Vol.2019-CG-175 No.2
2019/9/28

空間および周波数領域での煙の流れの編集

佐藤 周平[1]　土橋 宜典[2]　Theodore Kim[3]

概要：物理ベースの流体シミュレーションはリアルな映像を作成できる反面，所望の動きを持ったクオリティの高い映像を得るためには未だに高いコストが必要となる．この問題を解決するために，我々は周波数と空間の両方の領域で同時に既存の速度場を編集できる方法を提案する．この編集は，ラプラシアンピラミッドを用いて複数の周波数バンドに速度場を分解することで達成される．そして，ユーザは各バンドの重みを調整することで周波数分布を，また，空間的な重みづけを通して空間的な分布を制御できる．提案手法は，元の速度場の代わりに流れ関数上でこれらの操作を実行することで，非圧縮性を保つ．本稿では，格子ベースの煙シミュレーションを対象とする．いくつかの格子ベースの例により提案システムの能力を示す．

キーワード：流体シミュレーション，非圧縮性，流れ関数，ラプラシアンピラミッド

1. はじめに

流体の物理ベースシミュレーションによる映像制作は，リアルな映像が作成でき，現在では映画やゲームなどのエンタテインメント分野で当たり前のように使用されている．しかし，その計算コストはいまだに高く，パラメータの試行錯誤的な調整が求められる映像制作では，所望の映像の完成までに要する時間が大きな問題となっている．

この問題に対し，様々なアプローチにより映像制作にかかるコストを削減するための研究が行われている．それらの中でも，近年既存の流体のシミュレーションデータを再利用し，再度シミュレーションを実行することなく，データとは異なる流れを作成するための方法がいくつか提案されている．Raveendran ら[1]や Thuerey[2]の方法では，複数のシミュレーションデータを入力し，その間の状態の流れを作成できる．Sato らの方法[3]では非圧縮性を保ったまま，流れを後処理的に変形することができる．また Sato らは，複数のデータをカット&ペーストで合成する方法を提案している[4]．これらの方法はいずれも主に大域的な流れが異なる結果を作成するものであり，詳細な動きの編集については扱っていない．

シミュレーションデータを再利用し，詳細な動きを後処理的に変更する研究がいくつかなされている．Kim らの手法[5]では，入力のデータを基底空間に投影し，乱流のみを調整することができる．Sato らの方法[6]では，2 つの流れのデータを用意することで，一方の詳細な動きをもう一方のデータに転写することができる．しかし，これらの研究は詳細な動きを全体的に変更できるが，局所的に調整することはできない．

そこで本研究では，流体の流れを空間と周波数の両方の領域において局所的に編集するための方法を提案する．提案手法は動画処理の分野で研究されている動画の誇張手法[7]の考え方をベースとしている．まず，既存の速度場のデータについてラプラシアンピラミッドを構築し，周波数成分を表すバンドに速度場を分解する．そして，その各バンドの分布についてユーザ指定の重み係数によりその値を変化させる．この時，その重み係数を空間において局所的に指定することで，空間および周波数領域において局所的な編集が可能となる．ただし，このままでは編集後の流れが流体の非圧縮性を満たさない．そこで事前に速度場を流れ関数[3, 8]に変換し，流れ関数に対して上記の処理を適用することで，常に非圧縮性を満たした状態で編集が可能である．

本稿では，格子上に定義される煙のシミュレーションを対象とする．いくつかの実験例により提案手法の有効性を示す．

2. 関連研究

既存の流体のデータを再利用して新しい流れを作成するための方法がいくつか提案されている．Raveendran らは複数の液体の表面データを滑らかにブレンドする方法を開発した[1]．この方法は半自動的に複数の入力表面のマッチングをとり，入力同士の中間の状態を表現する新しい液体表面を合成する．Thuerey も液体と煙のための補間方法を提案した[2]．この研究では格子上に定義された符号付距離関数で表現される煙と液体のデータを対象としている．複数のデータが入力されると，オプティカルフローに基づいた方法によりこれらの入力がマッチするような変形が計算される．ベクトルポテンシャルを用いて，Sato らは流体の非圧縮性を保証しながら流れを変形する方法を提案した[3]．また，Sato らは流体の流れを組み合わせるための方法を提案した[4]．この方法は複数の入力の流れの境界周辺において流れを補間することで，流体のカット&ペースト編

