EUROGRAPHICS 2020 / U. Assarsson and D. Panozzo
(Guest Editors)

Volume 39 (2020), Number 2

Simulation of Dendritic Painting

José A. Canabal[1] Miguel A. Otaduy[1] Byungmoon Kim[2] Jose Echevarria[2]

[1]Universidad Rey Juan Carlos, Madrid, Spain
[2]Adobe, San Jose, California

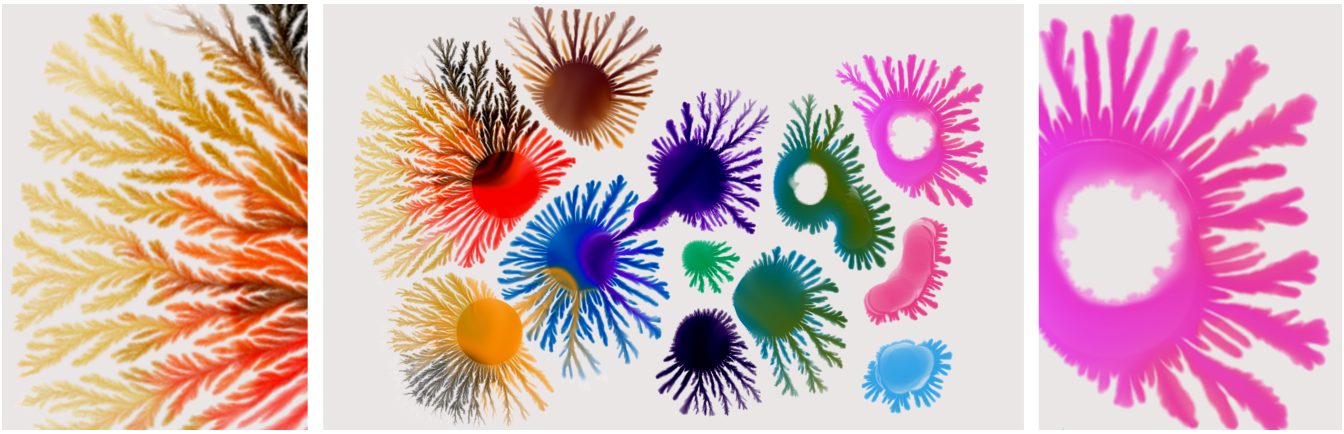

Figure 1: *Example of dendritic painting obtained with our system. Our proposed diffusion and reaction functions capture the complex boundary conditions of the multi-phase fluid interactions between the catalyst medium, the solvent and the ink. This demo runs at 12fps on a high-resolution grid of size 2000x2000.*

Abstract

We present a new system for interactive dendritic painting. Dendritic painting is characterized by the unique and intricate branching patterns that grow from the interaction of inks, solvents and medium. Painting sessions thus become very dynamic and experimental. To achieve a compelling simulation of this painting technique we introduce a new Reaction-Diffusion model with carefully designed terms to allow natural interactions in a painting context. We include additional user control not possible in the real world to guide and constrain the growth of the patterns in expressive ways. Our multi-field model is able to capture and simulate all these complex phenomena efficiently in real time, expanding the tools available to the digital artist, while producing compelling animations for motion graphics.

CCS Concepts

• ***Computing methodologies*** → ***Physical simulation;***

1. Introduction

Dendritic painting is a form of artistic expression that lets an artist produce rich, expressive, organic patterns without attention to detail, and focusing instead on the flow and color of the patterns [Boh11, Kow17] (Figure 3). Just a few drops of ink lead to rich and colorful patterns, but the artist is in turn hindered by limited controllability of the result. Digital painting simulation has emerged over the years as a solution to combine the virtues of the physical and digital worlds. Physical simulation of the painting media and techniques enables realistic reproduction of the physical creative processes, while automatic digitization of the painting result endows the artist with editing and control operations simply not possible in a physical medium. Digital painting simulation has been applied to diverse creative processes supporting highly viscous media such as acrylic or oil paint [YJC*13, CKIW15], as well as highly dispersive media such as watercolor or eastern inks [CAS*97, CT05, DKMI13]. Unfortunately, the simulation methods proposed to date for the various forms of digital painting fail to support the creative techniques of dendritic painting.

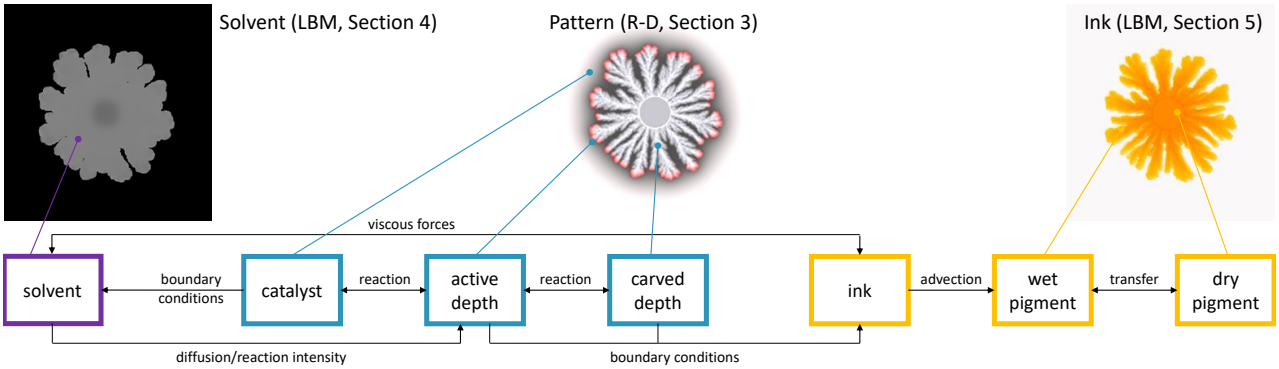

Figure 2: *Overview of our simulation model, with its three major layers, their components, and interactions. The solvent layer enables the growth of the pattern. The darker the grey, the more solvent there is (black represents absence of solvent). The pattern layer arises from the interplay between the base medium and the solvent (red represents the active depth ρ_a, the inner grey part the carved depth ρ_p, and the outer grey part the catalyst density ρ_y). The ink layer models the advection of the ink pigment over the pattern.*

In dendritic painting, the painting surface is covered first with a base medium, typically white acrylic paint diluted with water [Zam17]. During the creative process, the artist pours on the base medium a mix of acrylic ink and a solvent, typically alcohol or some acid. The complex multi-phase fluid dynamics between solvent, ink and base medium produce rich and expressive high-frequency branching effects on the ink as it flows and deposits. Simulating such complex branching phenomena in a physically based way would require a multi-phase fluid simulation of very high resolution with complex boundary conditions, not suitable for interactive digital painting.

