Painting with Turbulence

Chuqiao Wang*
Clemson University

Sam Wang†
Clemson University

Robert Geist‡
Clemson University



Figure 1: *A Simulated Painting with Turbulence.*

Abstract

In a 2012 article in *Leonardo*, Lior Shamir presented an image analysis of 513 works of art created by 9 well-known artists. The analysis used the *Wndchrm* algorithm, which is based on a large collection of numerical image content descriptors. The conclusion, surprising at first glance, was that the two artists whose work was most similar to one another were Vincent van Gogh, a post-Impressionist, and Jackson Pollock, an Abstract Expressionist. Nevertheless, in retrospect, the essence of the strong similarity can be captured in a single word, employed by Leonardo da Vinci some 500 years ago: *turbulence*. The goal of this project is to create a software system that provides an artist-directable framework for composing paintings that naturally carry the energy of turbulent flow. To this end, a 3D lattice-Boltzmann model is used to simulate flows with Reynolds numbers on the order of 10 million, e.g., air flow at 100 m/s, and this is coupled with a particle system generated by a commercial package, *Houdini*, to provide a 3D turbulent flow of paint droplets. At the user's option, the 3D flow can be captured by an arbitrarily placed, simulated canvas.

1 Overview

Automated analysis of artwork, in particular, of paintings, is gaining increased attention due to its dual promise of objective characterization of artistic technique and authentication of works of questionable provenance. One such recent study was conducted by [Shamir 2012], who considered 513 paintings by artists Van Gogh, Monet, Pollock, Kandinsky, Rothko, Dali, Ernst, and de Chirico. He applied the so-called *Wndchrm* algorithm, which is based on a collection of numerical image content descriptors. The conclusion, which was sufficiently surprising to warrant inclusion as the article's title, was that the two artists with most similar styles were Vincent van Gogh and Jackson Pollock.

It is our thesis that the similarities captured by Shamir's application of *Wndchrm* can in fact be captured by a single word, *turbulence*. This term, employed by Leonard da Vinci (*la turbulenza*) in the 16^{th} century [Davidson et al. 2011], applies to both the artistic styles and the lifestyles of these two great artists. Thus, in building a computational system capable of generating new art that might carry some of the same energy and vitality as that of these artists, the essential feature to include is turbulent flow.

To this end we employ a two-phase process. In the first phase we solve the 3D Navier-Stokes equation,

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

where $\mathbf{u}$ denotes the velocity field, p is pressure, and ν is viscosity, for air flow at 100 m/s over simulated geometry, a sphere of diameter 1m. The high speed, combined with the low viscosity of air and the indicated geometry, yields a Reynolds number slightly larger than 10 million, a level which guarantees turbulent flow, but also one at which numerical solvers can easily become unstable. We use a variation on the lattice-Boltzmann method of [Premnath and Banerjee 2011], in which the collision step preserves, to the extent possible, higher order central moments of the momentum distribution. The classical, single relaxation time (SRT) lattice-Boltzmann method can become unstable at Reynolds numbers as low as 500, and the extension to multiple relaxation times (MRT), in which ordinary moments of the momentum distribution are preserved, can fail to ensure Galilean invariance, i.e., independence of inertial frame of reference. See [Geier et al. 2006].

*email:chuqiaw@g.clemson.edu
†email:stwang@clemson.edu
‡email:geist@g.clemson.edu

The lattice-Boltzmann method is easily parallelized to execute across one or more GPUs. Although the 3D solver is not real-time, a typical grid with 26 million nodes can be updated at approximately 85 updates (time steps) per minute on a single NVIDIA Titan X GPU.

In the second phase, we import the time-dependent velocity field, $\mathbf{u}$, into *Houdini* [Side Effects Software 2015] as a point cloud with a customized Python Geometry Operator and apply a *Volume From Attrib* node to generate the velocity field as a volume. Particles are then generated from artist-controlled emitters within a Houdini *DOP* (dynamic operator) *Network*, which is an object that contains a particle dynamics simulator and provides simulation-wide controls. The particles are then advected by the velocity field using a *POP* (particle operator) *Advect by Volumes*. A single "brushstroke" is thus a particle stream that flows within the turbulent velocity field. The artist controls the position, orientation and size of each emitter, as well as the start time, end time, emission rate, particle lifespan, and life variance. A collision object, which acts as a virtual canvas, is included in the same simulation to catch the splashes of the brushstrokes. The artist controls the size, orientation, and position of the canvas. Particles that are in contact with the canvas are then selected with a *Group* node as a "brushstroke on canvas".