1 富山大学／プロメテック CG リサーチ
2 北海道大学／プロメテック CG リサーチ
3 Yale University

ⓒ2019 Information Processing Society of Japan

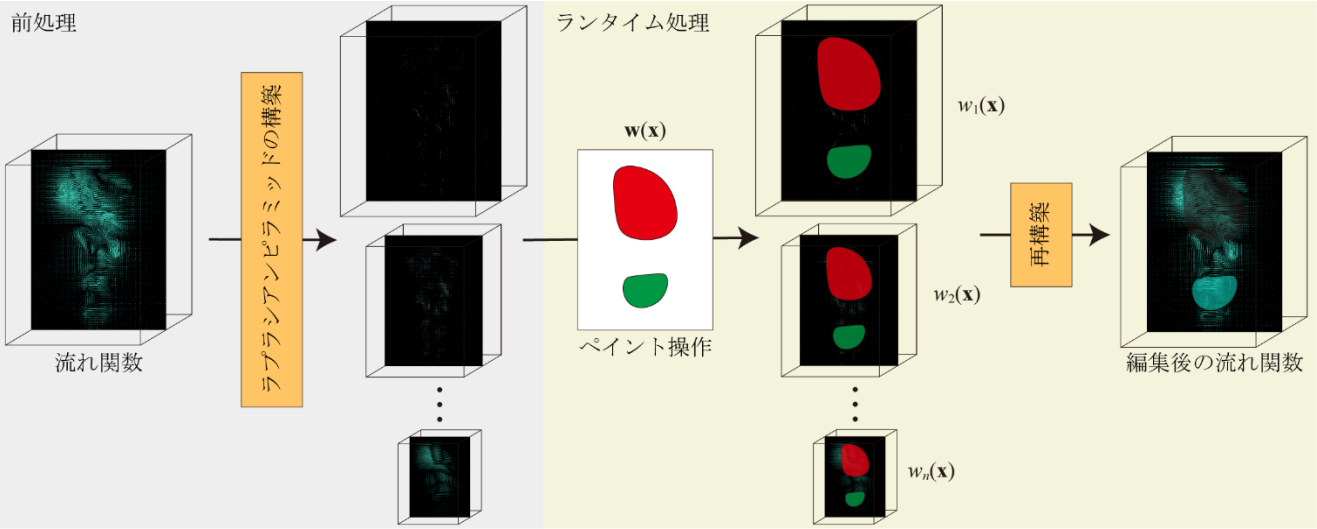

図 1：提案手法の概要．明快さのため，すべての 3 次元ベクトル場について，中心の平面のみを表示している．

集を可能とした．また，流体の物理法則を保持するように定式化がなされている．これらの方法を用いて，既存の流れから入力とは異なる新しい流れを合成することができる．しかし，これらの方法は大域的な流れが異なる結果を作成できるが，詳細な動きを編集することはできない．

既存のシミュレーションデータを用いて，詳細な動きを後処理的に変更するための方法がいくつか提案されている．Kim らは後処理的に入力の流れの乱流成分を変更するための方法を提案した[5]．この方法では，入力のデータから主成分分析（PCA）により基底関数を生成し，その基底空間において乱流に関するパラメータの異なる結果を再シミュレーションできる．ただし，PCA によるメモリの使用量が大きく，また，基底空間から速度を再構築する際にも高い計算コストを要する．Chu らは，多数のシミュレーションデータを CNN により学習し，入力の流れに詳細な動きを付加する方法を提案した[9]．Sato らは複数の流れのデータを入力し，一方のデータの詳細な動きをもう一方のデータに転写する方法を提案した[6]．この方法ではテクスチャ合成の技術をベースにして定式化がなされている．これらの方法により後処理的に詳細な動きを変更できるが，1 節でも述べたように，それを局所的に変更することはできない．

動画処理の分野において，入力の動画を誇張するための方法が提案されている．Liu らは動画中の物体の微小な動きを誇張する方法を提案した[10]．この方法では動画中の物体を正確にトラッキングし，その抽出された動きを強調する．Wu らはオイラーベースで動画中の変化を誇張する方法を提案した[7]．これによりピクセルの微小な色の変化を強調できる．その後 Wadhwa らは Wu らの方法[7]を complex steerable pyramid を用いて拡張した[11]．本研究では，Wu らの方法[7]の考え方の一部を応用し，流体の流れにおいて空間と周波数両方の領域での編集を実現する．

