D Approches phénoménologiques

- *Flow noise : textural synthesis of animated flow using enhanced Perlin noise* (Siggraph'01 Technical Sketches)

- *Phenomenological Simulation of Brooks* (EWAS'01)

- *Qualitative Simulation of Cloud Formation and Evolution* (EWAS'97)

Flow Noise

Contact
Ken Perlin
New York University
www.mrl.nyu.edu

Fabrice Neyret
IMAGIS

Flow textures defined with shaders that use Perlin Noise can look great, but they don't "flow" right, because they lack the swirling and advection of real flow. We extend Perlin Noise so that shaders that use it can be animated over time to produce flow textures with a "swirling"quality. We also show how to visually approximate advected flow within shaders.

Rotating Gradients

The classic Perlin Noise function[1] can be described as a sum of overlapping pseudo-random "wiggle" functions. Each wiggle, centered at a different integer lattice point (i; j; k), consists of a product of a weight kernel K and a linear function (a; b; c)i;j;k. K smoothly drops off away from (i,j,k), reaching 0 in both value and gradient at unit distance. Each (a; b; c)i;j;k = a(x_i)+b(y_j)+ c(z _ j) that has a value of 0 at (i,j,k). The result of summing all these overlapping wiggle functions: noise has a characteristic random yet smooth appearance.

Our modification is to rotate all the linear vectors (a; b; c)i;j;k over time, which causes each wiggle function to rotate in place. Because all the (a; b; c) vectors were uncorrelated before the rotation, they will remain uncorrelated after the rotation, so at every moment the result will look like Perlin Noise. Yet over time, the result will impart a "swirling" quality to flow. When multiple scales of noise are summed together, we make the rotation proportional to spatial frequency (finer noise is rotated faster), which visually models real flow.

Pseudo-Advection

Beyond swirling, fluids also contain advection of small features by larger ones, such as ripples on waves. This effect tends to stretch persistent features (for example, foam), but not newly created ones (for example, billowing), to varying degrees, according to their rate of regeneration, or structure memory M.

Traditional Perlin Turbulence is an independent sum of scaled noise, where the scaled noise is bi(x) = noise(2ix)=2i, and the turbulence is tN(x) =PN i=0 bi(x). This can define a displacement texture color(x) = C(x + ItN(x)), where C is a color table and I controls amplitude. Our pseudo-advection displaces features at scale i + 1 and location x0 in the noise domain to x1 = x0 + k ti(x0), where k is the amplitude of the displacement (see below). For small displacements, this can be approximated by x1_k ti(x1), so displacement k is proportional to an amplitude I specified by the user. We can scale this by desired structure memory M, since passive structures are totally advected, when M = 1, while very active structures are instantaneously generated, thus unstretched, when M = 0. Our advected turbulence function is defined by modifying the scaled noise to: bi(x) = b(2i(x_IM ti_1(x)))=2i and using this to construct the sum tN(x) = PN i=0 bi(x).

Results

Many flow textures can be created; some can be viewed at mrl.nyu.edu/flownoise/

Reference
1. Ebert, D. et al. (1998). *Texture and modeling*. Morgan Kaufmann Publishers, July 1998.

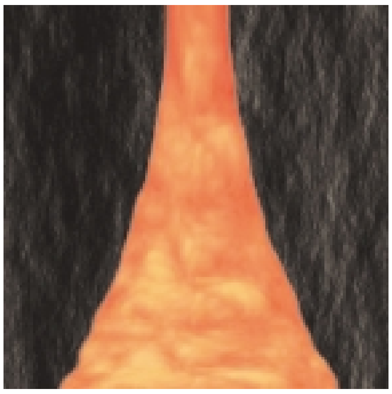
Figure 1. Lava flow.

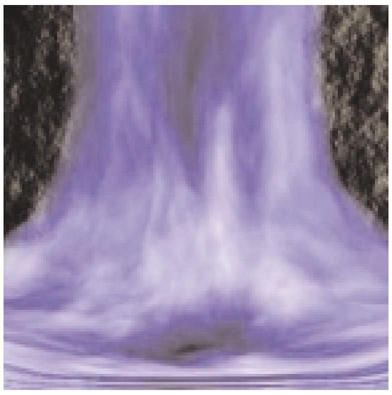
Figure 2. Waterfall.

