Vol.2011-CG-145 No.24
Vol.2011-CVIM-179 No.24
2011/11/18

基本速度場の合成による炎のアニメーション

佐藤 周平[†1] 土橋 宜典[†1] 山本 強[†1]

近年，流体解析を用いて写実的な水や炎などを表現する研究が数多く行われている．しかし，流体シミュレーションは計算コストが高く，所望の映像を得るまでに多大な時間がかかってしまう．そこで本研究では，データベースを利用して任意の炎のアニメーションを生成する手法を提案する．本稿では，流体解析により得られた速度場をデータベースとして保存しておき，所望の流れの場をデータベース内の速度場の線形和として表現することで，ユーザ任意のアニメーションを生成する．

Synthesizing Animation of Fire by Linear Combination of Basic Velocity Fields

Syuhei Sato,[†1] Yoshinori Dobashi[†1]
and Tsuyoshi Yamamoto[†1]

In recent years, many methods have been proposed for simulating realistic motions of fluids such as water and fire. However, the computational cost for simulating fluid motion is very expensive. We propose a method using a precomputed database to solve this problem. Our method efficiently generates the desired animation of fire by linearly combining precomputed basic velocity fields stored in a database.

†1 北海道大学
Hokkaido University

1. はじめに

近年，コンピュータグラフィックス (CG) において煙や水などの流体運動を流体解析を用いてシミュレーションする研究が盛んに行われている．流体現象は，映画やゲームなどの映像において屋内外のシーンを問わず重要な要素の 1 つであり，様々な場面で利用されている．このような流体現象は，流体力学の物理シミュレーションに基づいた方法により写実的な表現が可能である[5]．そのため，流体シミュレーションを用いて効率良く，効果的に流体現象を表現する研究がこれまで数多く行われている．本研究では，これら流体現象の中でも炎のシミュレーションに注目する．

炎を流体シミュレーションを用いて表現する研究がいくつか提案されている[6][7]．これらの手法により，映画やゲームにおいて炎の映像を作成する際，写実的なアニメーションを生成することができる．しかし，流体解析を利用したシミュレーションはリアルな映像が得られる半面，計算コストが高い点が問題となる．また，所望の映像を作成するためには，一般に，数多くのパラメータを試行錯誤的に決定しなくてはならず，煩雑な作業を必要とする．そのため，映像制作において所望の映像を得るまでに膨大な時間がかかってしまう．

上記の問題を解決するために，流体シミュレーションを制御することでユーザの所望するアニメーションを生成するための研究が行われている[8][9][10]．これらの手法には，外力などを用いてシミュレーションを直接制御することで所望の形状の流体を得るものやシミュレーションパラメータを自動で調整して所望の流体アニメーションを生成するものなどが存在する．しかし，これらの手法を用いて得られる制御結果は必ずしもユーザの要求と一致するとは限らず，制御パラメータや目標とする形状などを繰り返し調整しなければならない．また，流体シミュレーションを用いているため計算コストが高く，所望の映像を得るまでには時間がかかってしまう．また，炎を対象としたシミュレーションの制御手法は存在しない．

本研究では，これらの問題を解決するため，データベースを利用した手法を提案する．すなわち，あらかじめシミュレーションにより作成した複数の炎のデータベース（基本速度場）を組み合わせることで，効率的に所望の映像を生成する．本手法では，低解像度の炎のシミュレーションにより所望の映像を作成し，高解像度の基本速度場の線形和により低解像度の速度場を近似することで，効率的に高解像度の所望の映像を生成することを目的としている．本稿では，炎の複雑さの指定と，高解像度化について本手法の有効性を確認する．提案手法により，所望の複雑さを持つ炎のアニメーションが生成できる．

Vol.2011-CG-145 No.24
Vol.2011-CVIM-179 No.24
2011/11/18

2. 関 連 研 究

炎のシミュレーションに関する研究として Nguyen らによる手法[6] が存在する．これは，燃料と燃焼物とを個別に流体解析し，レベルセット法を用いてその相互作用を計算することで実際の物理現象に近い炎のシミュレーションを行う手法である．また，Kang らは格子法と粒子法によるシミュレーションを組み合わせて，炎や爆発などの化学反応を伴った流体現象を表現する手法を開発した[7]．これらの手法ではリアルな炎のシミュレーションが可能であるが，計算コストが非常に高い．また，目的の映像を作成するためには，非常に多くのパラメータを試行錯誤的に決定しなければならない．

