DOI: 10.1111/cgf.142639
EUROGRAPHICS 2021 / N. Mitra and I. Viola
(Guest Editors)

*Volume 40 (2021), Number 2*

# Patch Erosion for Deformable Lapped Textures on 3D Fluids

Jonathan Gagnon[1,2], Julián E. Guzmán[1,2], David Mould[3], and Eric Paquette[2]

[1]FOLKS VFX [2]École de technologie supérieure, Montreal, Canada [3]Carleton University, Ottawa, Canada

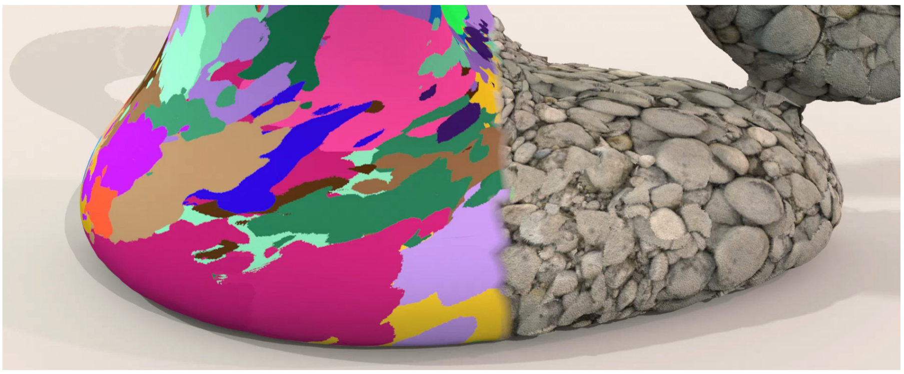

**Figure 1:** *Texturing a fluid animation with eroded deformable lapped textures. Deformable patches are colored on the left and the resulting lapped texture is shown on the right.*

**Abstract**

*We propose an approach to synthesise a texture on an animated fluid free surface using a distortion metric combined with a feature map. Our approach is applied as a post-process to a fluid simulation. We advect deformable patches to move the texture along the fluid flow. The patches are covering the whole surface every frame of the animation in an overlapping fashion. Using lapped textures combined with deformable patches, we successfully remove blending artifact and rigid artifact seen in previous methods. We remain faithful to the texture exemplar by removing distorted patch texels using a patch erosion process. The patch erosion is based on a feature map provided together with the exemplar as inputs to our approach. The erosion favors removing texels toward the boundary of the patch as well as texels corresponding to more distorted regions of the patch. Where texels are removed leaving a gap on the surface, we add new patches below existing ones. The result is an animated texture following the velocity field of the fluid. We compared our results with recent work and our results show that our approach removes ghosting and temporal fading artifacts.*

**CCS Concepts**
• ***Computing methodologies*** → *Texturing;*

## 1. Introduction

Texture mapping is the technique of choice to add details to surfaces. It is largely applied in settings where the surfaces are either rigid or deforming in an isometric fashion. However, fluid animations pose difficult problems with challenges arising from severe deformations of the surface, even including changes in surface topology. Texture mapping under such circumstances can yield significant artifacts, and dealing with these artifacts is still a nontrivial problem with only a few solutions.

Recent fluid texturing methods use patch-based texture synthesis, where patches contain a set of pixels from a texture exemplar [BSM*06, KAK*07, NKL*07, YNBH11]. Blending deformable patches may produce ghosting artifacts. Gagnon et al. [GDP16] used rigid opaque patches; their method significantly reduced ghosting, but the rigid patches were not able to follow the fluid flow closely. Gagnon et al. [GGV*19] proposed deformable patches with alpha blending to combine overlapping patches, but this method suffered from ghosting. Moreover, the fading of distorted patches was not always consistent with the velocity field, especially in areas with little fluid motion. The introduction of new patches and the removal of distorted patches over multiple frames introduced temporal inconsistencies in their method. Our objective is to create a texture on a fluid animation with neither ghosting artifacts nor temporal inconsistencies when the fluid barely moves.

Our solution combines lapped textures and deformable patches. To remove distorted patches, instead of fading out the whole patch, we apply an erosion process to remove distorted patch texels using a feature map. The main contributions of the proposed texture synthesis approach can be summarized as follows:

- Adaptation of the lapped textures framework for deformable patches on fluids;
- Feature-aware patch erosion preserving consistency to the texture exemplar;
- Automatic creation of a multitude of lapped texture shapes based on a feature map;
- Patch distribution update based on texture erosion.

## 2. Related Work

Neural networks have been used for style transfer on 2D smoke simulation [CKAS20] and later extended on volumetric fluid animation, changing also the shape of 3D fluid surfaces [KAGS20]. Where these methods focus on volumetric data, our approach concentrates on surface texture synthesis.

Kwatra et al. [KEBK05] and Jamriska et al. [JFA*15] proposed an animated 2D texture synthesis using a best-match search. However, the resulting animated texture looks stiff as the search windows do not deform. To avoid stiffness, it is possible to use textured patch advection, as proposed by Neyret et al. [Ney03] and Yu et al. [YNBH11]; in these methods, patches can deform, resolving the stiffness concern. Overly distorted patches are replaced by undistorted ones. Alpha blending is used to phase out a distorted patch over time; unfortunately, the blending can create ghosting artifacts where structural patterns are blurred.