Instead, in this paper, we propose a phenomenological simulation model for dendritic painting that models the branching phenomena explicitly using a pattern growth algorithm, and couples the pattern growth to solvent and ink fluid dynamics. We split the complexity of the full multi-phase phenomenon, while effectively capturing the complexity and richness of the resulting paint patterns. Our model is carefully designed for a painting context, where behaviors should be intuitively linked with properties and amounts of materials. As shown in Figure 1, our digital painting model is able to mimic complex real-world effects.

Our work entails the following major contributions:

- We introduce a Reaction-Diffusion (RD) model for dendritic patterns for digital painting. Thanks to a multi-field representation with carefully designed reaction and diffusion terms, our model achieves rich and controllable organic branching. This is in contrast to previous methods for the simulation of dendritic patterns in computer graphics, which have focused on more regular phenomena [RHLH18].
- We design a two-way coupling procedure between the pattern growth simulation and solvent and ink fluid dynamics, which effectively tackles the complex boundary conditions in the underlying multi-phase phenomenon.
- Overall, we provide the first simulation pipeline for dendritic painting. We complement the pattern growth and fluid dynamics simulations with pigment advection and various control and editing operations, to empower the artist with a digital tool to create rich and intricate artworks in a straightforward way.

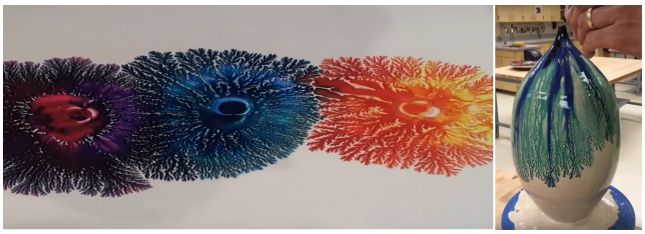

Figure 3: *Examples of dendritic painting. Left: Colorful patterns obtained with acrylic inks mixed with alcohol. Right: Similar results obtained with the* mocha diffusion *technique used in pottery.*

Our digital dendritic painting system performs the simulation of three different layers, depicted in Figure 2, which combine multiple simulation methodologies. The first layer simulates the growth of the pattern, with reaction-diffusion equations that govern the evolution of the catalyst material, as well as the active and carved pattern depths. This layer is described in Section 3. The second layer simulates the fluid dynamics of the solvent, using the Lattice-Boltzmann method (LBM), and is described in Section 4. And the third layer simulates the fluid dynamics and pigment mixing of the ink, using again LBM. This layer is described in Section 5.

2. Related Work

Digital painting and dendritic patterns are two classical areas that expand beyond the scope of this paper. In the following we discuss the references closer to our work.

Interactive watercolor. Due to its highly dispersive nature, watercolor may be considered the closest technique to dendritic painting. Previous watercolor simulations based on Navier-Stokes [CAS*97] [LVR05] and Lattice-Boltzmann [CT05] simulate the interaction of water, pigment and paper fibers to obtain natural flow effects and feathery patterns. However, dendritic patterns arise

from the interaction between different fluids alone (inks, solvent, medium), and previous models are not able to capture the multi-phase interactions required for the sharp characteristic branching patterns. On the contrary, we model explicitly the formation of dendritic patterns using a novel nonlinear reaction-diffusion model. Alternative sparser watercolor simulations like the one proposed by DiVerdi et al. [DKMI13] would require a completely new model and extremely high resolution to achieve patterns like ours.

Digital oil painting. Less close to dendritic painting, but relevant in computer graphics, are oil painting simulation techniques. Chen et al. [CKIW15] propose a hybrid fluid simulation model combining FLIP for the paint near the brush, and an Eulerian representation in the other areas. Stuyck et al. [SDHD16], on the other hand, propose an oil painting model for mobile devices based on the shallow water equations modified to suit this type of paint.

Dendritic pattern simulation. Dendritic patterns occur frequently in nature and so different models have been proposed and adapted for different phenomena. Kim et al. [KL03] proposed the Phase Field model [Kob93] for ice crystal growth over a user-controllable freezing map. Later on they proposed a more efficient hybrid method [KHL04] combining Phase Field with Diffusion Limited Aggregation (DLA) [WS83], with improved user control. More recently, Ren et al. [RHLH18] extended such Phase Field formulation to enable crystal growth following arbitrary orientation fields in 2D and 3D. We experimented with phase fields to model our patterns, but found that they tend to produce patterns that are too regular for our use case. Our proposed reaction diffusion equations generate more organic patterns instead.

Similar patterns can be obtained by a liquid slowly flowing between two parallel plates separated by a gap, forming so called *viscous fingers*. Bogoyavlenskiy [Bog01] proposed modeling them using DLA. Alternatively, Segall et al. [SVBC16] identified the phenomenon as a *Hele-Shaw cell*, and proposed an efficient solution to its governing fluid equations based on the use of complex holomorphic barycentric coordinates. Unfortunately, the physical phenomena behind dendritic painting and viscous fingers are different, and the governing fluid dynamics equations do not match; therefore, it is challenging to repurpose their models for interactive digital painting. Viscous fingering has also been used to simulate miscible mixing [SKK10], but their focus is on solids and liquids dissolving or changing to other substances rather than dendritic patterns.

In the case of morphogenesis, growth of living organisms has been simulated with Reaction-Diffusion equations [Tur52]. In computer graphics, Turk [Tur91] and Witkin and Kass [WK91] demonstrated plenty of varied patterns found in animals and nature. Wan et al. [WLWL10] used RD to generate maze-like patterns from images. Golding et al. [GKCBJ98] made an interesting analysis of the mechanisms that allow growth of bacterial colonies using RD models. Using the Fisher-Kolmogorov equation [NKGPSP37] for biological growth, different authors have explored the dynamics of bacterial colonies in order to replicate the organic shapes they produce. The basic idea under these models [Kit97, KMM*97] is to replicate the motility, reproduction and death of the bacteria by an RD model of three different fields: living bacteria, nutrients and dead bacteria. We have identified similarities in the morphology of bacterial growth patterns and dendritic patterns, as well as parallelisms between the parameters governing growth behavior. Therefore, we have used the RD models of bacterial growth as base formulation for our model, and we have adapted them to accommodate the processes and boundary conditions present in dendritic painting. RD models were used long ago in computer graphics to produce digital painting effects [Lew84], but not with the morphology of our patterns or coupled to fluid dynamics.