流体シミュレーションを制御することで所望の映像を生成する研究もいくつか提案されている[8)9)10)]．これらの手法では，外力によりシミュレーションを制御[8)9)] もしくはシミュレーショパラメータを自動調整[10)] することで所望の流体アニメーションを得る手法であるが，炎を対象にした手法は存在しない．

炎のアニメーションをインタラクティブに編集する方法[1)] が提案されている．しかし，この手法では炎の動きの解析に流体シミュレーションを用いておらず，炎の動きは単調なものとなってしまっている．また，前処理により計算した流体のシミュレーションデータを用いて効率的に流体の動きを生成する研究がいくつか存在する[2)3)4)]．本研究では，これらの手法の考え方を応用して効率的に所望のアニメーション生成を実現する．

3. 提案手法の考え方

提案手法の考え方を図 1 に示す．まず，渦補正力の係数など，パラメータをさまざまに設定して流体シミュレーションを行うことで，炎のデータベースを作成する．ただし，統計的に定常的な動きをする炎のシミュレーションを行い，シミュレーションパラメータが時間的に変化するようなシミュレーションは行わない．そして，得られたシミュレーション結果のうち，炎の速度場をデータとして保存する．ただし，全ての速度場を用いた場合，アニメーション生成時の計算量の増加とデータ容量が膨大になるため，保存した全速度場データに対し主成分分析（Principal Component Analysis：PCA）を用いて基本速度場（固有ベクトル）を算出し，これをデータベースとして利用する．

次に，データベース作成時よりも低解像度のシミュレーションを編集することで所望の炎を作成し，それを基にデータベースを利用して所望の炎の映像を生成する．ユーザは所望の炎の大きさや複雑さなどの特徴量を低解像度シミュレーションに対して指定する．低解像度

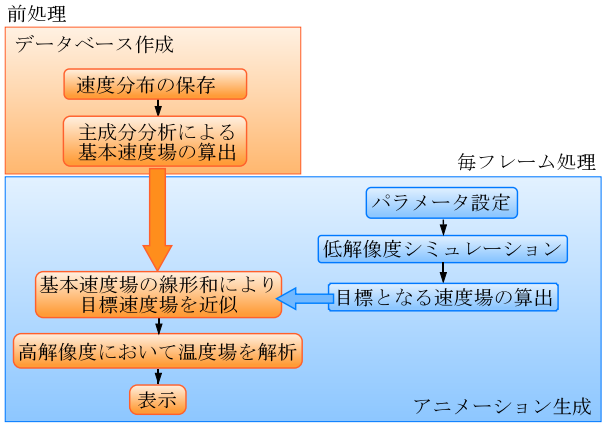


図 1 提案手法の考え方

なシミュレーションを編集することで流体解析における計算コストが少なくなり，所望の映像作成までの時間を削減することができる．そして，ユーザの意図が反映された低解像度シミュレーションの速度場をデータベース内の基本速度場の線形和として表現することで，所望の炎の速度場を合成する．この速度場に基づいて炎の温度場を移流させることで，所望の炎の映像が生成できる．以下で各処理について詳しく述べる．

4. 炎のシミュレーション方法

炎のシミュレーションデータの生成法について説明する．本稿では格子法により流体解析を行う．3 次元のシミュレーション空間を格子に分割し，シミュレーション空間の中央下端に炎を発生させる領域（ソース）を配置する．分割した各格子点に温度および速度を割りつける．シミュレーションにおける毎ステップの処理は以下のようになる．配置したソースに鉛直上向きの速度と温度を毎ステップ与えることで炎の発生を擬似的に表現する．その後，シミュレーション空間内の温度場および速度場を文献[6] の方法を応用して流体解析を行うことで炎をシミュレーションする．炎のシミュレーションにおける流体解析の支配方程式は