Only a few methods deal with 3D fluids. Bargteil et al. [BSM*06], Kwatra et al. [KAK*07], and Narain et al. [NKL*07] extended the work of Kwatra et al. [KEBK05] for a best-match search texture synthesis on a 3D mesh. Unfortunately, they inherited the limitations of the 2D texture synthesis of Kwatra et al. [KEBK05], with stiff-looking texture animations. Another limitation is inherent in best-match search methods: larger exemplars drastically increase computation times. Gagnon et al. [GDP16] proposed rigid patches advected on the free surface. Their patch-based method does not use best-match searches, avoiding the problem of texture exemplar resolution. However, the resulting texture animation shows stiff texture patterns. To deal with the stiffness problem, Gagnon et al. [GGV*19] extended the deformable patches method of Yu et al. [YNBH11] to 3D fluids. They also added a new sampling method to reduce the number of patches in order to reduce ghosting, but it is still an issue.

Our goal is to take advantage of deformable patches to avoid stiffness, and lapped textures to reduce blending. Lapped textures is a patch-based texture synthesis method [PFH00], successfully used to texture fluids by Gagnon et al. [GDP16]. We also improve upon deformable patches texture synthesis [YNBH11, GGV*19], replacing the patch fading by a feature-aware erosion method that removes distorted regions of patches.

## 3. Overview

Our approach is designed to add texture details to the surface of a fluid. It is applied as a post-process and does not interfere with the fluid simulation. We take as input an animated mesh and a velocity field. The animated mesh of the fluid is typically a completely different mesh at every frame (e.g., surface reconstruction from an implicit function). The velocity field has no restrictions other than allowing sampling at arbitrary locations on the fluid surface.

The steps of the approach are illustrated in Fig. 2. Our approach uses patch-based synthesis, with patches being textured with a texture exemplar. Patches are represented as small polygon meshes distributed over the whole surface. We cover the fluid surface with overlapping textured patches, applying a Poisson disk distribution on the surface of the fluid at the first frame of the animation.

The polygonal patches are advected with the velocity field. Advection is done on a per-vertex basis, and consequently the patches deform over time, causing the texture to become distorted. We remove distorted textured regions of patches using a feature-aware erosion. Features are defined by a feature map provided as an input by the user. The erosion considers three criteria: removal of lower-importance features, removal of distorted regions of patches, and removal progressing from the outside toward the inside of the patches. At each frame, an output texture is synthesized over the entire input mesh with the layered patches.

The patch distribution is updated during the synthesis to ensure full coverage of the surface. Any time a surface texel has no covering patch texel, a new patch is added. Also, if a patch is not used in the texture synthesis, either because it is underneath other patches or it has no remaining texels, then it is removed. In the next sections, deformable patches are discussed (Sec. 4), followed by the texture synthesis using feature-aware erosion (Sec. 5).

## 4. Deformable Patches

A patch is an open triangle mesh covering a local region from the surface of the fluid. Together, the patches cover the entire fluid surface. Patch vertices are advected by the fluid motion, causing initially well-shaped patches to deform over time; excessively deformed patches are progressively removed, using a process we detail in Sec. 5.2.

We initialize the set of patches at the first frame of the animation, placing patch centers on the fluid surface with the Poisson distribution of Gagnon et al. [GGV*19]. Each patch is then created by copying, from the mesh of the fluid, a subset of the polygons near the patch center, as illustrated in Fig. 3a. That is, the polygons for the patch perfectly match a portion of the fluid mesh at the moment when the patch is created. The subset of the fluid mesh is determined by proximity to the patch center $p$: we take polygons within Euclidean distance $r$, where $r$ is the user-specified patch radius. Of the nearby polygons, we exclude those that face away from the surface normal at $p$. With these criteria, each patch is guaranteed to contain at least one triangle, though most patches are considerably larger. After creation, the mesh topology of each patch is independent of the mesh topology of the fluid simulation.