Figure 4: *Influence of nonlinear diffusion (2) on pattern branching. From left to right, $k = 1$ (linear diffusion, which lacks branching), $k = 2$ (our choice in the examples), and $k = 2.3$. Differences in size come from the pattern growth speed (simulation time was 13, 40 and 55 seconds for these examples). Please refer to the supplementary video for an animated comparison.*

3. Dendritic Pattern Growth

As outlined in the introduction, we have designed an RD model for the simulation of the growth of dendritic patterns. Instead of a fully physics-based model, we devise a phenomenological model that maps the main aspects of dendritic painting into controllable components of the model. Our proposed RD model contains three scalar fields, with diffusion and reaction functions that capture the complex boundary conditions of the multi-phase fluid interaction, and thus produce a pattern with organic branching. We start this section with the high-level mathematical description of the model following RD equations, and then itemize the reaction and diffusion terms for the various spatiotemporal fields that represent the pattern. We motivate the design of such terms in the context of previous literature on RD models for pattern growth, and we discuss the modifications that induce the complex branching effects in dendritic painting. We conclude the section with details on discretization and efficient handling of isotropic growth.

3.1. Reaction-Diffusion Model

In the pattern layer, we model the spatiotemporal evolution of three scalar fields: the density of catalyst material (ρ_y), the active pattern depth (ρ_a), and the carved pattern depth (ρ_p). As noted in our discussion of related work, this model is inspired by works on the numerical simulation of bacterial growth [GKCBJ98]. The catalyst represents the amount of exposed base medium, which enables pattern growth. As the pattern carves through the surface of the base medium, the catalyst is consumed and the pattern stops growing. The active pattern defines the growth of the pattern on its boundary, and its evolution depends on the catalyst and the solvent. Finally, the carved pattern defines the region where ink can flow. The active pattern transforms into carved pattern, thus producing the effective growth of the ink pattern.

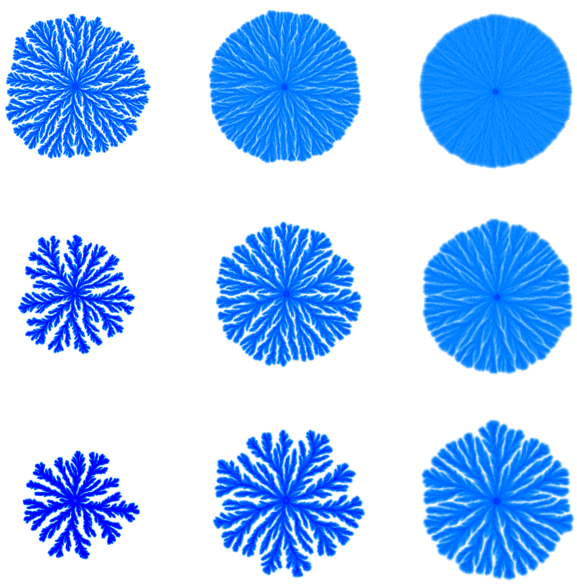

Figure 5: *Effects of α_a and α_y on pattern branching and growth. From left to right $\alpha_a = 0.1, 0.3, 0.75$. From top to bottom $\alpha_y = 0.3, 0.6, 1.0$. Differences in size come from the pattern growth speed (simulation time was 50 seconds for these examples). Please refer to the supplementary video for an animated comparison.*

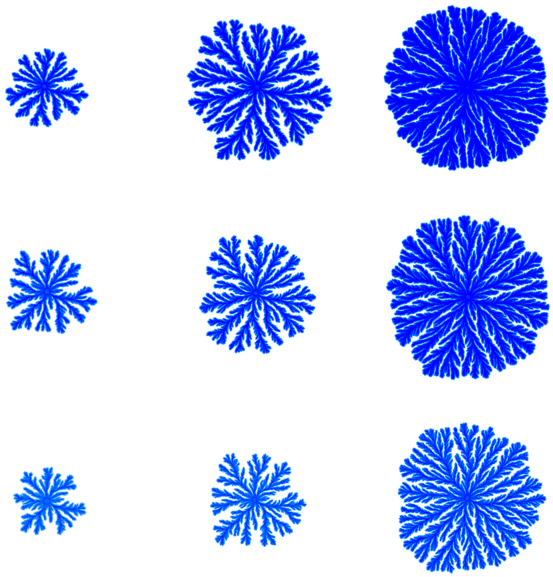

Figure 6: *Effects of β_a and β_y on pattern branching and growth. From left to right $\beta_a = 0.75, 1.0, 1.5$. From top to bottom $\beta_y = 0.75, 1.0, 1.5$. The examples with $\beta_a = 0.75$ took 10 times longer than the examples with $\beta_a = 1.5$ to reach the extent shown. Differences in size come from the pattern growth speed (simulation time was 50 seconds for these examples). Please refer to the supplementary video for an animated comparison.*

The evolution of each of the three scalar fields can be described using a generic RD equation:

$$\frac{\partial \rho_i}{\partial t} = D_i + R_i, \tag{1}$$

where D_i represents a generic diffusion function, dependent on the Laplacian of the scalar field, $\nabla^2 \rho_i$, and R_i represents a generic reaction function. $i \in \{y, a, p\}$ represents, respectively, the catalyst, the active pattern, and the carved pattern. The diffusion and reaction functions for the three scalar fields depend in complex ways on each other, as well as the density of solvent, ρ_s, simulated in the second layer. We achieve the complex branching effects in dendritic painting partly thanks to the versatility produced by the interaction of three scalar fields, and partly thanks to a careful choice of the reaction and diffusion functions, which we describe next.

3.2. Active Pattern

Most of the complexity of our pattern growth model lies in the RD functions of the active pattern. In addition to the catalyst and the already carved pattern depth, the RD functions of the active pattern depend on the density of solvent, ρ_s. In this way, the RD functions capture the complex effects occurring at the boundary of the solvent. We describe the simulation of the solvent layer later in Section 4, but we anticipate that the solvent is also influenced by the pattern to account for two-way coupling.

Kozlovsky et al. [KCGBJ99] required an RD model with four coupled fields to achieve fine organic branching patterns. In their model, the subtle differences between two of the fields produce the instabilities that lead to branching. Alternatively, Kawasaki et al. [KMM*97] found that these two fields can be merged into a single field with nonlinear diffusion, which produces comparable instabilities and branching, but with reduced computational complexity. Our RD model is inspired by the one of Kawasaki et al., but we modify the various reaction and diffusion terms to account for the coupling with the solvent's fluid dynamics, and to support other boundary conditions. Following Kawasaki et al., we propose a nonlinear diffusion function for the active pattern. In our case, it takes the form

$$D_a = H(\rho_s - \delta)\, \xi\, \alpha_a\, \nabla^2 \rho_a^k. \tag{2}$$

where the degree of nonlinearity k enhances instability at the front of the pattern, leading to the branching effect. We use $k = 2$ in all our examples, since we found it produces patterns closer to the real ones in our experiments. Figure 4 evaluates its influence.

The diffusion coefficient α_a affects strongly the thickness and sharpness of pattern branches, with smaller values leading to thinner and sharper branches (Figure 5). $H(\cdot)$ is a Heaviside function that models solvent boundary effects. Diffusion takes place only when the solvent density is above a threshold δ (0.1 in our examples). Finally, ξ is a noise function that provides random isotropic growth and guides the creation of sub-branches, and is discussed in more detail in Section 3.4.