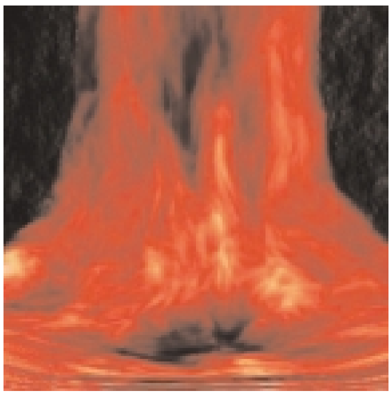
Figure 3. Waterfall with lava-flow texture.

Phenomenological Simulation of Brooks

Fabrice Neyret Nathalie Praizelin
Fabrice.Neyret@imag.fr NathaliePraizelin@wanadoo.fr
iMAGIS-GRAVIR/IMAG-INRIA

Abstract. The goal of our work is to simulate the shape and variations of the water surface on non-turbulent brooks both efficiently and at very high resolution. In this paper, we treat only the shape and animation. We concentrate on the simulation of quasi-stationary waves and ripples in the vicinity of obstacles and banks, and more particularly, shockwaves. To achieve this, we rely on phenomenological laws such as the ones collected over the last two centuries in the field of hydrodynamics: most of the visually interesting phenomena (apart from turbulence) are known qualitatively and characterized in reasonably simplified situations. It is thus wasteful to run a full-range Navier-Stokes simulation for quiet flows when only qualitative results are needed. The complexity of the velocity field along the streambed and around the obstacles is taken into account by solving a simple Laplace equation, assuming a stationary irrotational non-compressible ideal 2D flow. We obtain a stationary solution of the surface waves, that we perturb in order to get a quasi-stationary brook simulation. This yields a real-time simulation of the fluid visible features.

Keywords: natural phenomena, fluids, phenomenological simulation, interactive simulation.
URL: `http://www-imagis.imag.fr/Membres/Fabrice.Neyret/brooks/`

1 Introduction

Computer Graphics researchers and artists have been interested in reproducing the natural look and natural phenomena for a long time. However, objects resulting from fluid motion such as clouds, smoke, fire, wind, ocean waves, rivers or cascades are particularly difficult to simulate; especially from the engineering point of view that corresponds to the current trend for CG fluids.

These simulations are difficult as the physics is complex (and the parameters are not always known); its numerical solving is very expensive (which gets worse very quickly with spatial resolution); the needed spatial range is large (landscape scale); the visible characteristics are only emerging phenomena (*i.e.,* they are not explicitly modeled). While every human observer has a common knowledge about how a cloud, a brook or the ocean should look, and expects to see details as small as its retinal resolution allows. Moreover, as an element of a movie or a game, the artist needs to have control of some of the visible features of the fluid.

Our long term goal is the visually realistic efficient simulation of a brook, if possible in real time. For the moment, we only deal with non-turbulent brooks (in particular, without hydraulics jumps). In the scope of this paper, we concentrate only on shape and animation.

Water is a continuous (and transparent) medium, thus the motion of particles cannot be seen, except if an object (*e.g.,* a leaf) is carried with the flow. The only feature that can be seen is the air-fluid interface, mainly through its reflection of the sky and its refraction of the brook bottom. Therefore, the only useful problem is the determination of the fluid surface. In a perfectly regular flow, the surface is indeed stationary, even if the fluid velocity is large. This surface can be calculated from the 2D velocity field by the geometric construction of stationary waves. In this paper, we deal only with shockwaves and ripples, which are very salient features (having high frequency), as illustrated on Fig. 1. In reality, instabilities make the flow oscillating: a realistic brook is only quasi-stationary, and this beating effect is an important part of our intuition of alive water flow. Therefore, our simulation has to take this aspect into account.

The paper is structured as follow: we review in section 2 the various methods introduced in CG to produce simulations of fluids and in section 3 the physics we rely on. This yields to the principles and structure of our method, exposed in section 4, which we detail in the following sections: we describe the numerical solving yielding the stationary velocity field in section 5; the geometric construction of stationary shockwaves in section 6; the field perturbation process yielding quasi-stationarity in section 7; the complete shockwave structure allowing quasi-stationarity in section 8; results in section 9. As the current stage is only a first step in a long term project, we give some milestones for future work in section 10.