The polygonal patches are flattened with the parameterization of

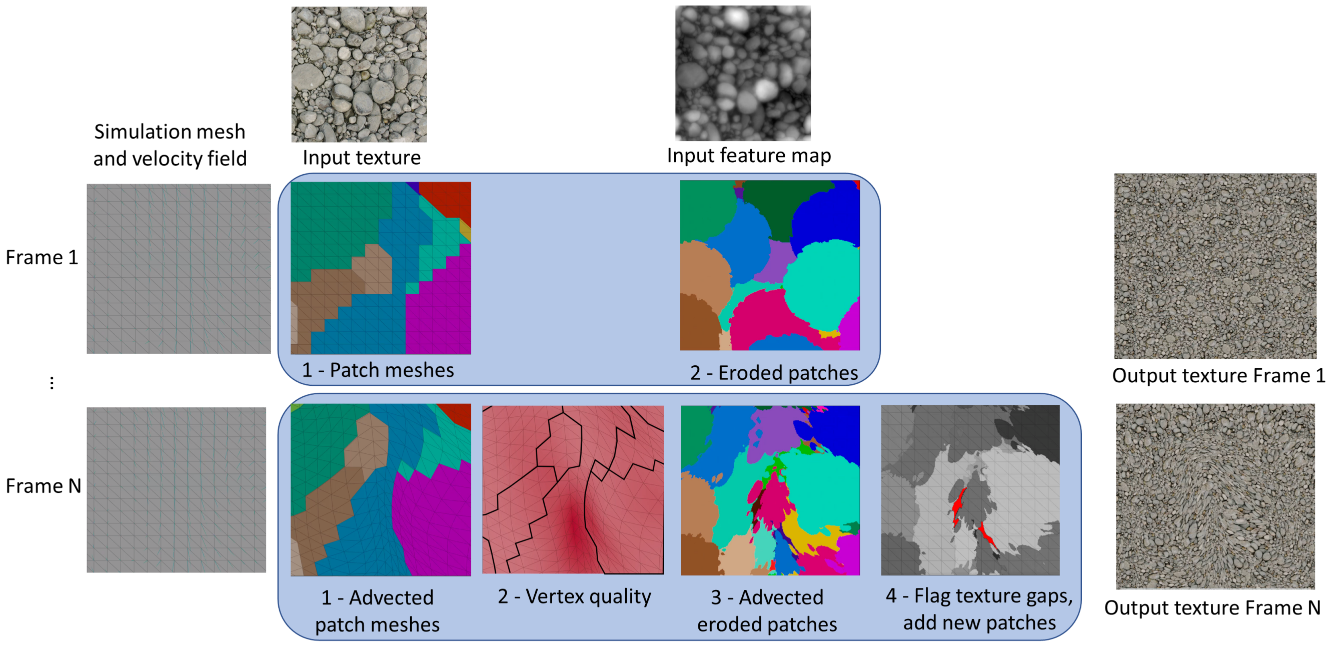

**Figure 2:** *Overview of our approach.*

Lévy et al. [LPRM02]. Each flattened patch thus has associated *uv* coordinates, used to map the exemplar onto the patch. The patch is translated so that the position corresponding to $p$ is at the center of the *uv* coordinate space. The outcome is illustrated in Fig 3b.

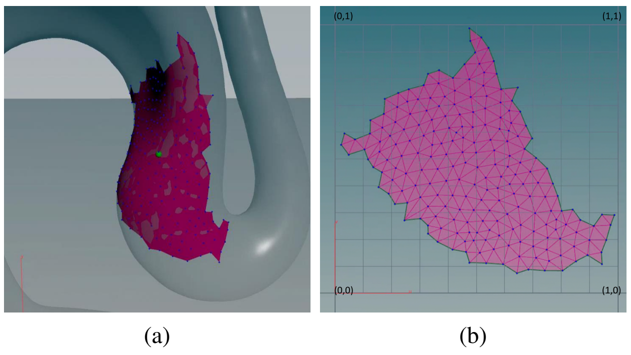

**Figure 3:** *Patch creation using polygons from the mesh of the fluid (a) around position p (highlighted in green) and the resulting uv coordinates (b) after flattening.*

Throughout the animation, we advect the individual vertices of the deformable patches according to the velocity of the fluid. To ensure that the patch matches the fluid surface, patch vertices are projected onto the fluid mesh, finding the closest location on the mesh to each vertex. After advection, vertices are sometimes too far from the surface, for example when there are topological changes due to splashes and droplets. Such vertices, and their related polygons, are removed from the patch. In this paper, we removed vertices at a distance greater than $r/4$ from the surface.

Rejecting any portion of a patch – whether from polygons pointing away from the normal at $p$, erosion (Sec. 5.2), or removing vertices far from the surface – can produce a gap, where part of the fluid surface is not covered by a patch. Any gaps in coverage will be detected when filling our atlas (Sec. 5.1), and we add new patches to cover gaps. Newly added patches are full size, covering a larger region than just the detected hole. We thus avoid repeatedly introducing new patches in regions where a hole is growing.

## 5. Texture Synthesis

Our texture synthesis computation is inspired by the lapped textures method. For every frame of an animation, we compute an output texture atlas covering the whole surface of the fluid. This section describes how the texture is computed using the overlapping patches (Sec. 5.1) and how to remove distorted patch texels using feature-aware erosion (Sec. 5.2).

### 5.1. Lapped Textures

Inspired by the lapped textures method [PFH00], our approach relies on a set of overlapping patches covering the 3D mesh. While Praun et al. [PFH00] use a static and manually created patch mask, our mask is dynamic, automatically derived from a patch erosion process. As such, for each patch, the set of texels considered as *valid* (not eroded) changes over the animation.

An output texture atlas is computed with the method of Lévy et al. [LPRM02]. For each output texel $c$, we go through all patches that overlap $c$. These patches are sorted: the older patches are on top and the newest ones are below. Beginning with the top-most patch overlapping $c$, if the patch texel overlapping $c$ is flagged as eroded, the patch is rejected and we continue to the next patch below. When we find the first patch that is not rejected, we set the color of texel $c$ to the corresponding texel color from the patch.

Should all patches be rejected for output texel $c$, we add a new patch as described in Sec. 4. The center $p$ of the new patch on the fluid mesh is set to the location of the uncovered texel $c$. This patch distribution update is inspired by that of Gagnon et al. [GDP16].

In contrast to the transparent patch texels of Gagnon et al. [GDP16], our patch texels are either fully visible or rejected by the erosion process. This prevents any ghosting artifacts that might be caused by alpha blending.

### 5.2. Feature-Aware Patch Erosion

Repeated advection and projection of patch vertices can cause a patch to become distorted, and we remove regions (texels) of the patch that have undergone excessive distortion. To select the portions to remove, we combine three strategies: (1) removal that considers the features of the texture exemplar, (2) removal of regions close to distorted patch vertices, and (3) removal progressing from the outside toward the inside of the patches.

The features of the texture exemplar are defined by a feature map. The feature map could come from various sources such as a feature distance [LH06], a height map, or it could be hand-created by an artist. For the results shown in this paper, we relied on height maps. The feature map contains values in the range $[0,1]$, and the erosion will first favor regions with lowest values. An example of feature map values is illustrated in Fig. 4.