Figure 7: *Exploring the effect of the diffusion noise function. From left to right: (i) $\xi_0 = 0.01$; (ii) $\xi_0 = 0.2$; (iii) $\xi_0 = 0.8$. Please refer to the supplementary video for an animated comparison.*

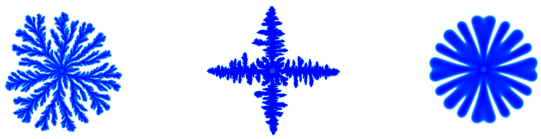

Figure 8: *Left: Random isotropic growth of the pattern, thanks to our diffusion noise function (6) and a 9-point Laplacian stencil. Middle: With noise function but a 5-point Laplacian stencil. Right: With the 9-point Laplacian stencil but no diffusion noise ($\xi = 1$). Please refer to the supplementary video for an animated comparison.*

We propose a reaction function for the active pattern of the form

$$R_a = \beta_a \rho_s \rho_y \rho_a (1 - \rho_a) - \beta_p \rho_a. \tag{3}$$

This function combines two effects: i) A pattern depth increase resulting from the combination of catalyst and solvent, with coefficient β_a. This effect grows progressively as the active depth grows, and then vanishes as the active depth saturates. ii) A transformation of active depth into carved pattern depth, modulated by a coefficient β_p. We use a constant value of $\beta_p = 0.15$ in all our examples. β_a, on the other hand, affects strongly the speed at which the pattern grows (Figure 6).

3.3. Carved Pattern and Catalyst

The carved pattern evolves due only to the transformation of active depth, as already accounted for in (3), with no diffusion. The RD functions of the carved pattern amount then to

$$D_p = 0, \qquad R_p = \beta_p \rho_a. \tag{4}$$

As mentioned above, the catalyst represents the local amount of base medium that is exposed to pattern growth. Then, the catalyst is initialized to a maximum value at places with base medium, and its density is reduced as the pattern depth grows. In particular, we model the evolution of the catalyst using a linear diffusion function and a reaction function that is bilinear w.r.t. the catalyst density and the active pattern depth.

$$D_y = \alpha_y \nabla^2 \rho_y, \qquad R_y = -\beta_y \rho_a \rho_y, \tag{5}$$

where α_y and β_y are, respectively, diffusion and reaction coefficients. Larger values of α_y produce thicker, less detailed branches (Figure 5), while smaller values of β_y also produce thicker branches, but preserve most of the detail (Figure 6).

3.4. Discretization and Random Isotropic Growth

Dendritic paint patterns grow following random isotropic branches. However, a standard discretization of our RD model on a regular grid suffers anisotropy artifacts, with preferential growth along the axes of the grid. Moukarzel [Mou92] proposed a discretization based on random lattices to model the growth of isotropic patterns, and thus avoided the artifacts of regular grids. Nevertheless, regular grids are beneficial for efficient massively parallel implementation.

We achieve random isotropic growth while retaining a regular-grid discretization, thanks to the inclusion of the noise function ξ in the diffusion of the active pattern (2). Specifically, the noise function is defined as

$$\xi = 1 + \xi_0 \, \mathrm{rand}(\mathbf{x}), \tag{6}$$

where $\mathrm{rand}(\mathbf{x})$ is a function that generates a random number in the interval $[-1, 1]$. The noise amplitude ξ_0 affects the probability of growing sub-branches, as shown in Figure 7.

The solution to the RD equations requires the discretization of the Laplacian. To this end, we use a 9-point stencil based on the weights of the D2Q9 discretization used in Lattice Boltzmann simulations. This discretization method avoids grid artifacts of the standard 5-point Laplacian [TAAS13]. Figure 8 demonstrates the combined effect of the diffusion noise function (6) and the D2Q9 discretization of the Laplacian.

4. Solvent Simulation

The complex interaction of solvent, ink, and base medium occurring at the interface of the solvent is handled through the pattern layer described in the previous section. Then, we model the solvent as a 2D fluid, with the pattern defining its effective simulation domain. We have adopted the Lattice Boltzmann method (LBM) to compute the fluid dynamics of the solvent, inspired partly by the successful application of LBM to the 2D simulation of ink dispersion [CT05, EWK*13]. SPH or MPM are alternative choices for the simulation of the solvent, but we did not explore them due to the success with LBM. No matter the method of choice, the complexity lies in the design of the boundary conditions.

We begin this section with a summary of LBM and our choice of discretization. Then, we describe the boundary conditions that account for the boundary of the pattern, and we conclude with the interaction between solvent and ink in the interior of the pattern.

4.1. Lattice-Boltzmann Model

LBM has been studied thoroughly in computer graphics. It enjoys important features that enable an efficient massively parallel implementation, and hence interactive digital painting at high resolutions: it handles incompressibility through efficient local operations (albeit at the price of memory, which is however a minor issue

in 2D), and it can resolve the boundary conditions of free-surface fluids with a single-phase simulation [Thü03]. Recent advances include efficient simulations on adaptive grids [TR09], or two-phase methods for highly detailed effects [GLX17].

LBM solves fluid dynamics by modeling particle operations on a lattice. It stores distribution functions of particles according to lattice-aligned velocities, and performs streaming and collision operations on these distribution functions to model, respectively, advection and incompressibility. The density and velocity of the fluid can be recovered at any time from the particle distributions.

Our LBM model of the solvent follows overall the formulation designed by Chu and Tai [CT05] for watercolor. However, it should be possible to use more modern LBM models, such as multiple-relaxation-time [LL00]. Our model adopts the D2Q9 lattice discretization, which is characterized by 8 vectors $\mathbf{e}_i$ ($i \in \{1 \dots 8\}$) connecting a lattice point to its 8 neighbors, and 9 particle distribution functions f_i ($i \in \{0 \dots 8\}$). f_0 represents the distribution function of particles at rest, while f_i ($i \in \{1 \dots 8\}$) represent the distribution functions of velocities along $\mathbf{e}_i$.

The solvent density ρ_s and velocity $\mathbf{u}$ can be reconstructed from the distribution functions as

$$\rho_s = \sum_{i=0}^{8} f_i, \qquad \mathbf{u} = \frac{1}{\rho_s} \sum_{i=1}^{8} f_i \mathbf{e}_i. \tag{7}$$

The streaming step computes tentative distribution functions due to advection:

$$f_i'(\mathbf{x}, t+\Delta t) = f_i(\mathbf{x} - \mathbf{e}_i, t). \tag{8}$$

In practice, the streaming step amounts to copying the values of the distribution functions to adjacent lattice points along the vector directions. Note that f_0 is not streamed.

