Example-based Synthesis of Turbulence by Flow Field Style Transfer

Syuhei Sato*
UEI Research

Yoshinori Dobashi
Hokkaido University / UEI Research

Tomoyuki Nishita
UEI Research / Hiroshima Shudo University

Abstract

The computational cost of physically-based fluid simulation for generating realistic animations is expensive. Therefore, it takes a long time for an animator to create the desired animation. To reduce such costs, several methods have been developed that employ an approach in which turbulence is synthesized as a post process. Since, with this approach, global motion can be obtained using low-resolution fluid simulation, realistic animations with the desired behavior can be created at low cost. In the field of image editing, *style transfer* methods, which can efficiently achieve the desired stylization by transferring features of an example image to a user-specified image, have been proposed. We combine these concepts and present a novel style transfer method for turbulence by reusing the existing fluid animations. Our system allows the user to transfer turbulent motion from one flow field to other flow fields. We do this by extending example-based image synthesis methods to the flow field: a patch-based synthesis and an optimization-based texture synthesis. Our method is designed such that the resulting flow fields satisfy the incompressibility condition. Our method can intuitively and easily create high-resolution fluid animations with the desired turbulent motion.

Keywords: fluid simulation, reusing existing fluid animations, incompressibility, texture synthesis, patch-based synthesis

Concepts: •**Computing methodologies** → **Animation;** *Physical simulation;*

1 Introduction

In many entertainment applications, such as movies and computer games, physically-based fluid simulation has generally been used. However, one of the problems of using this is the expensive computational cost, which makes it tedious to adjust the parameters to create the desired fluid animation. The animator must repeat the fluid simulation multiple times to find an appropriate set of parameters that can produce a satisfactory result. To avoid such costly simulation, post-processing approaches for synthesizing turbulence have been proposed: e.g., adding plausible turbulence [Kim et al. 2008], guide-based simulation [Nielsen and Christensen 2010].Using these methods, the time for creating the desired animation can be significantly reduced.

In this paper, we also propose a method for synthesizing turbulence but take a different approach to those used in the methods previously presented. Our method provides a tool for transferring turbulent motion between two input flow fields. The turbulence in a high-resolution flow field can be transferred onto a user designed low-resolution flow field. This transfer is done by extending an example-based image synthesis approach to the flow field. Firstly, we apply a patch-based synthesis to transfer turbulent motion while preserving the global distribution of the turbulence. Secondly, optimization-based texture synthesis is applied to the boundary of each patch to iron out the discontinuities and satisfy the incompressibility condition. We call our approach *turbulence style transfer*. In this paper, we apply this method to smoke simulation defined on a uniform grid. We show several examples of the use

*e-mail:syuhei_sato@dwango.co.jp

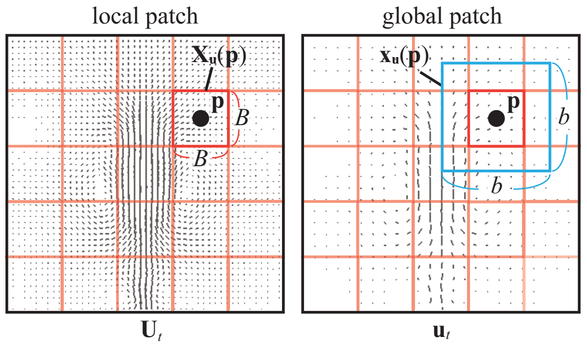

Figure 1: *Definition of local (the orange squares) and global (the blue square) patches. For visual clarity, we show only one global patch, but actually the global patch is defined for each position* $\mathbf{p}$.

of our method, with which the desired turbulence can be intuitively and easily transferred onto a user designed low-resolution simulation.

2 Related Work

Stam developed a semi-Lagrangian method [Stam 1999], which is an unconditionally stable solver for the Navier-Stokes equations. Since this method was introduced, many methods have been proposed for simulating various phenomena. Although realistic animations can be produced with these methods, the user must repeat the simulation and search for good parameter settings until the desired animation is obtained.

Several post-processing approaches for synthesizing turbulence have been proposed. Kim et al. [2008] synthesized detailed turbulent motion onto low-resolution simulation using wavelet noise. Neilsen et al. [2010] controlled a high-resolution fluid simulation according to a user designed low-resolution simulation by solving a minimization problem. Chu et al. [2017] proposed a method based on a convolutional neural network. By contrast, we reuse existing flow fields with the desired turbulent motion, and then the turbulence is transferred onto any other flow field. Therefore, the desired turbulence can be intuitively and easily synthesized just by specifying such flow field.