3. 提案手法

3.1 概要

図 1 に提案手法の概要を示す．提案手法のアルゴリズムは大きく分けて，前処理とランタイム処理の 2 つから構成される．まず前処理として，入力となる非圧縮な速度場のデータセットを用意し，この速度場を流れ関数へ変換する（3.2 節）．その後，ラプラシアンピラミッドを用いて，その流れ関数を複数の周波数バンドに分解する（3.3 節）．

次にランタイム処理において，編集を行う（3.4 節）．前処理において用意した流れ関数の各周波数バンドに対して，その重みを調整することで各バンドの分布を制御する．また，この重みは空間的に変化させることが可能であるため，空間領域でも各バンドの分布を制御できる．そして，編集後の各バンドから編集後の流れ関数を再構築する（3.4 節）．最後に流れ関数に対してカールオペレータを作用させて速度場に変換し，煙の密度を移流させ，結果を可視化する．

以下で各処理の詳細について述べる．

3.2 速度場から流れ関数への変換

入力の非圧縮な速度場を流れ関数へ変換する．この変換については数値流体解析の分野でこれまで扱われているものであり，本稿でも従来の方法を用いて変換する．Helmholtz-Hodge decomposition の理論に基づき，非圧縮性の速度場$\mathbf{u}$は流れ関数$\mathbf{\Psi}$を用いて以下のように表現される．

$$\mathbf{u} = \nabla \times \mathbf{\Psi} \tag{1}$$

ここで$\nabla\times$はカール演算子である．そして以下のポアソン方程式を解くことで$\mathbf{\Psi}$を求める[3, 8]．

$$\nabla \times \mathbf{u} = \nabla^2 \mathbf{\Psi} \tag{2}$$

ⓒ2019 Information Processing Society of Japan

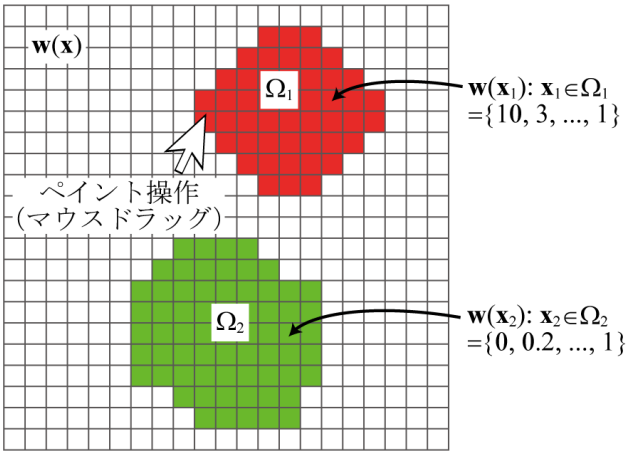

図 2：ペイント操作．Ω_1とΩ_2はそれぞれ赤と緑で示されるペイントされた領域を表す．

ここで∇^2はラプラス演算子である．この式はベクトル場の 3 つの成分ごとに独立に計算され，我々は conjugate gradient を用いてこれらの式を数値的に解く．

3.3 ラプラシアンピラミッドによる流れ関数の分解

前節で求めた流れ関数を複数の周波数バンドに分解する．本稿では，Wu らの方法[7]の考え方を導入し，ラプラシアンピラミッドを用いる．

まず，元の格子数$N \times N \times N$の流れ関数を$\mathbf{\Psi}_1$とし，ガウシアンフィルタを用いてそれを$N/2 \times N/2 \times N/2$にダウンサンプルした流れ関数$\mathbf{\Psi}_2$を得る．次に，ガウシアンフィルタを用いて$\mathbf{\Psi}_2$を$N \times N \times N$にアップサンプルした流れ関数$\mathbf{\Psi}_1'$を計算する．そして，$\mathbf{\Psi}_1 - \mathbf{\Psi}_1'$をレベル1の周波数バンドの分布$\hat{\mathbf{\Psi}}_1$とする．これを再帰的に繰り返し，レベル$n$までの分布$\{\hat{\mathbf{\Psi}}_1, \hat{\mathbf{\Psi}}_2, \ldots, \hat{\mathbf{\Psi}}_{n-1}, \mathbf{\Psi}_n\}$が得られる．ここで，$n$はユーザにより指定されたレベルの数である．レベル1から$n-1$は各周波数バンドの成分を表しており，レベルnはそれ以下の成分をすべて含んでいる．