The collision step requires the definition of equilibrium distribution functions:

$$\tilde{f}_i = w_i \left(\rho_s + \psi \bar{\rho}_s \left[\frac{3}{c^2} \mathbf{e}_i^T \mathbf{u} + \frac{9}{2c^4} \left(\mathbf{e}_i^T \mathbf{u} \right)^2 - \frac{3}{2c^2} \mathbf{u}^T \mathbf{u} \right] \right). \tag{9}$$

$c = \frac{\Delta x}{\Delta t}$, and in our examples we set $\Delta x = \Delta t = c = 1$ for simplicity. $\bar{\rho}_s$ is the average solvent density, which is set to 1 in our examples. w_i are constant weights, with $w_0 = 4/9$, $w_i = 1/9$ for directions aligned with lattice axes, and $w_i = 1/36$ for diagonal directions. ψ is a coefficient borrowed from the work of Chu and Tai [CT05] to account for boundary conditions in the advection step, and is described in detail in the next subsection.

The collision step interpolates between the tentative and equilibrium distribution functions from (8) and (9):

$$f_i(\mathbf{x}, t+\Delta t) = (1-\omega) f_i'(\mathbf{x}, t+\Delta t) + \omega \tilde{f}_i, \tag{10}$$

with ω a relaxation parameter, in the interval $0 \leq \omega \leq 2$ for stability ($\omega = 0.5$ in our examples).

4.2. Boundary Conditions

As noted earlier, the active pattern defines the boundary of the simulation domain for the solvent, which effectively translates into the simulation of the solvent as a free-surface fluid. Fortunately, as demonstrated by Chu and Tai [CT05], this is possible using a single-phase simulation, through small modifications to the regular LBM equations. We adapt their solution to our case, where the solvent domain is bounded by the active pattern.

We perform two modifications over the regular LBM equations. First, the computation of equilibrium distribution functions (9) with $\psi = 1$ could cause negative solvent density values at the boundary. Instead, we modulate the advection with the function ψ to ensure it acts only when the solvent density is large enough (i.e., sufficiently away from the boundary). In practice, we define ψ as a smooth step from 0 to 1 in the density range $\rho_s \in [0, \mu]$. We implement the smooth step using Hermite interpolation, and we set the threshold density $\mu = 0.5$ in all our experiments.

The second modification to the regular LBM equations accounts for the extent of the pattern to define boundary conditions on the streaming step. Specifically, we apply a *half-way-bounce-back* scheme [SY02] at the pattern boundaries. To identify the pattern boundaries, we compute a blocking factor $\kappa = \rho_y / \bar{\rho}_y$, where $\bar{\rho}_y$ is the initial catalyst density. Lattice points with $\kappa > 0$ indicate that the catalyst has not been fully consumed, hence they are treated as part of the pattern boundary. Given a boundary point, we compute a directional blocking factor κ_i for each direction $\mathbf{e}_i$ by averaging the blocking factor with the adjacent lattice point. Based on this directional blocking factor, we redefine the streaming step (8) as

$$f_i'(\mathbf{x}, t+\Delta t) = (1-\kappa_i) f_i(\mathbf{x} - \mathbf{e}_i, t) + \kappa_i f_j(\mathbf{x}, t), \tag{11}$$

where f_j is the distribution function in the direction opposite to $\mathbf{e}_i$. If no catalyst is consumed yet (i.e., $\kappa_i = 1$), the distribution function is bounced, and the solvent is effectively stopped at the boundary.

4.3. Coupling of Solvent and Ink

In the boundary of the pattern, the interaction between the various fluid phases is handled by our RD pattern simulation. In the interior of the pattern, however, the solvent and the ink mix in a smooth way. We solve this mixing by introducing viscous forces between the solvent and ink layers.

After streaming (11), and before computing the equilibrium distribution functions (9), we apply viscous forces to the solvent velocities. Given ink velocity $\mathbf{u}_k$ and a viscosity factor γ (0.1 in our examples), we recompute the solvent velocity as:

$$\mathbf{u} \Leftarrow \kappa \mathbf{u} + (1-\kappa) \left(\gamma \mathbf{u}_k + (1-\gamma) \mathbf{u} \right). \tag{12}$$

Recall that the blocking factor κ identifies pattern boundaries, hence viscosity is not applied at boundaries.

5. Ink Simulation

In this section we describe the last layer of our dendritic painting simulation, the ink. We account for three major phenomena: the dynamics of the ink flowing over the pattern, the mixing of pigments as they flow, and evaporation leading to dry ink.

5.1. Ink Fluid Dynamics

In our real-world experiments, we have observed that the solvent adopts a coarser pattern at its interface with the base medium, while

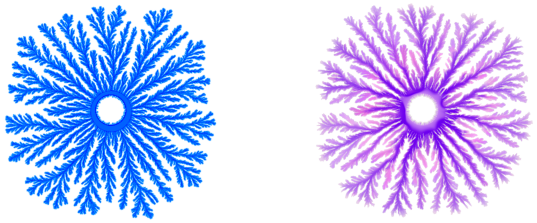

Figure 9: *Left: Pattern rendered based on a solid color modulated by ρ_p. Right: Colorful pattern rendered through the proposed ink advection scheme.*

the ink fits a finer and richer pattern within the domain occupied by the solvent. We model this effect by letting the solvent carve the pattern, regardless of the presence of ink or not, and then simulating the ink as a fluid that flows over the pattern. As described in Sections 3 and 4, we simulate the pattern and solvent layers in a coupled manner, with the catalyst density defining the boundary conditions for the solvent. To define the boundary conditions for the ink, we use instead the active and carved pattern depths.

Following the same approach as for the solvent, we simulate the ink using LBM. The algorithmic details are analogous, with the only exception of the computation of blocking factors. For the ink, we define the blocking factor κ as

$$\kappa = \max\left(1-(\rho_a+\rho_p),0\right), \tag{13}$$

such that $\kappa > 0$ indicates lattice points where the pattern has not reached a minimum depth, hence they are treated as part of the pattern boundary.

In Section 4.3 we have described viscous forces that couple the solvent and the ink in the interior of the pattern. In the ink simulation, we apply the same forces to the ink, with opposite sign, to account for action-reaction.

5.2. Pigment Mixing

We model the pigment, i.e., the color, as a passive medium advected by the ink fluid. Specifically, we represent the pigment field using a 3D vector field $\mathbf{p}$, corresponding to the CMY color space. We advect the pigment field using a semi-Lagrangian advection scheme [Sta99]:

$$\mathbf{p}(\mathbf{x},t+\Delta t) = \mathbf{p}(\mathbf{x}-\Delta t\,\mathbf{u}_k,t), \tag{14}$$

where $\mathbf{u}_k$ is the ink velocity. In practice, the semi-Lagrangian advection scheme may attempt to fetch color values from lattice points beyond the pattern's boundary. In such cases, we simply cancel the color update (14).