In the literature on texture synthesis, many optimization-based methods have been proposed. One popular method was proposed by Kwatra et al. [2005]. This method repeats a nearest neighbor search and solving linear system, and then a texture sufficiently similar to an example texture can be synthesized. Kwatra et al. [2007] also proposed a method for synthesizing the texture on a mesh representing a water surface. Narain et al. [2007] extended the method of Kwatra et al. [2007] such that details can be added to a water surface from example images. When multiple images, such as those representing water, foam, and so on, are given to this system, an appropriate image is automatically selected according to the local features of the water mesh. In our method, we focus on smoke defined on a grid. Therefore, those methods that focus on water meshes are not suitable for our purpose. Jamriska et al. [2015] proposed an appearance transfer method for 2D fluid ani-

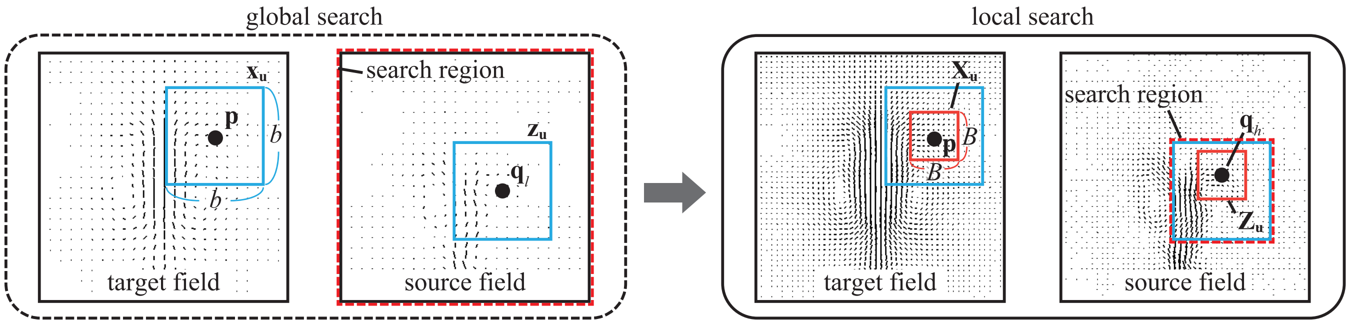

Figure 2: *Two-level patch search. A similar region is globally searched using the global patches (the blue squares), and then locally searched using the local patches (the orange squares). For the global and local search, a search region is shown by the red dotted squares.*

mations based on the above optimization-based texture synthesis. This method distinguishes between the features of the boundary and the inner region of the fluid using an alpha mask. Similar to this method, we create a mask that represents the presence of fluid from a density field and use it for patch detection.

Ma et al. [2009] synthesized the small-scale detail in a motion field from an example image using Kwatra's method [Kwatra et al. 2005]. This motion field is then added to an input low-resolution motion field; however, the detailed motion added is uniform over the entire field. In fluids, the size and distribution of turbulence varies. Therefore, this method [Ma et al. 2009] is not suitable for our purpose. We take into account features of the flow field by calculating the similarity using the velocity and density fields to synthesize plausible turbulence.

3 Our Method

In this section, we describe our example-based turbulence synthesis. For the sake of clarity, we describe the method for a 2D flow field. Two input flow fields are required and we assume that the number of frames for these flow fields is the same, though the method is not restricted to this assumption. In the following description, we use lower and upper case symbols to represent variables related to the low and high resolution grids, respectively.

3.1 Input Flow Fields

The inputs to our method are a low-resolution flow field with the desired global motion (target field) and a high-resolution flow field with the desired turbulent motion (source field). The target and source fields consist of velocity fields ($\mathbf{u}_t(i)$ and $\mathbf{U}_s(i)$) and density masks ($d_t(i)$ and $D_s(i)$), where $i = 1, 2, \cdots, I$ represents the frame number, and I is the number of frames. $d_t(i)$ and $D_s(i)$ are set to 1 for the grid points where the smoke density is larger than D_{th}, and set to 0 otherwise. D_{th} is a threshold for the density which is chosen so that the smoke region is extracted. The grid size and grid interval for the target and source fields are $n \times m$, $N \times M$ and Δx, ΔX, respectively ($n < N$, $m < M$ and $\Delta x > \Delta X$). From these inputs, we evaluate the similarity of the flow and the shape of the smoke.