Vol.2011-CG-145 No.24
Vol.2011-CVIM-179 No.24
2011/11/18

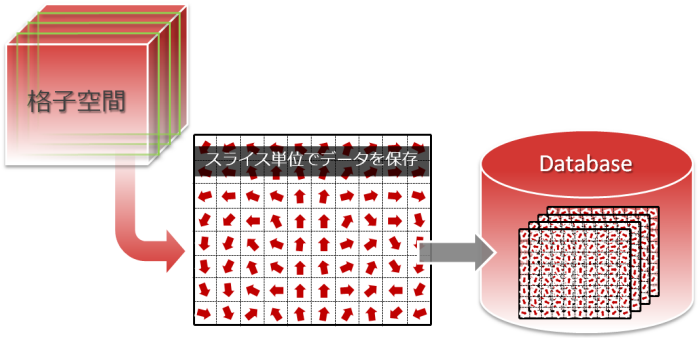


図 2 速度場のデータ

以下のようになる．

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

$$\nabla\cdot\mathbf{u} = 0 \tag{2}$$

p は圧力，ν は動粘性係数，$\mathbf{f}$ は外力である．式 (1) および (2) は，非圧縮性条件下における N-S 方程式であり，非圧縮性流体を扱う場合のみ式 (2) が成り立ち，この式は連続の式と呼ばれる．また，式 (1) は速度場の時間発展を表す方程式であり，右辺第 1 項から，移流項，圧力項，拡散項，外力項と呼ばれる．

温度場の時間発展は式 (1) により計算された速度場を用いて次式により表わされる．

$$\frac{\partial T}{\partial t} = -(\mathbf{u}\cdot\nabla)T - c_r\left(\frac{T - T_{amb}}{T_{max} - T_{amb}}\right)^4 + c_k\nabla^2 T \tag{3}$$

ここで c_r は冷却定数，T_{amb} は環境温度である．また，T_{max} はシミュレーション空間内での最大温度であり，シミュレーションにより得られた値を設定する．右辺第 2 項は流体温度の環境への放射損失を表している．また，c_k は熱伝導率を表しており，右辺第 3 項は温度の拡散項となる．

最後に浮力 $\mathbf{f}_{buo}$ と渦補正力 $\mathbf{f}_{conf}$ は，それぞれ以下の式で与えられる．

$$\mathbf{f}_{buo} = \kappa_b(T - T_{amb})\mathbf{y} \tag{4}$$

$$\mathbf{f}_{conf} = \varepsilon(\mathbf{N}\times\omega) \tag{5}$$

ここで，κ_b は浮力の係数，$\mathbf{y}$ は鉛直方向の単位ベクトルである．また，ε は係数，ω は $\omega = \nabla\times\mathbf{u}$ として表され，$\mathbf{N}$ は ω を用いて $\mathbf{N} = \nabla|\omega|/|\nabla|\omega||$ となる．この浮力と渦補正力は式 (1) の外力項に適用される．

以上の方法により算出された速度場を 1 ステップごとにシミュレーションデータとして保存する．ここでデータベースを作成する際のシミュレーションの格子数は水平方向 Nx，奥行き方向 Ny，鉛直方向 Nz とし，格子幅は dh とする．本稿では，奥行き方向に垂直な平面（スライス）ごとに速度場の合成を行うため，データの保存もスライス単位で行う（図 2 参照）．また，全データ数 N_{all} は，(データの種類）×（1 データごとのステップ数）となる．次節において，保存したデータに対し主成分分析を行うことで基本速度場を算出する．

5. 主成分分析（PCA）によるデータベースの圧縮

4 節で作成した炎の速度場データに対し，主成分分析を行いデータ数を圧縮することで，6 節における計算コストの削減とデータ容量の圧縮を行う．本稿では，PCA により算出された速度場をそのデータベースの基本速度場と呼ぶこととする．以下で詳細を説明する．

まず，スライスごとに速度場の圧縮を行うため，スライスの解像度の 3 倍 $(3\cdot Nx\cdot Nz)$ ×保存した全データ数 (N_{all}) の行列 X を以下のように作成する．

$$X = \begin{pmatrix} x_{0,0} & x_{1,0} & \cdots & x_{N_{all},0} \\ y_{0,0} & y_{1,0} & \cdots & y_{N_{all},0} \\ z_{0,0} & z_{1,0} & \cdots & z_{N_{all},0} \\ x_{0,1} & x_{1,1} & \cdots & x_{N_{all},1} \\ \vdots & \vdots & \ddots & \vdots \\ z_{0,NxNz} & z_{1,NxNz} & \cdots & z_{N_{all},NxNz} \end{pmatrix} \tag{6}$$