上記の方法で分解された流れ関数を再構築する場合は，$\mathbf{\Psi}_n$を各方向 2 倍にアップサンプルし，$\hat{\mathbf{\Psi}}_{n-1}$に加算する．この処理を再帰的に繰り返すことで流れ関数が再構築される．これを数式で表すと以下のようになる．

$$\mathbf{\Psi}(\mathbf{x}) = \sum_{k=1}^{n-1} \mathbf{X}_k \hat{\mathbf{\Psi}}_k(\mathbf{x}) + \mathbf{X}_n \mathbf{\Psi}_n(\mathbf{x}) \qquad (3)$$

ここで$\mathbf{X}$は各成分を$N \times N \times N$の解像度までアップサンプルするための行列を示す．また$\mathbf{x}$は空間的な位置を表す．

3.4 空間および周波数領域での編集

前節の方法で分解した各周波数バンドの分布に重み係数を作用させて，その分布を制御する．本稿ではラプラシアンピラミッドにより各周波数バンドに分解したため，それぞれのバンドの分布は元の流れ関数と同様の空間領域に定義されている．そのため，重み係数も同様に空間上に$\mathbf{w}(\mathbf{x}) = \{w_1(\mathbf{x}), w_2(\mathbf{x}), \ldots, w_n(\mathbf{x})\}$のように定義される．そして，編集後の流れ関数$\mathbf{\Psi}_e$はこの重みを作用させて以下の式により計算される．

$$\mathbf{\Psi}_e(\mathbf{x}) = \sum_{k=1}^{n-1} w_k(\mathbf{x}) \mathbf{X}_k \hat{\mathbf{\Psi}}_k(\mathbf{x}) + w_n(\mathbf{x}) \mathbf{X}_n \mathbf{\Psi}_n(\mathbf{x}) \qquad (4)$$

我々は図 2 に示すようにペイントツールを利用して，この重み$\mathbf{w}(\mathbf{x})$を指定する．しかし，隣接する領域間で重みが大きく異なるとペイントした領域の境界付近で流れが不連続になる．そのため，シームレスな画像の合成方法[12]を応用して各領域間の重みを滑らかにする．具体的にはまず，流れの補間方法[4]と同じように，各ペイント領域の境界の周りに補間領域を設定する．そして，文献[12]の方法を用いてその補間領域の重みが滑らかに変化するよう補間する．これにより，不連続を取り除き，滑らかな流れを作成できる．

表 1：格子数と計算時間．T_{cnv}，T_{pyr} そして T_{run} はそれぞれ流れ関数への変換，ラプラシアンピラミッドの構築およびランタイム処理にかかる時間である．T_{sim} は提案手法の結果と同じ格子サイズでシミュレーションを実行した場合の時間である．すべての時間の単位は，秒／フレームである．

図	格子サイズ	T_{cnv}	T_{pyr}	T_{run}	T_{sim}
図 3	256×256	-	-	-	-
図 4	192×192×256	250	6.5	4.0	90