The source field is firstly downsampled so that the grid size becomes the same as that of the target field ($n \times m$), and the low-frequency components of the velocity and density fields of the source field are obtained, denoted by $\mathbf{u}_s(i)$ and $d_s(i)$ respectively. Next, $\mathbf{u}_t(i)$, $d_t(i)$ and $\mathbf{u}_s(i)$ are linearly upsampled to the resolution of the source field ($N \times M$), denoted as $\mathbf{U}_t(i)$, $D_t(i)$ and $\mathbf{U}_{sl}(i)$, respectively. The high-frequency components of the source field $\mathbf{U}_{sh}(i)$ are then computed by $\mathbf{U}_s(i) - \mathbf{U}_{sl}(i)$.

Next, we subdivide the target field into small square regions, which we call *patches*. $\mathbf{U}_t$ (and D_t) are subdivided into patches, shown as orange squares in Fig. 1, with grid size $B \times B$ and grid interval ΔX. We denote the patches at $\mathbf{p}$ in the velocity and density fields by $\mathbf{X_u}(\mathbf{p})$ and $\mathbf{X}_d(\mathbf{p})$, respectively. We call these *local patches*. In addition, for each center $\mathbf{p}$, a square region whose size and grid interval are $b \times b$ and Δx ($b > B \times \Delta X/\Delta x$) is defined on $\mathbf{u}_t$ and d_t (the blue square in Fig. 1). These patches are denoted as $\mathbf{x_u}(\mathbf{p})$ and $\mathbf{x}_d(\mathbf{p})$ for the velocity and density fields, respectively, and we call these *global patches*. The global patches are used for the global search over the entire simulation space while the local patches are used for a local search inside a global patch for a similar region (see Section 3.2). In the case of 3D flow fields, we define the patches using voxels with a grid size of B^3 or b^3.

3.2 Patch-based Turbulence Synthesis

Our method adds high-frequency turbulent motion to the target field by searching for the best patch in the source field that is most similar to each of the patches in the target field. To efficiently detect the patch, we develop a two-level search algorithm using two resolutions (low and high) as shown in Fig. 2. First, for each local patch, we check the existence of the smoke using $D_t(i)$. When the smoke exists in a patch, the following two-level search process is executed. That is, a similar region is globally detected using the global patches on a low-resolution grid, and the local search is then done inside the region of the detected global patch. However, we found that repeating this two-level search process at each frame independently produced temporal artifacts in the resulting flow field. We address this problem as follows. The global search is done only once at the frame when the smoke firstly appears in the patch and, at the subsequent frames, the global patch is simply translated so that its center coincides with the center of the optimal local patch computed in the previous frame. After searching for and finding the best local patch, we use this to synthesize the high-frequency components $\mathbf{U}_h(i)$.

In the global search process, we find the best global patch in the source patch that is most similar to each of the global patches in the target field (see left in Fig. 2). We search the low-resolution (or downsampled) source field for the position $\mathbf{q}_l$ such that the following energy function is minimized.

$$E_l(\mathbf{p}, \mathbf{q}_l) = ||\mathbf{x_u}(\mathbf{p}) - \mathbf{z_u}(\mathbf{q}_l)||^2 + \alpha||\mathbf{x}_d(\mathbf{p}) - \mathbf{z}_d(\mathbf{q}_l)||^2, \quad (1)$$

where α is a user specified coefficient for adjusting the influence of the second term, the density difference, on the above equation. $\mathbf{z_u}(\mathbf{q}_l)$ and $\mathbf{z}_d(\mathbf{q}_l)$ are $b \times b$ size patches whose center is located at $\mathbf{q}_l$ in the velocity and density fields, $\mathbf{u}_s$ and d_s, respectively, in the downsampled source field. By the above search process, we can

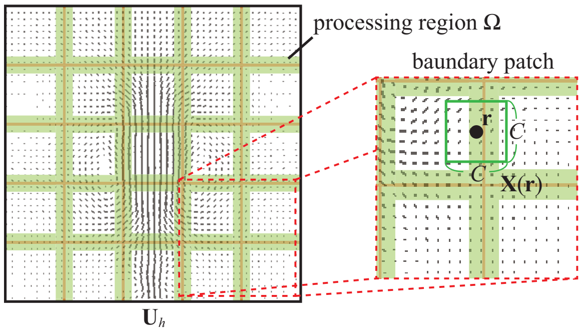

Figure 3: *Definition of a processing region* Ω *(the green shaded regions) and a boundary patch (the green square).*

find the best region in the source field corresponding to each of the global patches in the target region.