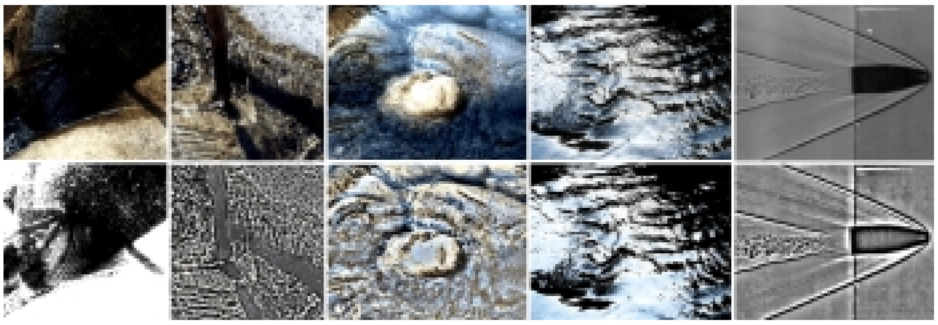

Fig. 1. The shockwaves and ripples features we are interested in (the bottom images are contrast enhancement of the top images). NB: the fluid in the right image (courtesy N.T. Clemens, University of Texas at Austin) is air: it shows front and back shockwaves, a wake, and thin ripples along the object, as for water. However it doesn't show the ripples in front of shockwaves that exist on water, due to surface tension.

2 Previous Work

The various CG approaches aiming at simulating fluids divide into 3 families:

- CFD (Computational Fluid Dynamics) inspired simulations corresponding to the current trend in CG [11, 6, 7, 22, 9]. They generate very rich visual results, reproducing the complexity of running fluids at the price of high computation costs; except for [9] which deals with 2D fluids and [22] that introduces a scheme allowing stability even out of the time step range required by the physical simulation.
- Signal processing approaches [19, 21, 4, 13, 24], also based on intensive calculations (far less than CFD, however), can be qualified as 'phenomenological' in that they aim at reproducing the effects (spatio-temporal shape or force field) without looking at the causes, oppositely to CFD. The problem is that statistical models tend to lose persistent features (*e.g.,* eddies) and that a good model accounting for static distribution (*e.g.,* for clouds density) doesn't trivially yield a good animated model. Among the mentioned papers, only [13, 24] deal with water (ocean waves).
- Empirical or phenomenological animations [8, 15, 27, 28, 10] rely on simplified analytical models: trochoids for ocean waves [8, 15, 10]; Laplace fields for wind [27]; hydraulics for rivers [28]. Some were introduced in early CG at a time where CFD wasn't affordable. New models are also proposed regularly since they offer a good appearance/cost ratio.

Note that spectrum-controlled signals such as Perlin noise [16, 17] are a useful ingredient for empirical animations: they are used in this spirit for cloud density in [4] and for random waves on rivers in [25]. It can also be used to account for small scale (*i.e.,* subgrid) animation in physical simulations, e.g. for fluids in [23] (stochastic interpolation) and for brooks in [26] (noise following the streamlines).

Note also that among all the mentioned papers, very few can deal with brooks:

- CFD can model turbulent running water very well, but cannot easily account for shockwaves and surface tension ripples. This would require very high resolution and dedicated solvers;
- Signal processing approaches give great results on wide areas (*e.g.,* ocean, wind) but cannot easily apply to narrow streams with obstacles since the statistics change at every location;
- Empirical models have been used for some aspects of brooks but not for visual features as crucial as shockwaves and ripples. BTW, these features have never been accounted for in CG.

Looking for information to develop our phenomenological model, we found specialized matter about hydraulics, waves theory, shockwaves and ripples in [12, 1, 2, 18]. We also looked into general fluid mechanics textbooks such as [5, 14]. We got very nice illustrations of real fluids features in [20, 3].

3 Tools from the physics

3.1 Waves theory

Propagation of waves on water surface corresponds to displacement waves and has strong similarities with sound waves in air, which are compression waves. The most important difference is velocity: the velocity of sound is constant in standard conditions (temperature, density), while water waves are *dispersive*, *i.e.,* their velocity depends on their wavelength λ: capillary waves ($\lambda < 1.7cm$) and gravity waves ($\lambda > 1.7cm$) go faster than intermediate waves ($\lambda = 1.7cm$ corresponds to the slowest waves, which have a 23 cm/s velocity). As compared to air, this means that whatever the fluid velocity (in a range), a stationary wave can exists (*i.e.,* having an adequate wavelength).
'Dispersive' means that a focused packet of waves tends to spread out. This implies that the energy doesn't travel at the same speed as waves in deep water: the *group velocity* is $\frac{1}{2}$ of the *phase velocity* for gravity waves and $\frac{3}{2}$ for capillary waves. This has numerous consequences, from the typical shape of ship waves to the location of stationary waves according to their wavelength. However, when the wavelength is greater than the fluid layer depth h (the *shallow water* case), this dispersive behavior disappears and the waves' velocity becomes constant, equals to $\sqrt{gh}$ with g the gravity acceleration.