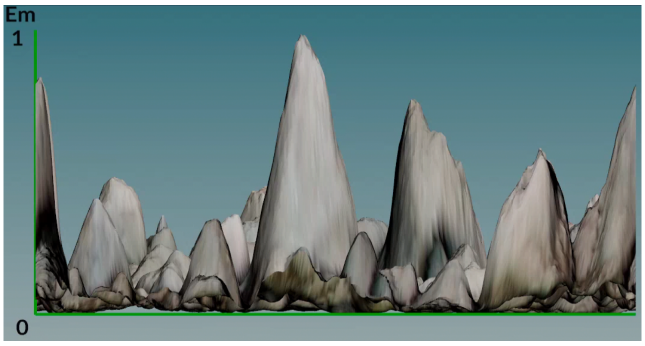

**Figure 4:** *Side view of a feature map $F_m$.*

On each frame of the animation, patches are advected and thus deform. As the simulation proceeds, the initially well-shaped patches become distorted as vertices move in different directions and at different speed following the velocity field. Deformed patches distort the texture features of the exemplar. With the goal of identifying distorted patch regions that we will remove, we evaluate the quality $Q_T$ of patch triangles. We compute the triangle quality with the same distortion metric as in other papers [SCOGL02, YNBH11, GGV*19]:

$$Q_T = max((\delta_{max} - \delta_t)/(\delta_{max} - 1), 0), \qquad (1)$$

where $\delta_{max}$ is a constant defined by the user. We consider a triangle to be too distorted when its distortion $\delta_t$ exceeds a maximal distortion $\delta_{max} = 3$ . We then compute a per vertex quality $Q_V$ by averaging $Q_T$ from the neighboring triangles.

As we saw in Sec. 4, we might reject polygons within the radius $r$ and thus get an irregular patch shape as seen in Fig. 5. When

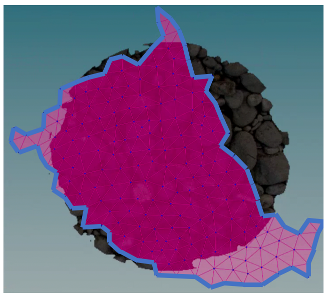

**Figure 5:** *Example of a patch mesh in uv domain, where the boundary ($B_V = 0$) is shown as the blue line around the patch.*

such a patch is not distorted ($Q_V = 1$), we will not erode the patch and the boundary of the rendered patch will not be related to the features, but to the polygonal patch boundary instead. To avoid this, we supplement the vertex quality with a boundary value $B_V$ that drops to zero at the boundary edges of the mesh:

$$B_V = \begin{cases} 1, & V \text{ is an interior vertex of the patch} \\ 1, & V \text{ is within } r/4 \text{ from } p \\ 0, & V \text{ is a boundary vertex further than } r/4 \text{ from } p \end{cases} . \qquad (2)$$

The second case is used to avoid eroding the center $p$ of the patch. Fig. 6 shows an example where, very close to $p$, polygons are rejected since they are pointing away from the normal vector at $p$. If we had $B_V = 0$ for one or more of the vertices of the triangle containing $p$, the interpolation of $B_V$ could lead to a very small value near $p$. In such a case, the same texel that triggered the patch creation could be eroded, leading to an infinite loop constantly failing to add a patch with a valid texel at $p$. We avoid this with the second case, enforcing $B_V = 1$ within a small distance from the patch center. Note that in all circumstances, the radius $r$ is always fixed and the same for all patches.

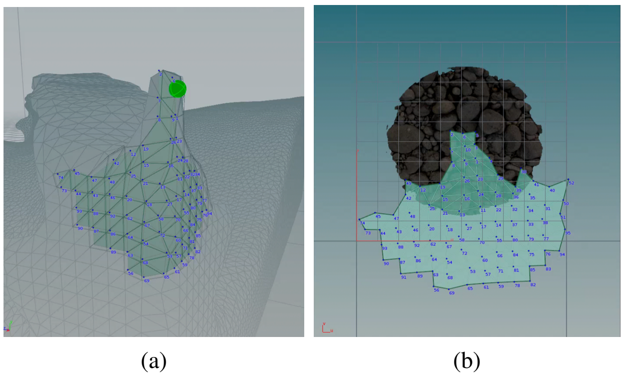

**Figure 6:** *Patch in 3D (a), where the corresponding patch center p in the uv domain (b) is very close to the patch boundary edges.*

Similarly to Yu et al. [YNBH11], we combine quality and boundary values in a vertex weight $w_V$:

$$w_V = Q_V B_V. \tag{3}$$

The last piece of our erosion strategy favors erosion from the outside toward the inside of the patch. We define $D(u,v)$ based on the $uv$ coordinates: $D(u,v) = \max(0, 1-||(0.5,0.5)-(u,v)||/0.5)$. Thus $D(u,v)$ is 1 at the middle of the parameterization and decreases to zero toward the limits of the $uv$ space.

We combine our three erosion criteria into the following erosion selection:

```
if ( $\frac{\sqrt{F_m(u,v)+D(u,v)}}{\max_{(s,t)}\left(\sqrt{F_m(s,t)+D(s,t)}\right)} - \varepsilon < 1 - w(u,v)$ ),
    erode
else
    keep
```

where $(u,v)$ is the texel position, $F_m(u,v)$ is the feature map value, and $w(u,v)$ is the texel weight interpolated from the per-vertex weights (Eq. 3) of the enclosing triangle. The square root over the feature map and the distance ensures that the area of the patch will be proportional to $1-w(u,v)$. By normalizing the square root by its maximal value over all texels, we keep the values of this term in the range of $[0,1]$, which is the same range as the weight. Fig. 7 shows the global shape of the normalized square-root term. The final element in our selection criterion is $\varepsilon \in ]0,1[$. By subtracting a small value $\varepsilon$ (we use $\varepsilon = 0.1$ for the results in this paper), all patches are initially eroded. This ensures that the boundary of the rendered patch has the irregular shape necessary for the lapped texture technique to hide the patch seams.

