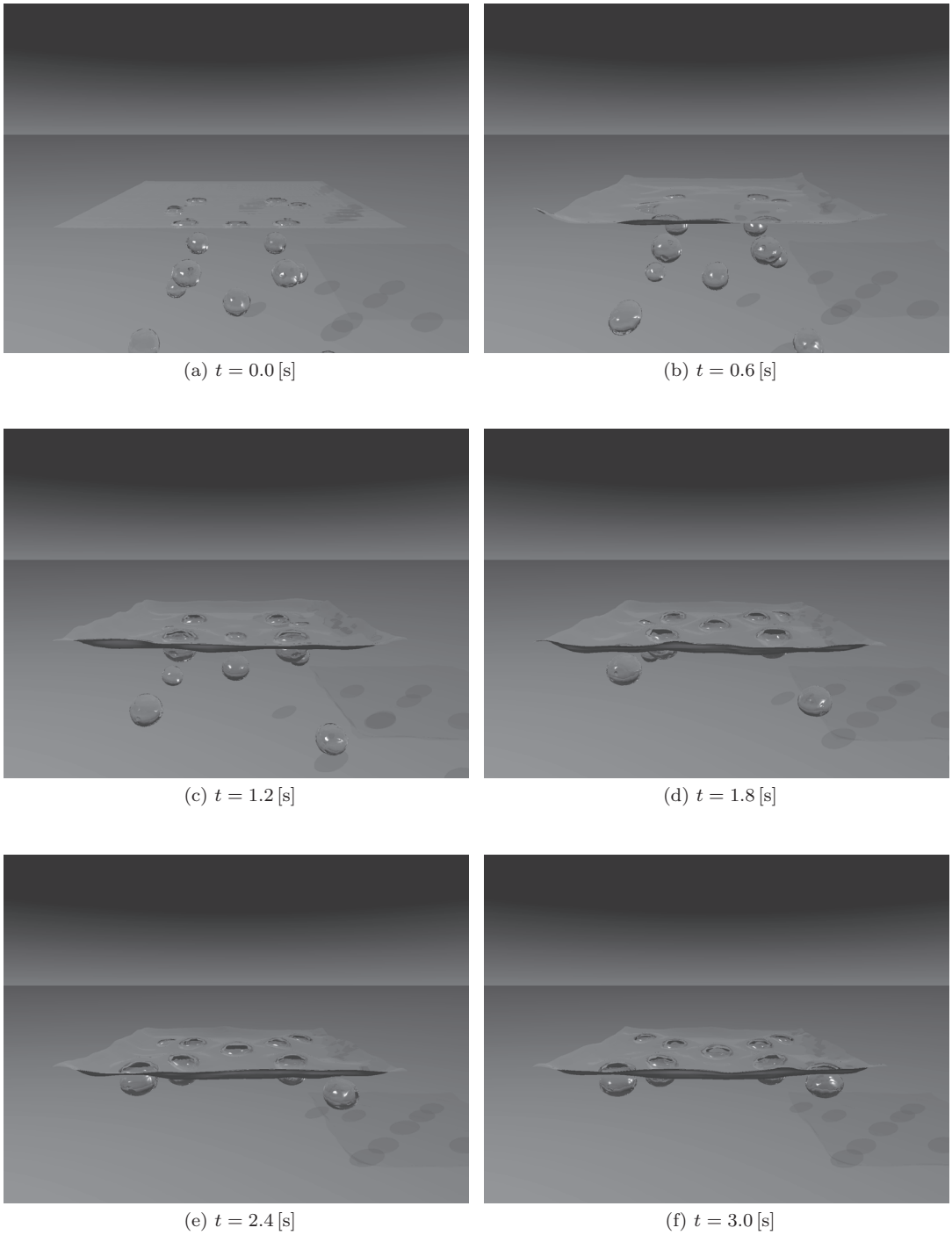


図 4.8: 現実で確認される水面で浮かぶ気泡 (著者撮影)