Next, in the local search process, we search the region in the source field for the local patch that is most similar to it in the corresponding global patch in the target field (see right in Fig. 2). To take into account temporal coherence, we introduce the high-frequency components of the velocity field $\hat{\mathbf{U}}_h(i)$ obtained by advecting the high-frequency components synthesized in the $(i-1)$-th frame $\mathbf{U}_h(i-1)$ according to the target field $\mathbf{U}_t(i-1)$, and its patch is denoted as $\mathbf{X}_{\hat{\mathbf{u}}}(\mathbf{p})$. Then, we search the search region in the source field (the red dotted square in Fig. 2) for the center position $\mathbf{q}_h$ of the local patch whose high-frequency components are similar to those of $\mathbf{X}_{\hat{\mathbf{u}}}(\mathbf{p})$ by minimizing the following energy function, E_h:

$$\begin{aligned} E_h(\mathbf{p}, \mathbf{q}_h) &= ||\mathbf{X}_{\mathbf{u}}(\mathbf{p}) - \mathbf{Z}_{\mathbf{u}}(\mathbf{q}_h)||^2 \\ &+ \alpha||\mathbf{X}_d(\mathbf{p}) - \mathbf{Z}_d(\mathbf{q}_h)||^2 \\ &+ \beta||\mathbf{X}_{\hat{\mathbf{u}}}(\mathbf{p}) - \mathbf{Z}_{\hat{\mathbf{u}}}(\mathbf{q}_h)||^2, \end{aligned} \tag{2}$$

where β is a user specified coefficient for adjusting the influence of the third term on Eq. 2. $\mathbf{Z}_{\mathbf{u}}(\mathbf{q}_h)$, $\mathbf{Z}_d(\mathbf{q}_h)$ and $\mathbf{Z}_{\hat{\mathbf{u}}}(\mathbf{q}_h)$ are $B \times B$ size patches for $\mathbf{U}_{sl}$, D_s and $\mathbf{U}_{sh}$, respectively, with the center at $\mathbf{q}_h$. Using the above search, the most similar patch is obtained from the source field for each local patch of the target field, and then we synthesize the high-frequency components of the velocity field $\mathbf{U}_h$ using these local patches. However, since we have ignored the spatial continuity between local patches, the velocities are discontinuous at the boundary. Wherefore, we eliminate these discontinuities by a smoothing process described in the next section.

3.3 Smoothing Velocities between Patches

In order to make the velocities smooth at the boundaries between local patches, we apply an optimization-based texture synthesis approach [Kwatra et al. 2005] to $\mathbf{U}_h$. However, since Kwatra's method does not take into account the incompressibility of the fluid, we cannot apply this method to the flow fields in a straightforward manner. We modify the energy function in the original method [Kwatra et al. 2005] to satisfy the incompressibility. The details of this process and our modification are described in the following.

First, a processing region, Ω, with a predefined width is created around the boundaries between each local patch on $\mathbf{U}_h$ (the green shaded regions in Fig. 3), where the width is specified by the user. In this process, only the high-frequency components ($\mathbf{U}_h$ and $\mathbf{U}_{sh}$) are considered. The high-frequency components contain the small-scale turbulent motion and vorticity, which are captured in the spatial derivative of the velocity field, but smooth flow (e.g., laminar flow) is not contained. Therefore, we formulate our energy function using the velocity gradients. In order to satisfy the incompressibility of the flow, the divergence of the velocity is also introduced to the energy function. More concretely, we minimize the following energy function E_b at the boundary region Ω, in order to remove the discontinuities between the local patches.

$$E_b(\mathbf{r}, \mathbf{s}) = \sum_{\mathbf{r},\mathbf{s}} ||\nabla \mathbf{X}(\mathbf{r}) - \nabla \mathbf{Z}(\mathbf{s})||^2 + \gamma(\nabla \cdot \mathbf{X}(\mathbf{r}))^2, \tag{3}$$

where γ is a relative weighting coefficient.$\mathbf{X}(\mathbf{r})$ and $\mathbf{Z}(\mathbf{s})$ are $C \times C$ size patches with centers located at $\mathbf{r}$ and $\mathbf{s}$ on $\mathbf{U}_h$ and $\mathbf{U}_{sh}$, respectively. These patches are called *boundary patch* (the green square in Fig. 3). To minimize Eq.(3), we adopt an Expectation-Maximization like approach, the same as Kwatra et al. [2005].