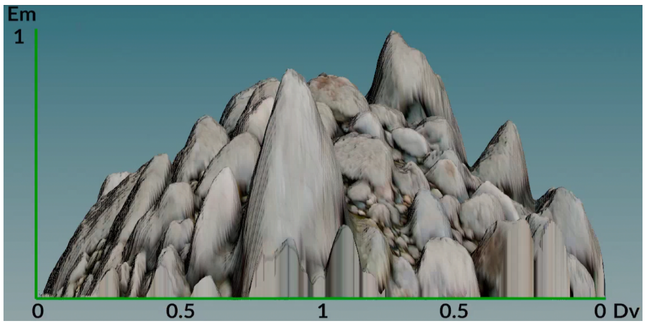

**Figure 7:** *Normalized $\sqrt{F_m(u,v)+D(u,v)}$ which is compared to $1-w(u,v)$ in the erosion process.*

In Fig 8 and the accompanying video, we show the erosion of a single patch. As the patch deforms, the erosion removes texels where the distortion is higher (on the right side for the patch seen in Fig 8d). A patch remains in use as long as at least one of its texels is used in the synthesized texture.

## 6. Results

We have validated our approach using several fluid simulation scenarios similar on those from related work:

**Table 1:** *Statistics of our examples. All times are in seconds per frame. Other steps in our process (included in the Total column), take very little time compared to the patch creation and update.*

| | Polys | Patches | Advection | Patch creation | Patch update | Total |
|---|---|---|---|---|---|---|
| Dam break | 85k | 4881 | 0.74 | 34.89 | 74.23 | 110.95 |
| Lava | 271k | 4664 | 2.07 | 96.69 | 197.39 | 296.15 |
| Viscous drop | 61k | 617 | 0.67 | 11.96 | 33.20 | 45.83 |

- 2D fluid with split and merge, as in the paper of Gagnon et al. [GGV*19] (Fig. 9 and 15)
- 3D viscous drop, as in the papers of Gagnon et al. [GDP16, GGV*19] (Fig. 10, 13 and 14)
- 3D liquid dam break, as in the papers of Gagnon et al. [GDP16, GGV*19] (Fig. 11)
- 3D lava drop, as in the papers of Gagnon et al. [GDP16, GGV*19] (Fig. 12)

As can be seen in the figures, as well as the accompanying video, our approach works well on 2D examples (Fig. 9), with 3D viscous fluids (Figs. 10), and with 3D fluids with no viscosity (Fig. 11). We also tested our approach with several texture types from structured to stochastic texture exemplars, as illustrated in Fig. 15. Table 1 reports statistics for some of our test cases. The patch and atlas creation steps are computed in parallel (on 12 cores in our tests). Atlas resolution was typically 6k$^2$. Our tests were conducted on an Intel Xeon® Silver 4214 CPU. The majority of the time is taken by the patch update.

In the comparison with Gagnon et al. [GDP16] and Gagnon et al. [GGV*19], we can see that our approach has no rigid artifact nor ghosting. The only visible artifact may be the visible patch boundaries when using large patches. We can also see that, when the texture exemplar has irregular patterns, such as pebbles or lava, our approach provides a significant visual improvement.

Fig. 13 compares our results with the method of Gagnon et al. [GDP16]. Their results show limited but noticeable distortion in curved regions. They use an orthogonal projection on each patch, for each frame of the animation, which can create more distortion than our flattening approach using Least Squares Conformal Maps [LPRM02]. With their method, thanks to the use of rigid patches, on a still frame, we can see clearly every feature from the texture exemplar. With our approach we can notice more deformations. However, when looking at the animation of Gagnon et al. [GDP16] in the accompanying video, it is possible to identify individual patches: the rigid patches produce a visual artifact. Our approach is free of such artifacts thanks to the deformable patches.

Fig. 14 compares our results and those of Gagnon et al. [GGV*19]. Their results contain ghosting artifacts since they blend all overlapping patches to compute each texel. We are better able to preserve the exemplar structure; in the previous method, ghosting gives the impression of an incorrect feature size. Moreover, as illustrated in the accompanying video, the fading of the patches can be visible when the fluid is slowing down, giving the impression that the update does not follow the velocity field. Conversely, our approach does not have any ghosting since only one

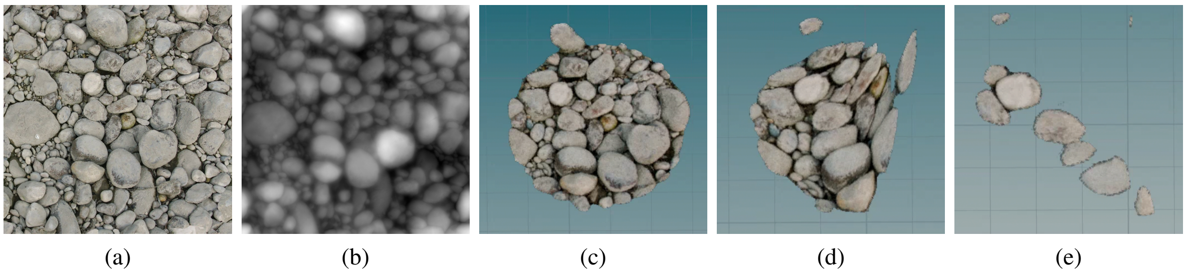

**Figure 8:** *A patch example using a pebble texture exemplar (a) with its feature map (b). We see the progression of the erosion when the patch has no distortion (c), when it is partially distorted (d), and when it is almost completely distorted (e).*