This approach also supports the advection of parameters for more accurate color mixing models like Kubelka-Munk [HM92], but a simple CMY interpolation was enough to produce colorful patterns in our case (Figure 9).

Figure 10: *The user can combine the dendritic patterns with traditional drawings. This demo runs at 44fps with a grid size of 1500x1500.*

5.3. Evaporation and Dry Ink

During a painting session, ink gets dry over time, and it stops flowing along the pattern. To model this effect, we add a secondary pigment field $\mathbf{p}_d$, which represents dry pigment. On every simulation step, we convert wet pigment into dry pigment following a linear relationship with drying coefficient β_d:

$$\frac{\partial \mathbf{p}}{\partial t} = -\beta_d\,\mathbf{p}, \qquad \frac{\partial \mathbf{p}_d}{\partial t} = \beta_d\,\mathbf{p}. \tag{15}$$

The drying coefficient can be set at will by the artist to stop or accelerate evaporation.

In the real world, the addition of solvent softens previously dried ink. We model this effect by converting dry pigment into wet pigment.

6. Results

6.1. Implementation Details

The described methods were implemented in C++ and OpenGL 4.5 and videos were recorded running on a Nvidia GTX 970. All the simulation pipeline runs on GPU using Compute Shaders, making the code suitable for a large range of hardware. We use a regular grid with the same resolution for all the layers of our system.

To store the data we use OpenGL textures. We use one RGB texture for the reaction-diffusion model, storing the scalar fields on each of its components. For the solvent we need four RGBA textures. Three of them for the nine distribution functions, density and velocity fields. The blocking factor and mask are stored in a the fourth one. The ink layer uses the same number of textures for the Lattice Boltzmann Method, but in this case, two additional RGBA textures are needed to store the color information, one for the flowing ink and another one for the pigment that has been fixed.

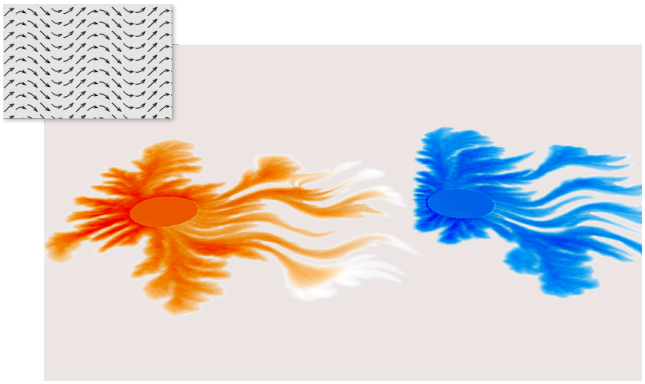

Figure 11: *Our model can also be guided by arbitrary vector fields. In this example we used a sinusoidal field (as shown on the inset). The supplementary video contains the whole animation.*

We used the following parameter values in our examples. Pattern: $k = 2$; $\alpha_a \in [0.1, 0.4]$; $\alpha_y \in [0.3, 0.8]$; $\beta_a = 3.0$; $\beta_y = 2.0$; $\xi_0 = 0.8$. Solvent: $\omega = 0.5$; $\mu = 0.5$; $\gamma = 0.1$. Ink: $\beta_d = 0.01$. In the example in Figure 1, the parameters vary smoothly across the domain according to a Perlin noise function.

6.2. Performance

Dendritic patterns can grow at different speeds depending on the mixtures used for the catalyst and inks. As seen in Section 3 and the supplementary videos, the same happens in our simulation depending on the choice of parameters. Independent from the parameters, our simulation runs at a frame rate that depends only on the grid size. In a similar way as other fluid simulations, changing the resolution affects the simulation results. The size and performance of the various examples is: teaser (Figure 1), 2000×2000 at 12 fps; butterfly (Figure 12) and girl (Figure 10), 1500×1500 at 44 fps; all other demos, 720×720 at 120 fps.

For typical parameters and the 720×720 grid, interactions can happen at a speed comparable to the real phenomena. We use an explicit integration scheme for all the examples, and this causes small time steps specifically in the reaction diffusion model. The size of the time step does not affect the results of our simulations, as long as the time step if small enough to satisfy stability. Due to the time-step limitation, we perform 8 steps per frame for the reaction-diffusion part, which affects performance significantly. The supplementary videos have been sped up by the corresponding factor indicated on them.

Average timing distribution for the simulations on the 720×720 grid is: 0.12 ms for the solvent simulation; 7.3 ms for the pattern simulation in total; and 3.3 ms for the ink model. Due to substepping, the reaction-diffusion part is the bottleneck in our implementation. It might be possible to extend the method to adaptive grids [TR09], and thus increase performance. Moreover, in our implementation the reaction-diffusion model is computed on all cells, and it should be possible to restrict it to active cells.

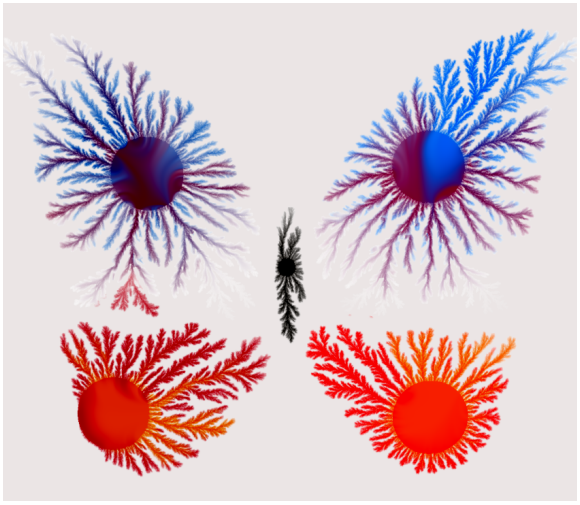

Figure 12: *The user can provide a binary mask, and we add this mask as an obstacle to the simulation, confining the growth of the pattern without unnatural stopping of the branches. This demo runs at 44fps with a grid size of 1500x1500. Please see the supplementary video for the full animation.*

6.3. Artist Control

Due to its nature, dendritic painting is hard to control. We provide different tools that allow the user to have more control over the final result of the painting.

6.3.1. Guidance Field

Apart from the radial patterns from the real world, our model supports being guided by arbitrary velocity fields, opening the path for custom behaviors that can be useful for motion graphics or more custom patterns. The user can thus provide a guidance field $\mathbf{v}$ created procedurally or extracted from an image (Figure 11).

To enforce the guidance field, we modify the RD model (1) of the active pattern depth to add another reaction term R'_a to (3).

$$R'_a = \beta_v \rho_a \mathbf{v}^T \nabla \rho_a. \tag{16}$$

This reaction is stronger when the gradient of the active pattern depth is aligned with the guidance field $\mathbf{v}$, inducing faster growth. β_v controls the global scale of the guidance (1.9 in the example).

