Procedural Fluid Textures

Sean C McDuffee
Blue Sky Studios
scmcduff@blueskystudios.com

Maurice van Swaijj
Blue Sky Studios
maurice@blueskystudios.com

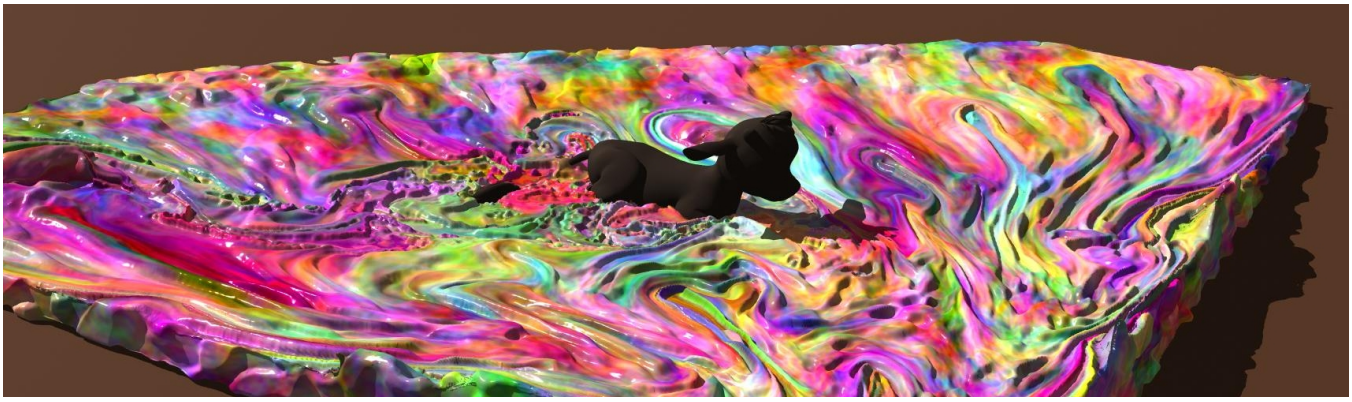

Figure 1: Image Credit: ©2017 Twentieth Century Fox Film Corporation. All rights reserved

ABSTRACT

We present an efficient system for synthesizing textures over fluid surfaces in a solid texturing context. The technique is simple and intuitive for artists using modern, commercially available fluid simulators. Instead of working with 2D surface maps like other fluid texture synthesis approaches, we advect 3D reference space transforms with the fluid simulation. The reference transforms are then projected onto the final surface mesh with a radius of influence control, and used for solid texturing lookup. Ray intersections of the fluid surface interpolate any transforms with overlapping control radii to determine the reference lookup of the solid texture. The final texture exhibits excellent spatial and temporal coherence with none of the artifacts that plagued previous map based approaches.

CCS CONCEPTS

• **Computing methodologies** → **Physical simulation**; *Procedural animation*; *Ray tracing*; *Texturing*;

KEYWORDS

fluid simulation, procedural texturing, ray tracing

ACM Reference Format:
Sean C McDuffee and Maurice van Swaijj. 2018. Procedural Fluid Textures. In *Proceedings of SIGGRAPH '18 Talks*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3214745.3214767

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
SIGGRAPH '18 Talks, August 12-16, 2018, Vancouver, BC, Canada
© 2018 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-5820-0/18/08.
https://doi.org/10.1145/3214745.3214767

1 INTRODUCTION

Texturing of dynamically changing surfaces, like fluids, offers a unique set of challenges to the computer graphics artist. Previously published results in this area primarily focused on the synthesis of maps. The simplest approaches advected texture coordinates through the fluid field [Witting 1999] and grew more complicated from there [Stam 1999] [Neyret 2003] [Rasmussen et al. 2003]. These approaches all have problems with ghosting and/or smearing of the texture. Other approaches used example based synthesis, employing an optimization method in addition to texture advection [Bargteil et al. 2006] [Kwatra et al. 2007]. While these methods fix ghosting and diffusion artifacts, the final textures did not follow the overall fluid motion the way artists may expect.

In addition to the visual issues with current approaches, the use of texture maps can potentially constrain memory resources, particularly in ray tracing systems. Procedural solid texturing [Ebert et al. 2002], where the material properties of a point are computed on the fly from a 3D coordinate, instead requires very little storage and real world fluid surfaces are often well represented by the types of materials procedural textures generate. While [Stam 1999] mentioned using solid texturing with fluid simulation as future work, we are unaware of others employing this approach.