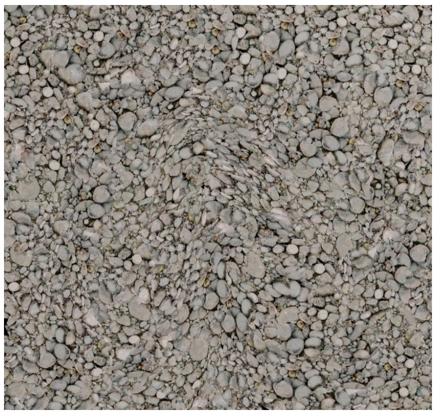

**Figure 9:** *2D fluid with split and merge using texture exemplar and feature map from Fig. 8 and the resulting texture synthesis.*

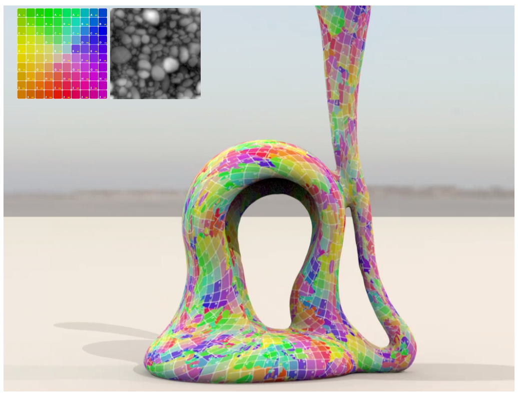

**Figure 10:** *Viscous drop using colored squares as the texture exemplar. This example uses the same feature map as Fig. 8.*

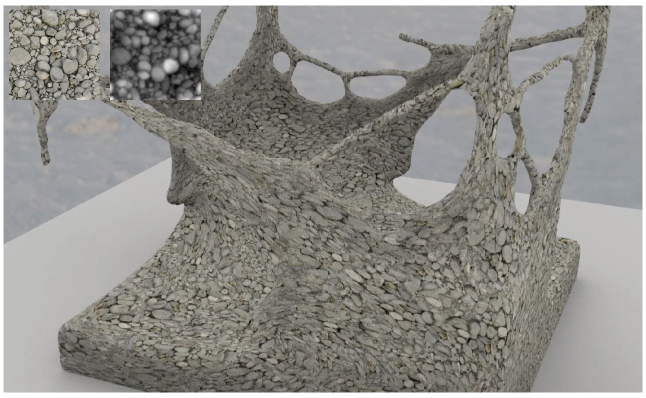

**Figure 11:** *Dam break simulation using a pebbles texture exemplar.*

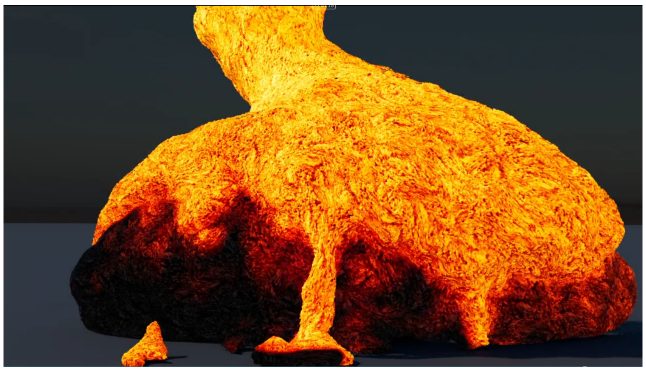

**Figure 12:** *Result of our approach on a lava simulation. We synthesized both a color and a displacement texture.*

patch texel is used for each output texel. Since we do not fade patches in or out using dynamic transparency, we also do not have visible fading that does not follow the velocity field. Our patch distribution update is also different as we do not have to delete patches too close to each other. Furthermore, we do not need to advect trackers with the patches as past methods did [YNBH11,GGV*19].

With the deformable lapped texture representation, the texture conforms well to the movement of the fluid surface. Fig. 16 and the accompanying video compare the fluid's velocity field (shown in blue) with the optical flow of the rendering (shown in green). The two fields are in close agreement, demonstrating that the texture synthesis is able to match the liquid flow.