6.3.2. Boundary Shapes

It is also interesting to constrain the growth of the patterns within a specific region of the canvas. Figure 12 shows an example where growth has been constrained to the inside of a logo. To achieve the effect with the branches progressively stopping without artifacts, we introduce obstacles as Neumann boundary conditions [Bri08] in the simulation of the active pattern and the catalyst. The density fields $\rho_i, i \in \{a, y\}$ satisfy $\mathbf{n}^T \nabla \rho_i = 0$, where $\mathbf{n}$ is the normal direction of the obstacle.

6.3.3. Other Painting Results

We have also experimented with the features of our painting system to produce effects possible in the real world, but difficult to

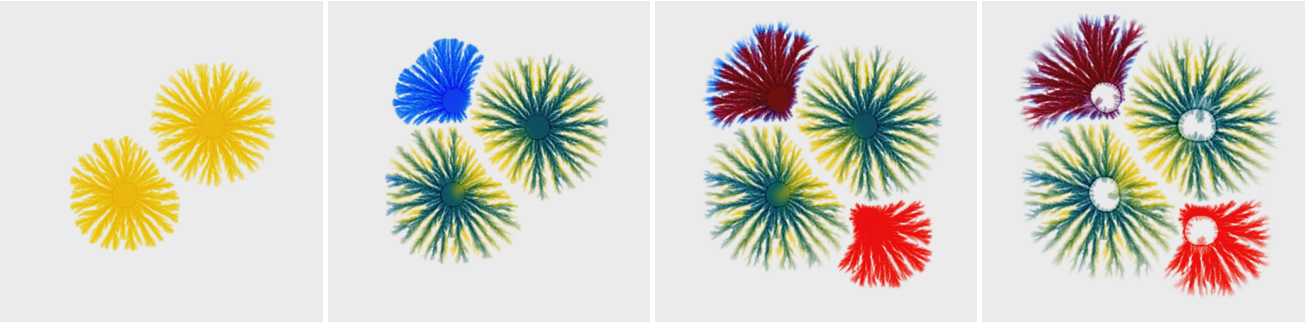

Figure 13: *Frames from another example of a painting session with our system (animated in the supplementary video).*

control. Figure 14 shows an example where the artist pours catalyst medium on a grown pattern, to produce an inward growing effect on the ink pattern. We show that our method produces results that match qualitatively those in the real world. Currently, the initial drop has a strong effect in our simulation due to our choice for the initialization of the various layers when the drop is added. We use a hard-edge circle as an input shape, but this input shape could be blurred, and hence make the initial drop less dominant.

Figure 13 shows the evolution of a painting session, where the artist merges and superimposes patterns of different colors. Notice how new pigments flow along previously carved patterns. Figure 1 shows the final result of another painting session.

7. Conclusions

We have presented the first system for the simulation of dendritic painting. For doing that, we leveraged the vast amount of previous work on these fascinating patterns, extending and adapting it to an interactive painting context. Our methods expands the current tools for the digital painter, while producing mesmerizing procedural animations that can be used in motion graphics.

In this work we focused on the core experience of this peculiar type of painting. However, additional interesting effects are yet to be explored. We set the parameters for our phenomenological model empirically to match the features of recurrent observed behaviors. Due to the inherent simplification, it is possible that our model may miss to reproduce some complex behaviors. Further research on the interaction of the ink and the solvent would allow to simulate some violent reactions we have seen in our real experiments that lead to more intricate behaviors of the ink inside the patterns. Concerning neighboring patterns, we choose boundary conditions that prevent the patterns from touching, as evidenced in real painting videos[†]. In some cases neighboring patterns may merge. However, this effect is present under larger ink amounts, where the flow of ink dominates the behavior. We can alter the blocking factor of the ink boundary conditions when the flow of ink is large.

Also, given the liquid state of the catalyst medium, additional interactions can be enabled my manipulating the whole simulation domain using arbitrary velocity fields, or interactive fluid simulations, as done in digital marbling simulations.

In the same spirit of some previous work, future research may include simulation over 3D manifolds for 3D object ornamentation. The extension of our model to generate 3D patterns is also an interesting line of research, enabling never-seen-before volumetric dendritic painting for immersive media.

Beyond the specific application of dendritic painting, our work shows how to approximate a complex multi-phase fluid problem through a phenomenological model that couples pattern growth and a simpler fluid simulation. We believe this could be inspiring for the simulation of other dendritic growth phenomena or other complex painting techniques. One example is acrylic pour painting, where cell shaped structures emerge by mixing acrylic paint with silicone, and whose growth is affected by temperature.

Acknowledgments. This work was funded in part by the Spanish Ministry of Science (RTI2018-098694-B-I00 VizLearning).

References

[Bog01] BOGOYAVLENSKIY V. A.: Mean-field diffusion-limited aggregation: A "density" model for viscous fingering phenomena. *Phys. Rev. E 64* (Nov 2001), 066303. 3

[Boh11] BOHNETT D.: Artwaters new unique abstract painting technique crosswaters kit. https://www.youtube.com/watch?v=ZGI9y9N3jlg, April 2011. 1

[Bri08] BRIDSON R.: *Fluid Simulation*. A. K. Peters, Ltd., Natick, MA, USA, 2008. 8

[CAS*97] CURTIS C. J., ANDERSON S. E., SEIMS J. E., FLEISCHER K. W., SALESIN D. H.: Computer-generated watercolor. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1997), SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., pp. 421–430. 1, 2

[CKIW15] CHEN Z., KIM B., ITO D., WANG H.: Wetbrush: Gpu-based 3d painting simulation at the bristle level. *ACM Trans. Graph. 34*, 6 (Oct. 2015), 200:1–200:11. 1, 3

[CT05] CHU N. S.-H., TAI C.-L.: Moxi: Real-time ink dispersion in absorbent paper. *ACM Trans. Graph. 24*, 3 (July 2005), 504–511. 1, 2, 5, 6

[DKMI13] DIVERDI S., KRISHNASWAMY A., MÄCH R., ITO D.: Painting with polygons: A procedural watercolor engine. *IEEE Transactions on Visualization and Computer Graphics 19*, 5 (May 2013), 723–735. 1, 3

[EWK*13] ECHEVARRIA J. I., WILENSKY G., KRISHNASWAMY A., KIM B., GUTIERREZ D.: Computational simulation of alternative photographic processes. *Computer Graphics Forum 32*, 4 (2013), 7–16. 5