ここで，$x_{n,i}$，$y_{n,i}$，$z_{n,i}$ は 2 次元のスライスにおける各格子点の速度の x，y，z 成分を表している．また，i が 1 次元での格子点座標を示し，n は全データ N_{all} における n 番目のデータを表している．上記の行列に対し，主成分分析を行うことで，基本速度場となる行列 U が以下のように算出される．

$$U = \begin{pmatrix} x_{0,0} & x_{1,0} & \cdots & x_{N_{pca},0} \\ y_{0,0} & y_{1,0} & \cdots & y_{N_{pca},0} \\ z_{0,0} & z_{1,0} & \cdots & z_{N_{pca},0} \\ x_{0,1} & x_{1,1} & \cdots & x_{N_{pca},1} \\ \vdots & \vdots & \ddots & \vdots \\ z_{0,NxNz} & z_{1,NxNz} & \cdots & z_{N_{pca},NxNz} \end{pmatrix} \tag{7}$$

Vol.2011-CG-145 No.24
Vol.2011-CVIM-179 No.24
2011/11/18

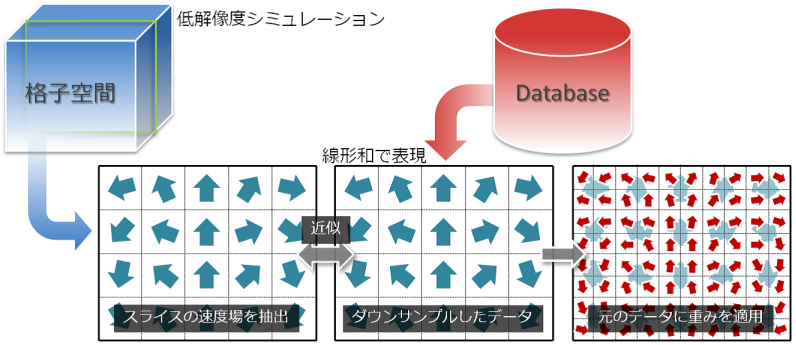


図 3 所望の速度場の生成

ここで，N_{pca} はユーザにより指定される基本速度場の数であり，主成分分析において算出する固有ベクトルの数である．また，式 (7) の各列が 1 つの基本速度場を表しており，左から第 1 主成分，第 2 主成分,...，第 N_{pca} 主成分となる．これを全スライスに対して行い，スライスごとに N_{pca} 個の基本速度場を算出しデータベースへ保存する．この基本速度場を組み合わせることで，各スライスにおける所望の速度場を生成する．

6. 基本速度場の線形和による所望の速度場の生成

データベース作成時よりも低解像度なシミュレーションにより所望の炎を作成し，前節で算出した基本速度場の線形和で表現することで高解像度な流体シミュレーションを行わずに所望の速度場を生成する．提案法における処理の流れを以下に示す．まず，低解像度シミュレーションにおいて渦補正力の係数 ε をユーザが自由に指定し所望の複雑さの炎を作成する．低解像度シミュレーションも 4 節の方法を用いて解析を行い，格子数は高解像度のシミュレーションに対して $1/M$ 倍，格子幅は M 倍となる．この炎の速度場を精度よく近似する高解像度な速度場を N_{pca} 個の基本速度場の線形和で生成する．生成した高解像度な速度場を用いて温度場を移流させることで所望の解像度における所望の炎を生成する．このとき，ソースへの温度の設定と式 (3) による温度の時間発展の計算は低解像度シミュレーションと同様に行う．以下では，低解像度なシミュレーションの速度場を基に，所望の高解像度な速度場を基本速度場の組み合わせで近似する方法について詳しく説明する．また，この処理の流れを図 3 に示す．

まず，低解像度における速度場を V_{L,y_l} とする．ここで，y_l はスライスの番号を表し，$y_l = 0, 1, 2, ..., Ny/M - 1$ である．また，データベース内の基本速度場を $V_{H,y_h,n}$ で表し，$y_h = 0, 1, 2, ..., Ny - 1$，$n = 0, 1, 2, ..., N_{pca}$ である．ただし，格子数の違いからスライスの数が異なるため，y_h/M において V_{L,y_l} が存在しない部分については前後のスライスから線形補間により生成する．次に，以下の条件を満たす重み $w_{y_h,n}$ を算出する．

$$|V_L - \sum_{n=0}^{N_{pca}} w_n V_{H,n}|^2 = 0 \tag{8}$$

しかし，基本速度場 $V_{H,n}$ と低解像度の速度場 V_L では格子数に違いがあるため，上記の計算ができない．そこで，基本速度場を V_L の格子数へダウンサンプリングし式 (8) を以下のようにする．