In the E-step, $\mathbf{X}(\mathbf{r})$ is calculated by solving the linear system of Eq. (3), while keeping $\mathbf{Z}$ fixed at the value obtained in the M-step. This is done by setting the derivation of Eq. (3) with respect to $\mathbf{U}_h$ to zero. To obtain a numerical solution to the linear system, we use the conjugate gradient (CG) method. This linear system is very similar to that used in the previous methods [Sato et al. 2016].Next, in the M-step, we search $\mathbf{Z}(\mathbf{s})$ on $\mathbf{U}_{sh}$ keeping $\mathbf{X}$ fixed at the value estimated in the E-step, such that Eq. (3) is minimized. Since $\mathbf{X}$ is fixed, the second term of Eq. (3) becomes constant in this M-step. Therefore, the energy E_b is calculated using only the first term of Eq. (3). We alternately update $\mathbf{X}$ and $\mathbf{Z}$ by repeating the above E-step and M-step. When the energy transition E_b is sufficiently small, we terminate this smoothing process, and obtain the boundary-smoothed high-frequency components $\mathbf{U}_h$. Finally, the resultant velocity field $\mathbf{U}$ is obtained as $\mathbf{U}_h + \mathbf{U}_t$. To visualize the flow, the smoke density is advected along with $\mathbf{U}$.

4 Results

Figs. 4, 5 show results created using our method. We used a desktop PC with an Intel Core i7-5820K CPU, 32GB memory to compute all the examples. α, β and γ are set to 0.004, 0.5 and 10, respectively. The videos corresponding to these examples can be found in the supplementary material.

Fig. 4 shows results synthesized using various source fields with different turbulent motion, in 2D flow field. Grid size of target and source fields is 24×32 and 192×256. Fig. 4 (a) shows the target field linearly upsampled to 192×256 grid size. Figs. 4 (b1), (c1) and (d1) are source fields. Figs. 4 (b2), (c2) and (d2) are our results created by transferring the turbulent motion of (b1), (c1) and (d1) to (a). The computation time for our method and 192×256 size fluid simulation is 0.7 and 1.0 seconds per frame on average.

Fig. 5 shows various results in 3D flow field. Grid size of target and source fields is $16 \times 16 \times 24$ and $128 \times 128 \times 192$. Fig. 5 (a) shows the target field linearly upsampled to the size of the source field. Figs. 5 (b1) and (c1) are source fields. Figs. 5 (b2) and (c2) are our results created by transferring the turbulent motion of (b1) and (c1) to (a). The computation time for our method and $128 \times 128 \times 192$ size fluid simulation is 270 and 50 seconds per frame on average. In this 3D case, our method is 5 times slower than the simulation. The most expensive process in our method is the search process. However, since the search process for each patch is independent, we expect a significant acceleration would be achieved by using the GPU to execute the search process for each patch in parallel. Therefore, fast computing can expect by implementing the method on GPU. As shown in our 2D and 3D results, the turbulent motion is similar to each source field, and the global motion is preserved to be same with the target field.

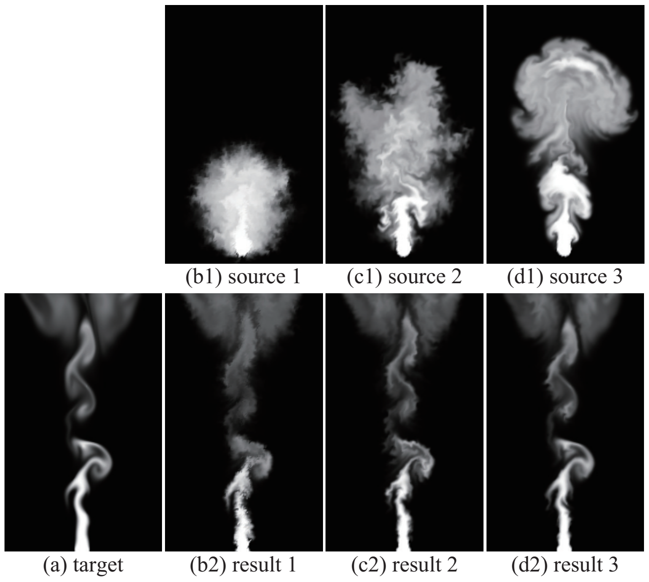
(b1) source 1 (c1) source 2 (d1) source 3
(a) target (b2) result 1 (c2) result 2 (d2) result 3

Figure 4: *Experimental results in 2D flow field.*

5 Conclusions

We have proposed a method for synthesizing turbulent motion by reusing existing flow fields while preserving the incompressibility of the fluid. Our method used patches to synthesize turbulent motion while preserving the global distribution of the turbulence. In addition, we applied an optimization-based texture synthesis approach to smooth the velocity at the boundary of each patch. Furthermore, the energy function for the texture synthesis is modified such that the incompressibility is satisfied.