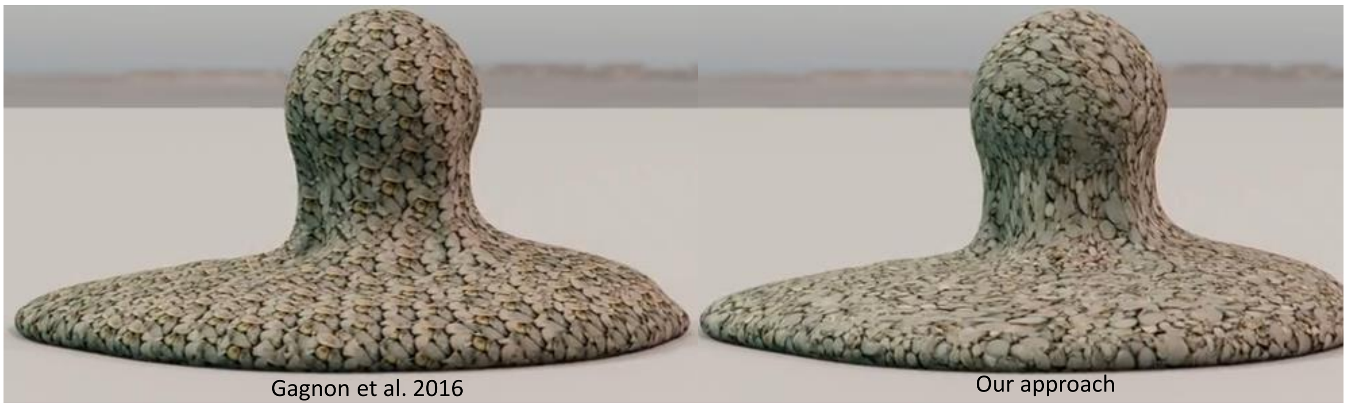

**Figure 13:** *Comparison between Gagnon et al. [GDP16] (left) and our approach (right).*

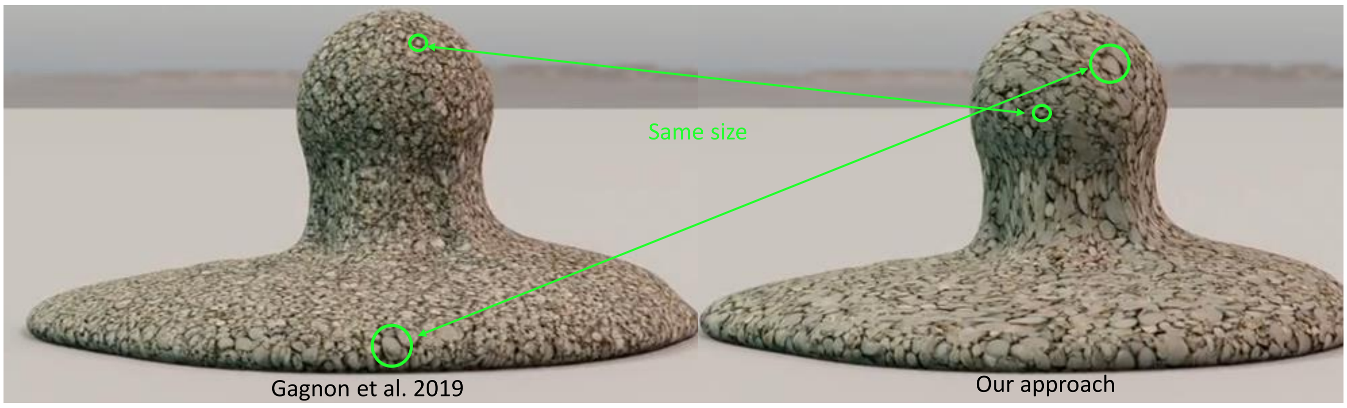

**Figure 14:** *Comparison between Gagnon et al. [GGV*19] (left) and our approach (right).*

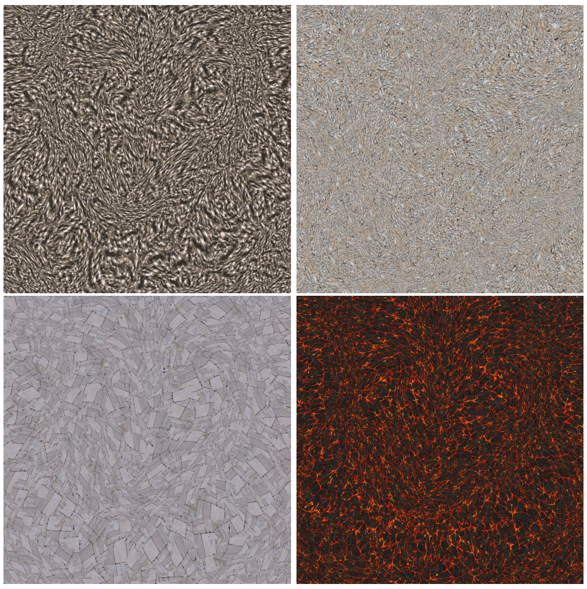

**Figure 15:** *Split and merge test with various texture exemplars.*

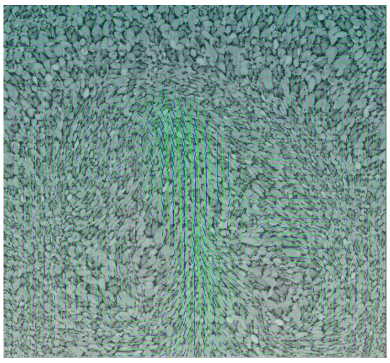

**Figure 16:** *Test case demonstrating that the optical flow of the texture synthesis matches the fluid simulation's velocity field.*

## 7. Limitations

While our approach solves several problems (including ghosting, stiffness, and exemplar resolution), it has some limitations. Our approach relies on the exemplar texture having an accompanying feature map. When a height map is available, we can use that as a reasonable proxy. Arbitrary textures, however, may not have a feature map available, and hence the user must separately draw or compute one (e.g., using feature distance [LH06]).

Regarding the mesh of the animated fluid, our approach is flexible and would work as well if the mesh is consistent or completely different from frame to frame. Nevertheless, our approach imposes some restrictions on the triangle mesh of the animated fluid, requiring reasonably homogeneous triangles. Flat regions of the fluid cannot be represented with larger triangles as this would be incompatible with the patch creation process. Moreover, elongated patch triangles will have a faster falloff in quality as they deform following the velocity field, and will unnecessarily accelerate the erosion. In our implementation, we used SideFX$^{TM}$ Houdini to create the mesh of the fluid, and we used the remesh operator to obtain triangles closer to the equilateral shape.