$$|V_L - \sum_{n=0}^{N_{pca}} w_n V_{HL,n}|^2 = 0 \tag{9}$$

ここで，$V_{HL,n}$ は基本速度場 V_H をダウンサンプリングしたものである．これにより式 (9) を満たす重み w_n が算出できる．そして，この重みを元の基本速度場 V_H に作用させることで V_L を近似した高解像度の速度場 V_{LH} が算出される．

$$V_{LH} = \sum_{n=0}^{N_{pca}} w_n V_{H,n} \tag{10}$$

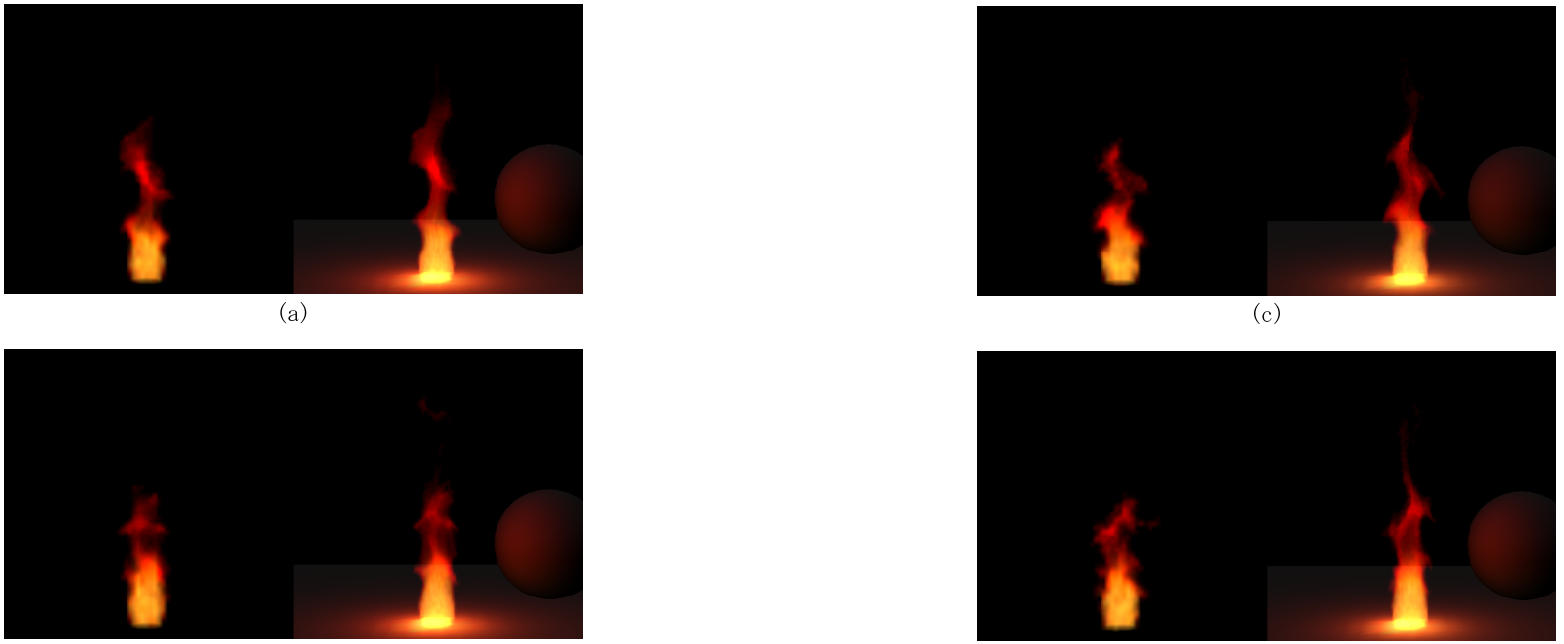
この V_{LH} を全スライスに対して算出し，所望の 3 次元の速度場を得る．