One of the limitations of our method is that it is difficult to apply it to flow fields with largely different directions and speeds, because we evaluate the velocities directly as described in Sec. 3.2. To address this problem, we plan to evaluate flow fields after rotating and normalizing them. We also plan to extend our method to other types of fluid, such as fire, cloud, and water.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number JP15H05924.

References

CHU, M., AND THUEREY, N. 2017. Data-driven synthesis of smoke flows with cnn-based feature descriptors. *ACM Transactions on Graphics 36*, 4, Article 14.

JAMRISKA, O., FISER, J., ASENTE, P., LU, J., SHECHTMAN, E., AND SYKORA, D. 2015. Lazyfluids: appearance transfer for fluid animations. *ACM Transactions on Graphics 34*, 4, Article 92.

KIM, T., THUREY, N., JAMES, D., AND GROSS, M. 2008. Wavelet turbulence for fluid simulation. *ACM Transactions on Graphics 27*, 3, Article 3.

KWATRA, V., ESSA, I., BOBICK, A., AND KWATRA, N. 2005. Texture optimization for example-based synthesis. *ACM Transactions on Graphics 24*, 3, 795–802.

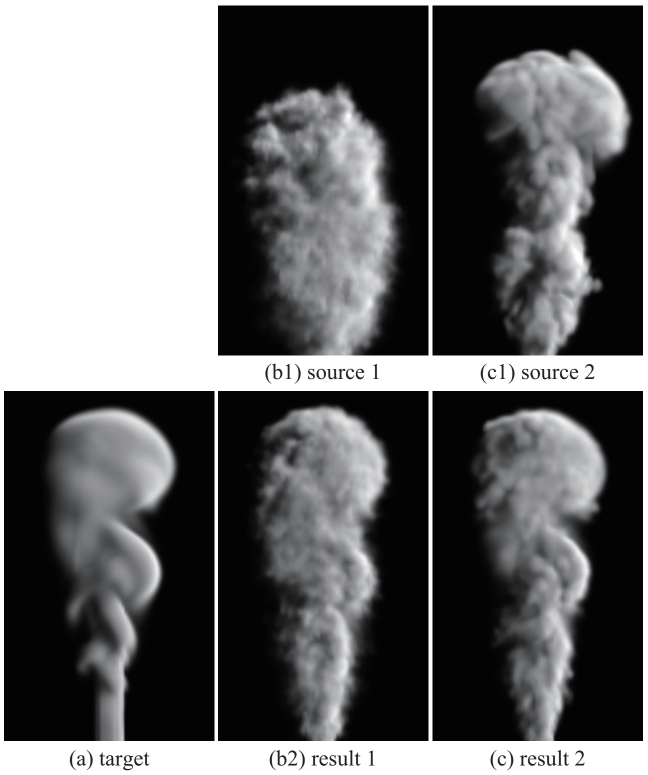
(b1) source 1 (c1) source 2
(a) target (b2) result 1 (c) result 2

Figure 5: *Experimental results in 3D flow field.*

KWATRA, V., ADALSTEINSSON, D., KIM, T., KWATRA, N., CARLSON, M., AND LIN, M. C. 2007. Texturing fluids. *IEEE Transactions on Visualization and Computer Graphics 13*, 5, 939–952.

MA, C., WEI, L., GUO, B., AND ZHOU, K. 2009. Motion field texture synthesis. *ACM Transactions on Graphics 28*, 5, Article 110.

NARAIN, R., KWATRA, V., LEE, H.-P., KIM, T., CARLSON, M., AND LIN, M. C. 2007. Feature-guided dynamic texture synthesis on continuous flows. In *Proceedings of the 18th Eurographics conference on Rendering Techniques*, 361–370.

NIELSEN, M. B., AND CHRISTENSEN, B. B. 2010. Improved variational guiding of smoke animations. *Computer Graphics Forum 29*, 2, 705–712.

SATO, S., DOBASHI, Y., AND NISHITA, T. 2016. A combining method of fluid animations by interpolating flow fields. In *Proceedings of SIGGRAPH Asia 2016 Technical Briefs*, Article 4.

STAM, J. 1999. Stable fluids. In *Proceedings of ACM SIGGRAPH 1999, Annual Conference Series*, 121–128.