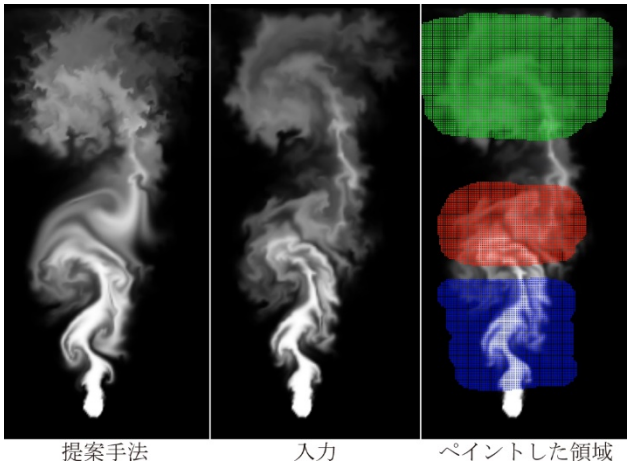

図 3：2 次元の流れへの適用例．

4. 実験結果

提案手法による 2 次元および 3 次元の結果を示す．すべ

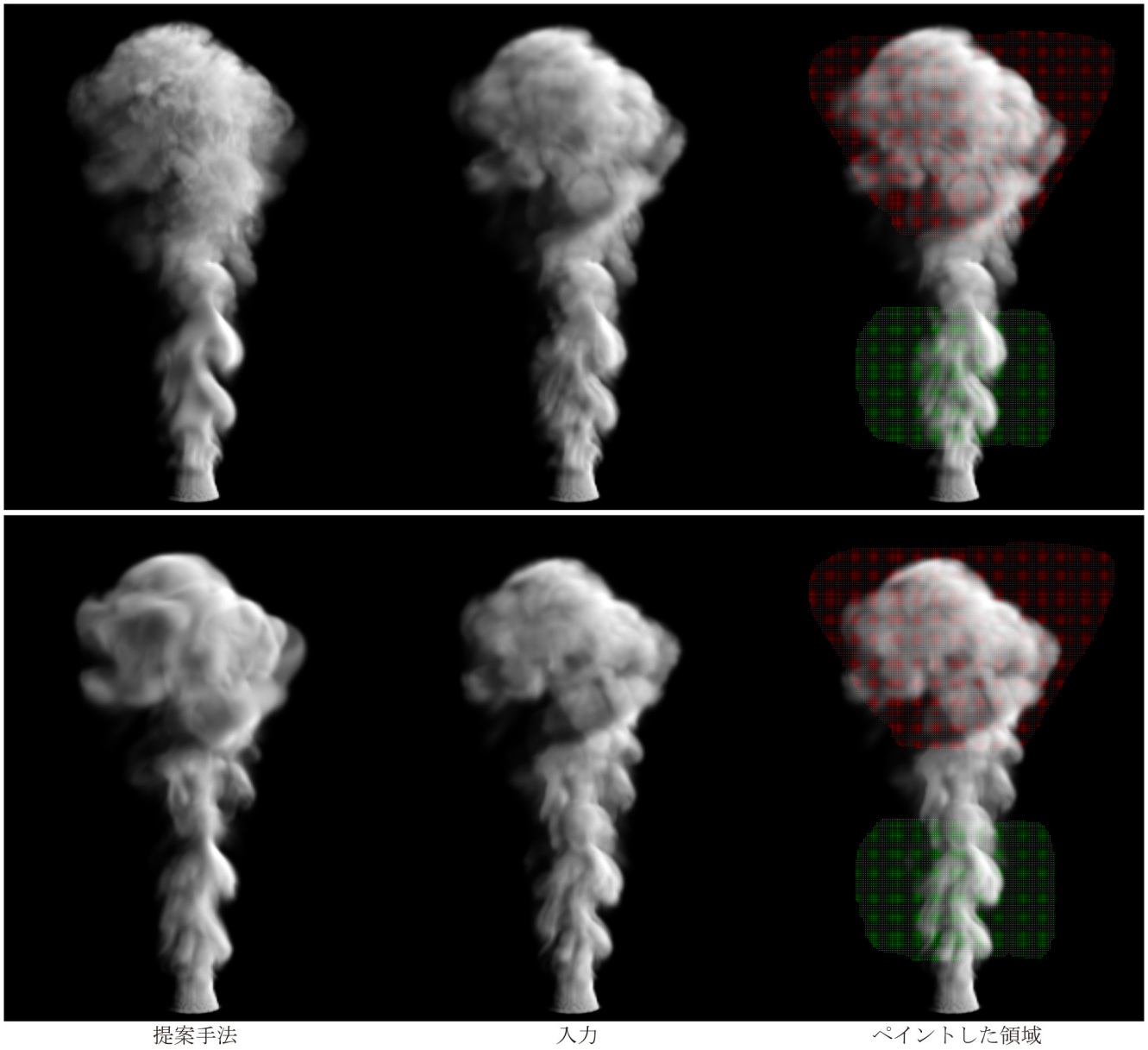

図 4：3 次元の流れへの適用例.

ての結果は，CPU が Intel Core i9-9900K およびメモリが 32GB の PC を用いて計算されている. 格子サイズと計算時間については表 1 にまとめられている．3 次元の場合，速度場から流れ関数に変換する際に 3 次元ベクトル場の各成分に関してポアソン方程式を解く必要があり，この時間はシミュレーションよりも長い時間がかかる．ただし，この変換は一度でよく，また，提案手法のランタイムの計算時間はシミュレーションと比較して十分高速である．すべての結果において指定したラプラシアンピラミッドのレベルの数は 5 である.