Another limitation is due to our texel removal strategy. When a patch is distorted and parts of it are being removed, it is sometimes possible to notice the moving boundary, especially when the patches are relatively big. The texel removal produces an illusion of motion as some portions of a patch are eliminated to reveal the patch beneath. This can be seen in the accompanying video. Moreover, the erosion toward the boundary could result in a different look if the exemplar is significantly different at the center compared to the boundary. Using stationary exemplars, as we did in most of our examples, prevents this issue.

Another limitation comes from the patch update process. When we need to add a patch, it is added at the location of the first texel of the atlas that is not covered by any texel from the patches. However, this local strategy might be outperformed by a global strategy that takes into account all missing texels for the current frame. Future work could explore alternative distribution processes, but we have not noticed quality issues resulting from our current design.

## 8. Conclusion

In this paper, we presented a new patch-based approach for synthesizing textures on fluid surfaces. Our approach extends lapped textures [PFH00] to fluids, building on past work by Gagnon et al. [GDP16]. We use a feature-aware patch erosion approach in order to better preserve similarity to the exemplar. Using erosion rather than fading eliminates ghosting artifacts. Further, unlike methods depending on temporal fading, our texture stops changing when the underlying fluid stops moving. We showed numerous comparisons with previous work on fluid textures. We confirmed that our approach is coherent spatially and temporally by showing that the surface texture follows the velocity field of the fluid simulation, comparing the optical flow with the fluid velocities.

There are still some opportunities for future work. We would like to investigate more deeply the patch replacement strategy, ensuring that we get good surface coverage without excessive number of patches. Future work could explore automatic methods for computing a feature map from an exemplar. Finally, texturing surfaces with dynamic and non-homogeneous textures is an area of interest.

## 9. Acknowledgements

This work was funded by FOLKS VFX. We thank the employees and R&D team of FOLKS VFX for their feedback and involvement throughout the project. We would also like to thank SideFX$^{TM}$ for providing Houdini$^{TM}$ licenses for research.

## References

[BSM*06] Bargteil A. W., Sin F., Michaels J. E., Goktekin T. G., O'Brien J. F.: A texture synthesis method for liquid animations. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (ACM, Sept 2006). 1, 2

[CKAS20] Christen F., Kim B., Azevedo V. C., Solenthaler B.: Neural smoke stylization with color transfer. In *41st Annual Conference of the European Association for Computer Graphics, Eurographics 2020 - Short Papers, Norrköping, Sweden, May 25-29, 2020 [online only]* (2020), Wilkie A., Banterle F., (Eds.), Eurographics Association, pp. 49–52. 2

[GDP16] Gagnon J., Dagenais F., Paquette E.: Dynamic lapped texture for fluid simulations. *Vis. Comput. 32*, 6-8 (June 2016), 901–909. 1, 2, 4, 5, 7, 8

[GGV*19] Gagnon J. J., Guzmán J. J. E., Vervondel V. V., Dagenais F. F., Mould D., Paquette E. E.: Distribution update of deformable patches for texture synthesis on the free surface of fluids. *Computer Graphics Forum 38*, 7 (Oct. 2019), 491–500. 1, 2, 4, 5, 6, 7

[JFA*15] Jamriška O., Fišer J., Asente P., Lu J., Shechtman E., Sýkora D.: Lazyfluids: Appearance transfer for fluid animations. *ACM Trans. on Graph. 34*, 4 (2015). 2

[KAGS20] Kim B., Azevedo V. C., Gross M., Solenthaler B.: Lagrangian neural style transfer for fluids. *ACM Trans. Graph. 39*, 4 (July 2020). 2

[KAK*07] Kwatra V., Adalsteinsson D., Kim T., Kwatra N., Carlson M., Lin M.: Texturing fluids. *IEEE Trans. Visualization and Computer Graphics 13*, 5 (2007), 939–952. 1, 2

[KEBK05] Kwatra V., Essa I., Bobick A., Kwatra N.: Texture optimization for example-based synthesis. *ACM Trans. Graph. 24* (July 2005), 795–802. 2

[LH06] Lefebvre S., Hoppe H.: Appearance-space texture synthesis. *ACM Trans. Graph. 25*, 3 (July 2006), 541–548. 4, 8

[LPRM02] Lévy B., Petitjean S., Ray N., Maillot J.: Least squares conformal maps for automatic texture atlas generation. In *IN SIGGRAPH 02 CONFERENCE PROCEEDINGS* (2002), pp. 362–371.

[Ney03] Neyret F.: Advected textures. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2003), Eurographics Association, pp. 147–153. 2

[NKL*07] Narain R., Kwatra V., Lee H.-P., Kim T., Carlson M., Lin M.: Feature-guided dynamic texture synthesis on continuous flows. In *EGSR '07* (2007). 1, 2

[PFH00] Praun E., Finkelstein A., Hoppe H.: Lapped textures. In *Proceedings of SIGGRAPH '00* (2000), Annual Conference Series, pp. 465–470. 2, 3, 8

[SCOGL02] Sorkine O., Cohen-Or D., Goldenthal R., Lischinski D.: Bounded-distortion piecewise mesh parameterization. In *Proceedings of the conference on Visualization '02* (Washington, DC, USA, 2002), VIS '02, IEEE Computer Society, pp. 355–362. 4

[YNBH11] Yu Q., Neyret F., Bruneton E., Holzschuch N.: Lagrangian texture advection: Preserving both spectrum and velocity field. *IEEE Trans. Visualization and Computer Graphics 17*, 11 (Nov. 2011), 1612 –1623. 1, 2, 4, 5, 6