(a) $t = 0.0\,[\mathrm{s}]$ (b) $t = 0.6\,[\mathrm{s}]$

(c) $t = 1.2\,[\mathrm{s}]$ (d) $t = 1.8\,[\mathrm{s}]$

(e) $t = 2.4\,[\mathrm{s}]$ (f) $t = 3.0\,[\mathrm{s}]$

図 4.9: 水面で浮かぶ気泡のシミュレーション

4.3.3　様々な気泡の挙動

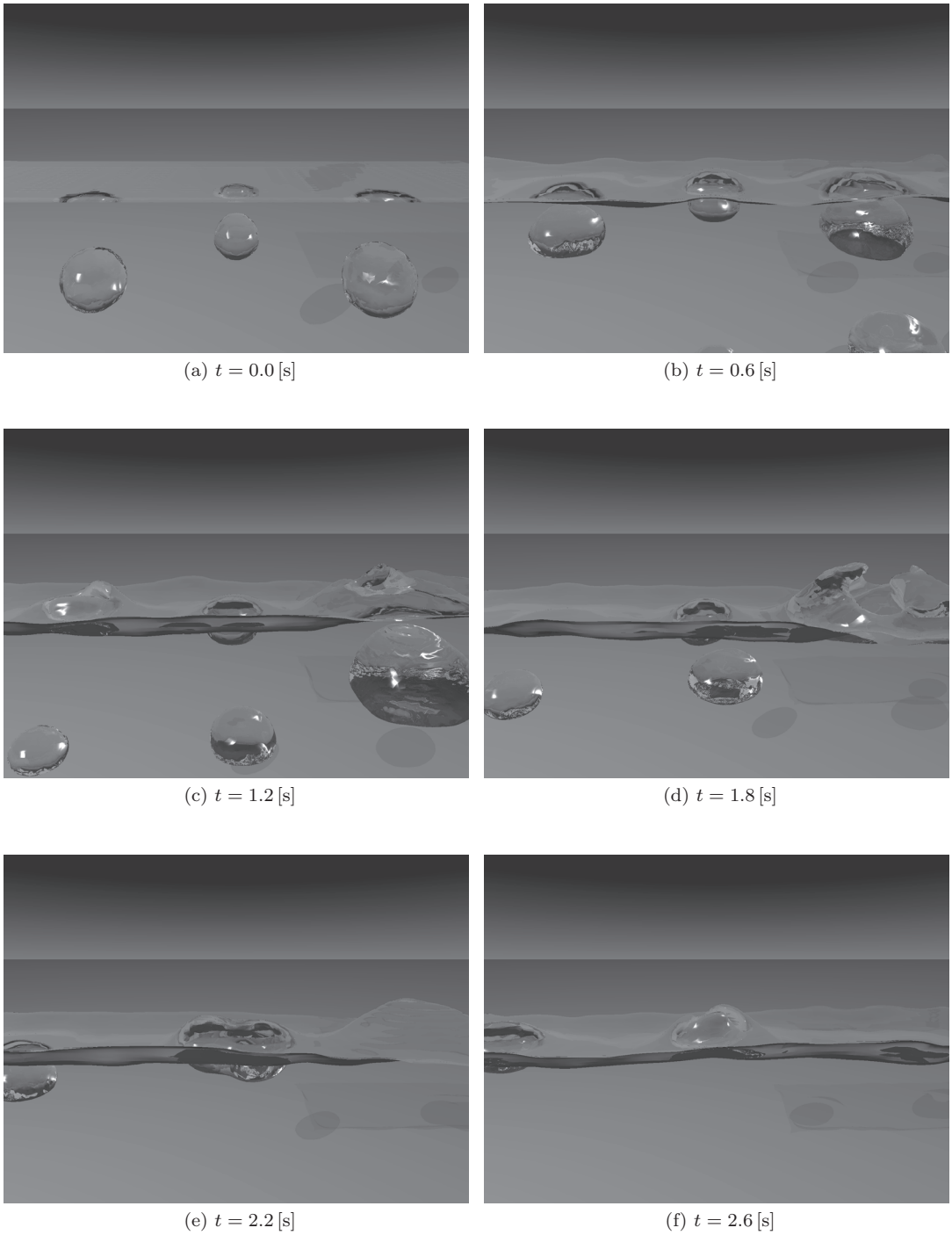
実験内容

これまでの実験では特定の挙動に焦点を当ててシミュレーションを行ってきたが，本実験ではそれらの挙動が同時に発生するようなシーンを再現する．シミュレーションでは大きさの異なる 6 つの気泡を配置し，大きさによって割れる気泡と割れない気泡を同時に再現する．また，割れずに水面で浮かぶ気泡と上昇する気泡との合流により，気泡同士が結合して割れる様子も再現する．

実験結果

結果の画像を図 4.10 に示す．シミュレーションには液体粒子 85,197 個，気体粒子 24,003 個，固体境界粒子 13,776 個を使用し，$t = 3.0$ [s] までシミュレーションを行った．1 ステップ当たりの計算時間は 5.0 [s/step] である．

図 4.10(a) において 3 つある気泡の下に，残りの 3 つの気泡が配置されている．図 4.10(b) から (c) にかけて，真ん中の小さな気泡は水面に到達しても割れず，左右の大きな気泡は水面で破裂している様子が確認できる．その後，図 4.10(d) から (e) にかけて，上昇してきた大きな気泡と水面で浮かんでいた小さな気泡が結合し，図 4.10(f) で破裂している様子が確認できる．ここで，本手法によって再現された水面で浮かぶ気泡は，図 4.9 のように一度水面に到達するとずっと割れずに残るため，シミュレーションの途中で破裂させるには外部から何かしらの作用を加える必要がある．実験では，図 4.10(d) から (f) に示すように，上昇してきた気泡の動きを外部からの作用として，水面で浮かぶ気泡に加えている．