A set of emitters with different positions is used to generate a wedge of brushstrokes varying in size and shape. Complete brushstrokes are then subject to higher level artistic controls which include color, translation, rotation, and scale. As particles collide with the canvas, a count is stored in the alpha channel of each canvas pixel. The count is later used to create a bump map, which gives an illusion of paint thickness during rendering. A shader that takes a color map, a bump map, and an opacity map is created and attached to a rectangular surface with dimensions that match the virtual canvas. The surface, instead of the particles that form the brushstrokes, is then rendered as the brushstrokes and composited with a canvas as the final image.

An example is shown in Figure 1. Note that, due to the extensive collection of controls available to the artist, markedly different images may be generated from a single, captured flow field. The composition of the painting of Figure 1 is based on ideas from "all-over painting", which refers to "a canvas covered in paint from edge to edge and from corner to corner, in which each area of the composition is given equal attention and significance" [Museum of Modern Art 2015]. It also engages principles coherent to those of Pollock's action painting. The brushstrokes show a random, but controlled motion as seen in the accompanying animation.

2 Related Work

Closely related work is that of [Lee et al. 2006] and [Lee et al. 2007], wherein the authors explicitly simulate the work of Jackson Pollock. They solve a 1D Navier-Stokes equation that represents the axial flow of a paint jet stream, and they couple this with a second one-dimensional PDE solver that controls deviation of the stream from the negative z axis. They model the interaction of the paint with the canvas using implicit surfaces to define the regions covered. There is no attempt to represent paint thickness.

An effective and detailed model of paint, both oil and pastel, applied with a 3D brush appears in [Chu et al. 2010]. This includes both paint thickness and ridges or streaks created by brushstrokes.

More recently [Lu et al. 2013] describe *RealBrush*, a data-driven system, which imports exemplars of single strokes, smears, and smudges, all done in a medium and a style of interest to the artist. The system then synthesizes new strokes in the same style. This allows artists to create images with the expressive qualities of arbitrary media.

Our approach differs from these in that we make little attempt to capture the properties of the paint per se or its interaction with a brush or canvas or previously applied paint and instead endeavor to capture only the energy of the artistic force propelling it, i.e., *la turbulenza*. Nevertheless, adding paint and brush interaction models to our approach would likely offer significant improvements, and this remains a current direction for further investigation.

References

CHU, N., BAXTER, W., WEI, L.-Y., AND GOVINDARAJU, N. 2010. Detail-preserving paint modeling for 3d brushes. In *Procedings of the 8th international symposium on non-photorealistic animation and rendering (NPAR 2010)*, Association for Computing Machinery, Inc.

DAVIDSON, P., KANEDA, Y., MOFFATT, K., AND (EDS.), K. S. 2011. *A Voyage Through Turbulence*. Cambridge University Press, Cambridge CB2 8BS UK.

GEIER, M., GREINER, A., AND KORVINK, J. G. 2006. Cascaded digital lattice Boltzmann automata for high reynolds number flow. *Physical Review E 73*, 6, 066705.

LEE, S., OLSEN, S., AND GOOCH, B. 2006. Interactive 3d fluid jet painting. In *NPAR '06: Proc. 4th Int. Symp. on Non-photorealistic Animation and Rendering*, ACM Press, New York, NY, 97–104.

LEE, S., OLSEN, S., AND GOOCH, B. 2007. Simulating and analysing jackson pollock's paintings. *Journal of Mathematics and the Arts 1*, 2, 73–83.

LU, J., BARNES, C., DIVERDI, S., AND FINKELSTEIN, A. 2013. Realbrush: Painting with examples of physical media. *ACM Trans. Graph. 32*, 4 (July), 117:1–117:12.

MUSEUM OF MODERN ART, 2015. Allover painting. http://www.moma.org/cef/abex/html/know_more9.html.

PREMNATH, K., AND BANERJEE, S. 2011. On the three-dimensional central moment lattice boltzmann method. *J Stat. Phys. 143*, 747–794.

SHAMIR, L. 2012. Computer analysis reveals similarities between the artistic styles of van gogh and pollock. *LEONARDO 45*, 2, 149–154.

SIDE EFFECTS SOFTWARE, 2015. Houdini 14.0 documnetation. http://www.sidefx.com/docs/houdini14.0/.