2 OUR APPROACH

Artists today primarily utilize FLIP solvers for fluid simulation and our texturing solution is based on that. Our approach is most similar to Yu et al [Yu et al. 2009] where they attach texture patches to particles in their fluid sim on the surface and rendering is done by blending the patches together. We, however, attach 3D reference space lookup positions to the particles in the FLIP fluid volume and then project the reference field onto spheres on the free surface for rendering.

To do this, we start by adding a reference space position to each FLIP particle in the sim. Artists are free to choose any reference field they like, such as the initial world space position of the particle. As FLIP particles are introduced into the sim, the reference points on the new particles are interpolated from their neighbors. When the final render surface is generated from the fluid, the reference position field is then projected onto the vertices of the mesh. We use Houdini as our solver and all these operations are available out of the box. We add the reference world positions as particle attributes to the FLIP particles. We set the FLIP solver to interpolate the attribute to any new sim particles from the surrounding ones. Then we use an attribute transfer when generating the surface mesh to map the reference positions from the FLIP particles to the vertices of the mesh.

Each vertex of the render mesh is also assigned a radius of influence for the reference data. This radius can be set by the artist to anything but we found that it's best to use the smallest radius at each vertex such that the entire surface is covered as a whole. A simple heuristic, for example, is to use half the longest edge adjacent to a vertex. This minimal covering keeps the rendering efficient since fewer procedures need to be calculated, and results in the smoothest motion of the texture.

At render time, a ray intersection on the surface is checked against the nearby reference points whose radii overlap the intersection point. This is handled efficiently by grouping the reference spheres into a bounding volume hierarchy. The final reference point of a hit is computed using radial basis function interpolation of the overlapping spheres. This reference location is then used in a solid texture supplied by an artist to compute the final material of the surface. Functions which deform the surface can easily be applied as well.

3 CONCLUSIONS

We found this technique gave us exactly the look our artists were looking for and was simple to set up using existing systems. One issue we found was that a uniform covering of the surface gave the best results so adaptive meshing of the surface posed an issue sometimes. Higher resolution meshes and hence higher resolution sampling of the reference field also gave the best results. Future improvements could include the advection of orientations [Kwatra et al. 2007] along with the position data so a full reference transform could be used for lookup in anisotropic procedures.

REFERENCES

Adam W. Bargteil, Funshing Sin, Jonathan E. Michaels, Tolga G. Goktekin, and James F. O'Brien. 2006. A texture synthesis method for liquid animations. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation (SCA '06)*. Eurographics Association, Aire-la-Ville Switzerland, 345–351. https://dl.acm.org/citation.cfm?id=1218111

David S. Ebert, F. Kenton Musgrave, Darwyn Peachey, Ken Perlin, and Steven Worley. 2002. *Texturing and Modeling: A Procedural Approach (3rd. ed.)*. Morgan Kaufmann Publishers Inc.

V. Kwatra, D. Adalsteinsson, T. Kim, N. Kwatra, M. Carlson, and M. C. Lin. 2007. Texturing fluids. *IEEE Transactions on Visualization and Computer Graphics* 13, 5 (Sept. 2007), 939–952.

Fabrice Neyret. 2003. Advected textures. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation (SCA '03)*. Eurographics Association, Aire-la-Ville Switzerland, 147–153. https://dl.acm.org/citation.cfm?id=846297

Nick Rasmussen, Duc Quang Nguyen, Willi Geiger, and Ronald Fedkiw. 2003. Smoke simulation for large scale phenomena. In *ACM SIGGRAPH 2003 Papers (SIGGRAPH '03)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 703–707.

Jos Stam. 1999. Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques (SIGGRAPH '99)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 121–128. https://doi.org/10.1145/311535.311548

Patrick Witting. 1999. Computational fluid dynamics in a traditional animation environment. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques (SIGGRAPH '99)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 129–136. https://doi.org/10.1145/311535.311549

Qizhi Yu, Fabrice Neyret, Eric Bruneton, and Nicolas Holzschuch. 2009. Scalable Real-Time Animation of Rivers. *Computer Graphics Forum* 28, 2 (2009), 239–248.