これにより，液膜が破れて気泡同士が結合，破裂する挙動の再現が可能となる．現実世界では外部からの作用が無くとも，時間経過による水面に浮かぶ気泡の破裂が確認できる．この要因として，重力によって液膜表面の液体が流れ落ちることで，液膜がさらに薄くなるという現象がある．しかし本手法において，液体粒子で構成された液膜は粒子 1 個分の幅が最小の厚さとなる．そのため，粒子 1 個分の幅でも割れない場合，本実験のように外部からの作用を加えない限り，シミュレーションの途中で破裂させることは難しいと考えている．



(a) $t = 0.0\,[\mathrm{s}]$ (b) $t = 0.6\,[\mathrm{s}]$

(c) $t = 1.2\,[\mathrm{s}]$ (d) $t = 1.8\,[\mathrm{s}]$

(e) $t = 2.2\,[\mathrm{s}]$ (f) $t = 2.6\,[\mathrm{s}]$

図 4.10: 様々な気泡の挙動

第5章

結論

本論文では，複数の流体を用いた多相流体シミュレーションを，非圧縮を満たしながら行うことができるSPH法を提案した．提案法では，流体境界付近の正確な密度計算を実現するために，近傍の粒子分布から計算される粒子密度を導入した．非圧縮性の確保については非圧縮性SPH法の一種であるIISPHを用いて，さらにIISPHで解く圧力のポアソン方程式を粒子密度ベースの式に修正することで，従来法よりも安定した多相流体シミュレーションを可能とした．提案法を用いた実験では，レイリー・テイラー不安定性の実験による従来法との比較や，4種類の異なる液体のシミュレーションを行い，より大きなタイムステップ幅でも安定した相互作用計算が行えることを示した．気泡シミュレーションの結果からは，提案法に異方性フィルタを用いた表面張力計算を導入することで，液体内を上昇する気泡の変形や，液膜を張った気泡が水面で浮かぶ様子，または破裂する様子などの再現が可能となることがわかった．これにより，既存の粒子による手法では不可能だったシーンも含め，より幅広い多相流体流れのシーンを安定した相互作用計算により再現できることを確認した．