[†] https://youtu.be/hZy4kGqoJq8?t=453

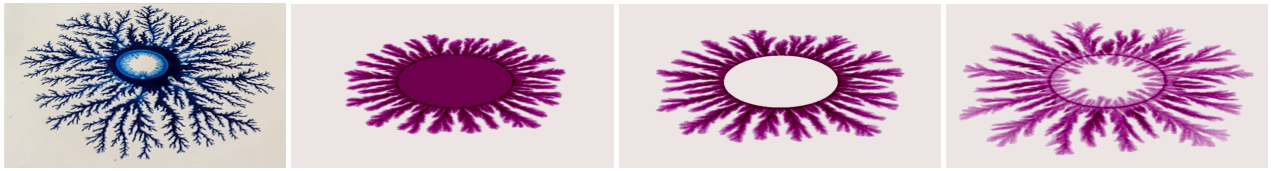

Figure 14: *During a painting session, some artists experiment with adding medium on top of a grown pattern. This makes the internal part of the pattern to grow towards this fresh medium. Our simulations support this creative decision, with results mimicking the real effect as can be seen in the accompanying video. From left to right: real example after medium has been added, and different frames from one of our simulations included in the supplementary video. In the simulation, we did not try to match the color of the real painting.*

[GKCBJ98] GOLDING I., KOZLOVSKY Y., COHEN I., BEN-JACOB E.: Studies of bacterial branching growth using reaction–diffusion models for colonial development. 3

[GLX17] GUO Y., LIU X., XU X.: A unified detail-preserving liquid simulation by two-phase lattice boltzmann modeling. *IEEE Transactions on Visualization and Computer Graphics 23*, 5 (May 2017). 6

[HM92] HAASE C. S., MEYER G. W.: Modeling pigmented materials for realistic image synthesis. *ACM Trans. Graph. 11*, 4 (Oct. 1992), 305–335. 7

[KCGBJ99] KOZLOVSKY Y., COHEN I., GOLDING I., BEN-JACOB E.: Lubricating bacteria model for branching growth of bacterial colonies. *Phys. Rev. E 59* (Jun 1999), 7025–7035. 4

[KHL04] KIM T., HENSON M., LIN M. C.: A hybrid algorithm for modeling ice formation. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Goslar Germany, Germany, 2004), SCA '04, Eurographics Association. 3

[Kit97] KITSUNEZAKI S.: Interface dynamics for bacterial colony formation. *Journal of the Physical Society of Japan 66*, 5 (1997). 3

[KL03] KIM T., LIN M. C.: Visual simulation of ice crystal growth. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2003), SCA '03, Eurographics Association, pp. 86–97. 3

[KMM*97] KAWASAKI K., MOCHIZUKI A., MATSUSHITA M., UMEDA T., SHIGESADA N.: Modeling spatio-temporal patterns generated bybacillus subtilis. *Journal of Theoretical Biology 188*, 2 (1997), 177 – 185. 3, 4

[Kob93] KOBAYASHI R.: Modeling and numerical simulations of dendritic crystal growth. *Physica D: Nonlinear Phenomena 63*, 3 (1993), 410 – 423. 3

[Kow17] KOWALSKI K.: Satisfying mocha diffusion compilation. https://www.youtube.com/watch?v=WjocYCaPsUM, November 2017. 1

[Lew84] LEWIS J.-P.: Texture synthesis for digital painting. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques* (1984), SIGGRAPH '84, pp. 245–252. 3

[LL00] LALLEMAND P., LUO L.-S.: Theory of the lattice boltzmann method: Dispersion, dissipation, isotropy, galilean invariance, and stability. *Phys. Rev. E 61* (2000), 6546–6562. 6

[LVR05] LAERHOVEN T., VAN REETH F.: Real-time simulation of watery paint. *Journal of Visualization and Computer Animation 16* (07 2005), 429–439. 2

[Mou92] MOUKARZEL C.: Laplacian growth on a random lattice. *Physica A: Statistical Mechanics and its Applications 190*, 1 (1992), 13 – 23. 5

[NKGPSP37] N. KOLMOGOROV A., G. PETROVSKII I., S. PISCOUNOV N.: Étude de l'équation de la diffusion avec croissance de la quantité de matiére et son application a un probléme biologique. *Moscou Univ. Math. Bull. 1* (01 1937). 3

[RHLH18] REN B., HUANG J., LIN M. C., HU S.-M.: Controllable dendritic crystal simulation using orientation field. *Computer Graphics Forum 37*, 2 (2018), 485–495. 2, 3

[SDHD16] STUYCK T., DA F., HADAP S., DUTRÉ P.: Real-time oil painting on mobile hardware. *Computer Graphics Forum 36* (09 2016). 3

[SKK10] SHIN S.-H., KAM H. R., KIM C.-H.: Hybrid simulation of miscible mixing with viscous fingering. *Computer Graphics Forum 29*, 2 (2010), 675–683. 3

[Sta99] STAM J.: Stable fluids. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (1999), SIGGRAPH '99, pp. 121–128. 7

[SVBC16] SEGALL A., VANTZOS O., BEN-CHEN M.: Hele-shaw flow simulation with interactive control using complex barycentric coordinates. In *Proc. of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2016), SCA '16, pp. 85–95. 3

[SY02] SUCCI S., YEOMANS J.: The lattice boltzmann equation for fluid dynamics and beyond. *Phys Today 55* (01 2002). 6

[TAAS13] THAMPI S. P., ANSUMALI S., ADHIKARI R., SUCCI S.: Isotropic discrete laplacian operators from lattice hydrodynamics. *Journal of Computational Physics 234* (2013), 1 – 7. 5

[Thü03] THÜREY N.: *A single-phase free-surface Lattice Boltzmann Method*. Master's thesis, University of Erlangen-Nuremberg, Germany, 2003. 6

[TR09] THÜREY N., RÜDE U.: Stable free surface flows with the lattice boltzmann method on adaptively coarsened grids. *Computing and Visualization in Science 12*, 5 (Jun 2009), 247–263. 6, 8

[Tur52] TURING A. M.: The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences 237*, 641 (1952), 37–72. 3

[Tur91] TURK G.: Generating textures on arbitrary surfaces using reaction-diffusion. In *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1991), SIGGRAPH '91, ACM, pp. 289–298. 3

[WK91] WITKIN A., KASS M.: Reaction-diffusion textures. In *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1991), SIGGRAPH '91, ACM, pp. 299–308. 3

[WLWL10] WAN L., LIU X., WONG T., LEUNG C.: Evolving mazes from images. *IEEE Transactions on Visualization and Computer Graphics 16*, 2 (March 2010), 287–297. 3

[WS83] WITTEN T. A., SANDER L. M.: Diffusion-limited aggregation. *Phys. Rev. B 27* (May 1983), 5686–5697. 3

[YJC*13] YOU M., JANG T., CHA S., KIM J., NOH J.: Realistic paint simulation based on fluidity, diffusion, and absorption. *Computer Animation and Virtual Worlds 24*, 3-4 (2013), 297–306. 1

[Zam17] ZAMOR M.: Dendrite fractals : Easy technique - great results - different styles - acrylics or india ink. https://www.youtube.com/watch?v=hZy4kGqoJq8, November 2017. 2