7. 実験結果

提案手法を適用して生成した炎のアニメーション結果を図 4 に示す．実験環境は，CPU が Intel Core i7 2600K（メモリ 4GB），GPU が NVIDIA GeForce GTX 580 となっている．データ作成時のシミュレーション空間は 64×64×128 の格子に分割し，各データのフレーム数は 150 である．また，シミュレーションパラメータのうち渦度係数 ε を変化させた 3 種類（$\varepsilon = 0.0, 15.0, 30.0$）のデータをデータベースとしてもっており，PCA により 128 個の基本速度場へ圧縮した．このときのデータ容量は約 0.81GB であり，基本速度場の算出までに要した時間は約 60 分である．さらに，編集に用いた低解像度シミュレーションでは空間を 32×32×64 の格子に分割した． 図 4(a),(b) はともに低解像度シミュレーションにおける渦度係数 ε を 10.0 に設定したときの結果であり，各図の左側が低解像度シミュレーションの結果を示し，右が基本速度場の合成により生成した結果である．どちらの結果についても低解像度の結果と同じ形の炎が得られているのが分かる．

図 4(c),(d) はともに低解像度シミュレーションにおける渦度係数 ε を 25.0 に設定したと

Vol.2011-CG-145 No.24
Vol.2011-CVIM-179 No.24
2011/11/18



(a) (c)

(b) (d)

図 4 適用例

きの結果である．こちらの結果についても図 4(a),(b) と同様に低解像度の結果と同じ形の炎が得られているのが分かる．

さらに，全ての結果から，データとして作成していない渦度係数を持った結果であっても合成が可能であることが確認できた．

8. まとめと今後の課題

本稿ではデータベースを用いることで，所望の炎の映像を効率よく生成する手法を提案した．いくつかの渦度をもった炎の速度場に対し PCA を用いて基本速度場を事前に算出，保存しておき，それらを組み合わせることでデータとして作成していない渦度をもった炎も生成できることを確認した．また，低い解像度のシミュレーションから高解像度の速度場を生成できることについても確認できた．

今後の課題としては，低解像度シミュレーションで指定できる項目の数を増やすことがあげられる．これにより，さらに多様な炎アニメーションの生成が期待できるが，新たに異なる条件でのシミュレーションデータを保持する必要があるためデータ容量が増大する可能性がある．また現状では，本手法とシミュレーションによる生成を比較して，計算コスト削減のメリットは微少であるため，計算時間を小さくすることも課題としてあげられる．これに関しては，各処理の GPU 化などにより高速化を図る必要がある．

謝辞 この研究は独立行政法人科学技術振興機構，CREST によりサポートされています．

Vol.2011-CG-145 No.24
Vol.2011-CVIM-179 No.24
2011/11/18

参 考 文 献

1) A. R. Fuller, H. Krishnan, K. Mahrous, B. Hamann, K. I. Joy, "Real-time Procedural Volumetric Fire," In Proceeding of the 2007 symposium on Interactive 3D graphics and games, pp.175-180, 2007
2) A. Treuille, A. Lewis, Z. Popovic, "Model Reduction for Real-time Fluids," ACM Transactions on Graphics, 25, 3, 2006
3) T. Kim, N. Thurey, D. James, M. Gross, "Wavelet Turbulence for Fluid Simulation," ACM Transactions on Graphics, 27, 3, 2008
4) M. Wicke, M. Stanton, A. Treuille, "Modular Bases for Fluid Dynamics," ACM Transactions on Graphics, 28, 3, 2009
5) J. Stam, "Stable Fluids," In Proceeding of ACM SIGGRAPH 1999, pp.121-128, 1999
6) D. Q. Nguyen, R. Fedkiw, H. W. Jensen, "Physically Based Modeling and Animation of Fire," In Proceeding of ACM SIGGRAPH 2002, pp.721-728, 2002
7) B. Kang, Y. Jang, I. Ihm, "Animation of Chemically Reactive Fluids Using a Hybrid Simulation Method," In Proceeding of ACM SIGGRAPH/Eurographics symposium 2007 on Computer animation, pp.199-208, 2007
8) R. Fattal, D. Lischinski, "Target-driven Smoke Animation," In Proceeding of ACM SIGGRAPH 2004, pp.439-446, 2004
9) L. Shi, Y. Yu, "Taming Liquids for Rapidly Changing Targets," In Proceeding of ACM SIGGRAPH/Eurographics symposium 2005 on Computer animation, pp.229-236, 2005
10) Y. Dobashi, K. Kusumoto, T. Nishita, T. Yamamoto, "Feedback control of cumuliform cloud formation based on computational fluid dynamics," ACM Transactions on Graphics, 27, 3, 2008