本研究の今後の発展として，炭酸水などにおける気泡の発生が挙げられる．気泡の発生は，液体内に溶け込んでいる空気がグラスといった固体表面に存在する細かな傷やちりに集まる現象が要因となっている．これを粒子法により再現する場合，液体内に溶け込んでいる空気をどのように定義するかが問題となる．ただ，気泡の発生は液体内に突然気泡が現れるような現象であると考えると，溶け込んでいる空気を考慮せずとも，液体粒子の中に気体粒子を発生させたり，液体粒子を気体粒子に置き換えたりすることで実現はできる．しかし，これらは非物理的な操作であるため，質量保存や安定した相互作用計算といった性質が失われる可能性がある．現段階で最も有用だと考えられる手法として，局所的に粒子サイズを変化させる方法がある [21, 14, 26]．これらでは粒子ごとに流体表面からの距離を計算し，距離が遠い位置にある粒子を大きく，表面付近の粒子を小さくすることで，粒子数を削減した状態できれいな表面形状を得ることができる．サイズを変化させる過程で粒子同士の結合や，複数の粒子に分割するといった操作を行っているが，これらは質量保存や運動量保存を考慮した上で行っている．そのため，液体粒子を分割する際に気体粒子を発生させることで，物理的な制約を満たしながら気泡の発生が再現できるのではないかと考えている．

その他，GPU実装による計算の高速化が挙げられる．本研究では，OpenMPを用いてマルチスレッドCPUで並列計算を実行しているが，それでも流体粒子100,000個程度で1ステップ当たり5秒近く計算時間がかかっている．本研究で使用しているほとんどのアルゴリズムは並列処理が可能なので，GPUによる並列計算を行うことで，大幅な計算時間の短縮が期待できる．考慮しなければならない要素としては，提案法では反復計算を終了させる条件として全粒子の平均密度誤差を求めているため，あるスレッドが計算しているときに他のスレッドからのアクセスを制限するよう工夫する必要がある．

謝辞

本論文を締めくくるにあたり，研究活動において多岐に渡りご指導ご協力いただきました藤澤誠助教に深く感謝致します．また，研究内容に関して様々な助言をいただきました三河正彦准教授に感謝致します．そして，研究活動だけでなく，日常生活においても多くのご協力をいただきました物理ベースコンピュータグラフィックス研究室の皆様，ソーシャルロボット研究室の皆様に御礼申し上げます．

参考文献

[1] POV-Ray - The Persistence of Vision Ray. www.povray.org/.

[2] Nadir Akinci, Gizem Akinci, and Matthias Teschner. Versatile surface tension and adhesion for sph fluids. *ACM Trans. Graph.*, Vol. 32, No. 6, pp. 182:1–182:8, 2013.

[3] Nadir Akinci, Markus Ihmsen, Gizem Akinci, Barbara Solenthaler, and Matthias Teschner. Versatile rigid-fluid coupling for incompressible sph. *ACM Trans. Graph.*, Vol. 31, No. 4, pp. 62:1–62:8, 2012.

[4] Markus Becker and Matthias Teschner. Weakly compressible sph for free surface flows. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '07, pp. 209–217, 2007.

[5] Jan Bender and Dan Koschier. Divergence-free smoothed particle hydrodynamics. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA '15, pp. 147–155, 2015.

[6] Landon Boyd and Robert Bridson. Multiflip for energetic two-phase fluid simulation. *ACM Trans. Graph.*, Vol. 31, No. 2, pp. 16:1–16:12, 2012.

[7] Oleksiy Busaryev, Tamal K. Dey, Huamin Wang, and Zhong Ren. Animating bubble interactions in a liquid foam. *ACM Trans. Graph.*, Vol. 31, No. 4, pp. 63:1–63:8, 2012.