When dynamic instabilities are negligible a stationary flow triggers only stationary waves. By construction, stationary waves have fronts whose propagation exactly opposes the flow velocity $\overrightarrow{V}$. If the propagation angle of this front, relative to upstream, is θ its velocity is thus $||\overrightarrow{V}||.cos\theta$. Note that since the energy does not propagate at the same speed as waves it is carried along by the flow, at a speed of $\frac{V}{2}$ for gravity waves, thus showing energy in a cone of aperture 19.5^o downstream (for ships... and ducks!). For the same reason capillary waves can propagate upstream but they dissipate quickly.

The case of brooks corresponds to shallow water, so gravity waves are not dispersive: we can tell about a constant wave velocity c as in air (although it depends on the layer depth). This is why the main wave features (*e.g.,* shockwaves) are similar to the soundwave experiments in air (see Fig. 1.5). However, the smallest waves remain dispersive; this is why there are ripples on water surface, *e.g.,* upstream of shockwaves.

3.2 Geometric properties of shockwaves

The *Froude number* $F_r = V/c$ is equivalent to the *Mach number* for sound, *i.e.,* shockwaves occur when the flow runs faster than $Froude = 1$. Brooks are generally supercritical (*i.e., $Froude > 1$*). In an homogeneous velocity field, the shockwave generated by a small object (*e.g.,* a stick) is a 2D cone (*i.e.,* 2 lines, which we call *left* and *right* Froude lines, streamwise), whose aperture α is equal to $arcsin(\frac{c}{V})$, as for a supersonic bang cone (see Fig. 2.1). Note that the slope α is very sensitive to V: it is $\frac{\pi}{2}$ for $F_r = 1$; $\frac{3\pi}{8}$ for $F_r = 1.08$; $\frac{\pi}{4}$ for $F_r = 1.4$. If the field is not homogeneous, the shockwave curves so that it locally fits to this angle relative to $\overrightarrow{V}$ (same if c varies, due to a change in depth), see Fig. 2.2. A consequence is that left lines (resp. right lines) originary from different locations never intersect.

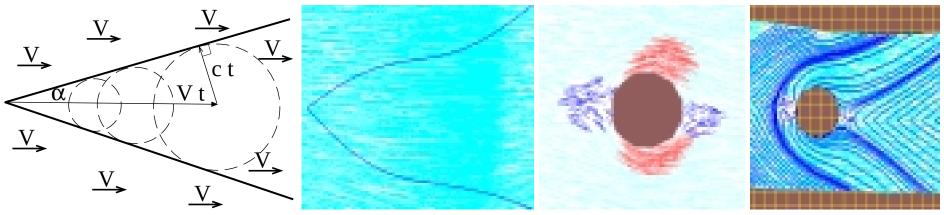

Fig. 2. *Left:* shockwave 'cone', consisting of 2 Froude lines having a slope α. *Middle left:* the lines' slope changes with flow velocity. *Middle right:* flow around an obstacle. The flow, coming from the left, is globally supercritical. It is slowed down upstream and downstream of the obstacle inducing subcritical areas (*i.e., $Froude < 1$*, in dark); it is accelerated on the obstacle sides (grey areas correspond to $Froude > 2$). *Right:* convergence or divergence of Froude lines and accumulation on shockwaves location. The 3 images were generated using our simulator.

Note that as for supersonic air flows, the water surface is potentially covered with slight shockwaves called *Froude waves*, triggered by every small disturbance of the flow. The previous remark implies that in an area with decreasing velocity the upstream lines tend to converge towards the downstream lines, which have a greater aperture. Conversely, for an increasing velocity the downstream lines tend to converge towards the upstream lines. An extreme case occurs if a line starts orthogonally to the flow (*i.e.,* on a location with $F_r = 1$): it constitutes the asymptote for all the Froude waves around. This accumulation of perturbations is a geometric interpretation of shockwaves (see Fig. 2.4).

The stream runs around obstacles in the flow (*e.g.,* stones) but it is slowed down in the areas immediately upstream and downstream of an obstacle and accelerated on the sides (see Fig. 2.3). For this reason, there are always several critical transitions around an obstacle where shockwaves are triggered (see Fig. 5). A shockwave is orthogonal to $\overrightarrow{V}$ at the location where $Froude = 1$ then it curves as it enters areas with faster velocity.

The surface flow traversing a shockwave suddenly slows down to $Froude < 1$. But it progressively retrieves its velocity thus possibly triggering another (small) shockwave. This cycle can repeat several times. The regular patterns of ripples on the sides of obstacles (as well as along gutters) and of herringbones on fast areas in the stream (see Fig 1.4), are probably linked to this phenomenon.