図 3 は 2 次元の流れに対して，提案手法により編集を行った例である．2 次元の場合，流れ関数はスカラ場となるため，速度場から流れ関数への変換はシミュレーションよりも高速に計算できる．この例では，3 つの領域を指定しており, 各領域に設定されている重みは, 赤$=\{0,0,0,1,1\}$, 緑$=\{5,3,0,0,1\}$，青$=\{1,1,3,1,1\}$となる．緑の領域では，高周波を含むレベルの係数を大きくしているため，詳細な動きが強調される．逆に赤の領域では上位 3 つのレベルを 0 としており，ローパスフィルタのように高周波成分を取り除いた流れとなっている．青い領域では，中間レベルのバンドの重みを大きくしており，ある程度スケールの大きい動きが強調されている.

次に 3 次元の例を示す（図 4）．3 次元の場合，シミュレーション空間の中心に配置された 2 次元平面上でペイント操作を行い，奥行き方向の重みについては平面上と同じ値を設定する. 図 4 右は指定した領域を示す. 上段の例では，

ⓒ2019 Information Processing Society of Japan

それぞれ重みが，赤= {15, 1, 1, 1, 1}，緑= {0, 0, 1, 1, 1}のように設定されており，赤／緑の領域で詳細な動きが強調／削除されている．一方，下段の例では中間のレベルの影響を増加（赤= {0, 0.5, 5, 1, 1}）／減少（緑= {0.5, 0.1, 0.2, 1, 1}）させており，入力と比べ比較的大きなスケールの動きが変化している．結果に示す通り，提案手法により流れを空間と周波数の両方の領域において局所的に編集できる．

5. まとめと今後の課題

本稿では，流体の流れを空間および周波数両方の領域で編集するための方法を提案した．動画誇張の考え方に基づき，ラプラシアンピラミッドを用いて流れを分解し，各バンドの分布をペイント操作で編集できるようにした．また，結果が常に非圧縮性を満たすよう，速度場を事前に流れ関数へ変換して，編集処理を行った．

今後の課題として，時間方向においても重みを局所的に変更し，かつ滑らかに重みを変化させるような方法の開発があげられる．これには Wu らの方法のように，格子点単位で時間方向の流れ関数の変化を周波数領域に変換し，周波数成分を制御することが考えられる．また，液体や炎など，ほかの流体に適用した場合の結果を確認することも予定している．

謝辞 本研究は JSPS 科研費 JP15H05924 の助成を受けたものである．

参考文献

[1] K. Raveendran, C. Wojtan, N. Thuerey, and G. Turk. Blending liquids. *ACM Transactions on Graphics* 33, 4 (2014), Article 137.
[2] N. Thuerey. Interpolations of Smoke and Liquid Simulations. *ACM Transactions on Graphics* 36, 1 (2016), Article 3.
[3] S. Sato, Y. Dobashi, Y. Yue, K. Iwasaki, and T. Nishita. Incompressibility-preserving deformation for fluid flows using vector potentials. *The Visual Computer* 31, 6 (2015), p. 959–965.
[4] S. Sato, Y. Dobashi, and T. Nishita. Editing fluid animation using flow interpolation. *ACM Transactions on Graphics* 37, 5 (2018), Article 173.
[5] T. Kim and J. Delaney. Subspace fluid re-simulation. *ACM Transactions on Graphics* 32, 4 (2013), Article 62
[6] S. Sato, Y. Dobashi, T. Kim, and T. Nishita. Example-based turbulence style transfer. *ACM Transactions on Graphics* 37, 4 (2018), Article 84.
[7] H.-Y. Wu, M. Rubinstein, E. Shih, J. Guttag, F. Durand, and W. Freeman. Eulerian video magnification for revealing subtle changes in the world. *ACM Transactions on Graphics* 31, 4 (2012), Article 65.
[8] Ryoichi Ando, Nils Thuerey, and Chris Wojtan. A stream function solver for liquid simulations. *ACM Transactions on Graphics* 34, 4 (2015), Article 53.
[9] M. Chu, and N. Thuerey. Data-Driven Synthesis of Smoke Flows with CNN-based Feature Descriptors. *ACM Transactions on Graphics* 36, 4 (2017), Article 14.
[10] Ce Liu, Antonio Torralba, William T. Freeman, Frédo Durand, and Edward H. Adelson. Motion Magnification. *ACM Transactions on Graphics* 24, 3 (2005), p. 519-526.
[11] Neal Wadhwa, Michael Rubinstein, Frédo Durand, and William T. Freeman. Phase-based Video Motion Processing. *ACM Transactions on Graphics* 32, 4 (2013), Article 80.
[12] P. Perez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Transactions on Graphics* 22, 3 (2003), p. 313–318.

ⓒ2019 Information Processing Society of Japan