[8] R. A. Gingold and J. J. Monaghan. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society*, Vol. 181, No. 3, pp. 375–389, 1977.

[9] Jeong-Mo Hong and Chang-Hun Kim. Discontinuous fluids. *ACM Trans. Graph.*, Vol. 24, No. 3, pp. 915–920, 2005.

[10] Jeong-Mo Hong, Ho-Young Lee, Jong-Chul Yoon, and Chang-Hun Kim. Bubbles alive. *ACM Trans. Graph.*, Vol. 27, No. 3, pp. 48:1–48:4, 2008.

[11] M. Ihmsen, J. Cornelis, B. Solenthaler, C. Horvath, and M. Teschner. Implicit incompressible sph. *IEEE Transactions on Visualization & Computer Graphics*, Vol. 20, No. 3, pp. 426–435, 2014.

[12] Markus Ihmsen, Julian Bader, Gizem Akinci, and Matthias Teschner. Animation of air bubbles with sph. In *GRAPP 2011 - Proceedings of the International Conference on Computer Graphics Theory and Applications*, pp. 225–234, 2011.

[13] Markus Ihmsen, Jens Orthmann, Barbara Solenthaler, Andreas Kolb, and Matthias Teschner. Sph fluids in computer graphics. In *Proceedings of Eurographics 2014 - State of the Art Reports*, 2014.

[14] Christopher Jon Horvath and Barbara Solenthaler. Mass preserving multi-scale sph pixar technical memo #13-04. Technical report, 2013.

[15] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, pp. 163–169, 1987.

[16] L. B. Lucy. A numerical approach to the testing of the fission hypothesis. *Astron. J.*, Vol. 82, pp. 1013–1024, 1977.

[17] J. J. Monaghan. Smoothed particle hydrodynamics. *Annual Review of Astronomy and Astrophysics*, Vol. 30, No. 1, pp. 543–574, 1992.

[18] J. J. Monaghan. Smoothed particle hydrodynamics. *Reports on Progress in Physics*, Vol. 68, No. 8, p. 1703, 2005.

[19] 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.

[20] Matthias Müller, Barbara Solenthaler, Richard Keiser, and Markus Gross. Particle-based fluid-fluid interaction. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '05, pp. 237–244, 2005.

[21] Jens Orthmann and Andreas Kolb. Temporal blending for adaptive sph. *Computer Graphics Forum*, Vol. 31, pp. 2436–2449, 2012.

[22] B. Solenthaler and R. Pajarola. Predictive-corrective incompressible sph. *ACM Trans. Graph.*, Vol. 28, No. 3, pp. 40:1–40:6, 2009.

[23] Barbara Solenthaler and Renato Pajarola. Density contrast sph interfaces. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '08, pp. 211–218, 2008.

[24] K. Szewc, J. Pozorski, and J.-P. Minier. Simulations of single bubbles rising through viscous liquids using smoothed particle hydrodynamics. *International Journal of Multiphase Flow*, Vol. 50, pp. 98–105, 2013.

[25] Alexandre Tartakovsky and Paul Meakin. Modeling of surface tension and contact angles with smoothed particle hydrodynamics. *Phys. Rev. E*, Vol. 72, p. 026301, 2005.

[26] Rene Winchenbach, Hendrik Hochstetter, and Andreas Kolb. Infinite continuous adaptivity for incompressible sph. *ACM Trans. Graph.*, Vol. 36, No. 4, pp. 102:1–102:10, 2017.

[27] T. Yang, R. R. Martin, M. C. Lin, J. Chang, and S. Hu. Pairwise force sph model for real-time multi-interaction applications. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 23, No. 10, pp. 2235–2247, 2017.

[28] Jihun Yu and Greg Turk. Reconstructing surfaces of particle-based fluids using anisotropic kernels. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, pp. 217–225, 2010.

[29] Wen Zheng, Jun-Hai Yong, and Jean-Claude Paul. Simulation of bubbles. *Graphical Models*, Vol. 71, No. 6, pp. 229–239, 2009.

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