4 Our Method

The arguments in section 3.1, in the case of brooks, lead us to solve the flow as if it were non-dispersive and then to 'dress' the solution to add the details (*i.e.*, ripples) coming from the dispersive part of the flow. Section 3.2 contains all the necessary ingredients to geometrically build the shockwaves in the non-dispersive case. We will turn the stationary velocity field precalculated by CFD into a real-time quasi-stationary flow using perturbations in the spirit of [27]. This yields the following pseudo-algorithm, containing the various tasks to be solved:

Velocity field construction:

- we first need to build the stationary velocity field corresponding to a given brook, taking into account banks, obstacles and depth variations (see section 5).

Shockwaves construction:

- Next, we need to determine the starting point of shockwaves: as suggested, they are on the most upstream location on the isovalue curves of $F_r = 1$. Since areas of $F_r < 1$ are immediately upstream and downstream of obstacles, we can check for the departure of these isocurves along the obstacle boundary then follow these curves in the water up to the targeted location. It is the location where the Froude waves, orthogonal to the flow, are tangent to the curve: we choose for characterizing criterion $\overrightarrow{V} \perp iso_{F_r=1}$. If this location does not exist on the curve (*e.g.*, if the area is along a bank) then the starting location is the upstream end of the curve.
- Once a starting point is obtained, we have to draw the left and right lines of the shockwave. Each line is drawn simply by tracing successive segments whose slope relative to the local $\overrightarrow{V}$ is $arcsin(\frac{c}{V})$.

Ripples construction:

- Ripples are drawn at the same time immediately upstream the shockwave and parallel to it. As they fade quickly, we simply draw 4 of them with a given offset.
- The other type of ripples, looking like Froude waves starting at the obstacle sides, have to be drawn. As suggested before, we assume that they are caused by repeated weak shockwaves. We look for the supercritical areas on the boundaries (*i.e.*, obstacles and banks) in which we launch Froude wave lines separated by a given offset.

5 Stationary Velocity Field

This section deals with the CFD precomputation of the stationary velocity field.

5.1 Physical modelisation

We assume that brooks are quasi-stationary, which we will simulate by perturbing a stationary solution (moreover, the stationary field is a precalculation, while its perturbation is real-time). We assume that the flow is 2D, *i.e.*, that it is not qualitatively different within a vertical column. If we want to account for variable depth, we simply have to weight $\overrightarrow{V}$ with depth in the equations, *i.e.*, to consider the rate of flow instead of the velocity. Water is incompressible in common situations. As we only need a qualitative velocity field, we assume for simplicity that the fluid is ideal (inviscible)[1] and irrotational. With these hypothesis, the fluid is represented by a Laplace equation:

[1]The introduction of viscosity can change the field by triggering the separation of the boundary layer. As solving is a precalculation, we could afford to solve a more comprehensive stationary fluid model. But it would probably show dynamic instabilities, while we want to rely on a stationary field, introducing instability in a separate stage. However, separation points can also be predicted and injected as boundary conditions in the ideal fluid model.

$\Delta\phi = 0$, with $\overrightarrow{V} = grad\phi$ (ϕ is called the potential) [2]. A typical solution is figured in Fig. 3. If we want to account for variations of the depth h, the equation changes to $h\Delta\phi + \nabla h.\nabla\phi = 0$.

Boundary conditions are Neumann kind on banks and obstacles ($\overrightarrow{V}.\overrightarrow{N} = 0$ turns to $\nabla\phi.\overrightarrow{N} = 0$, with $\overrightarrow{N}$ the normal to the boundary) and Dirichlet kind on the two brooks ends: we choose $\phi = \phi_0$ on the upstream end and ϕ_1 on the downstream end. Since there is an extra degree of freedom, we can fix $\phi_0 = 0$. We get ϕ_1 from the brook average velocity estimated by the Chézy formula[3] $V_{av} = C\sqrt{ih}$, where h is the brook depth and i its slope. The constant C depends on the nature of the brook bottom: we choose $40\,m^{1/2}s^{-1}$, corresponding to a low slope mean width river with a bottom made of small stones. If l is the length of the brook segment and z_0, z_1 are the altitude at the two ends, we have $V_{av} = \frac{\phi_1 - \phi_0}{l}$, thus $\phi_1 = C\sqrt{(z_1 - z_0)lh}$.

The user provides an image representing the brook with banks and obstacle in dark (the depth variation in the brook being represented by grey levels) and absolute lengths (size of the image in meters, and the difference of altitude). Note that this image could also come from MNT data, or from a procedural tool.

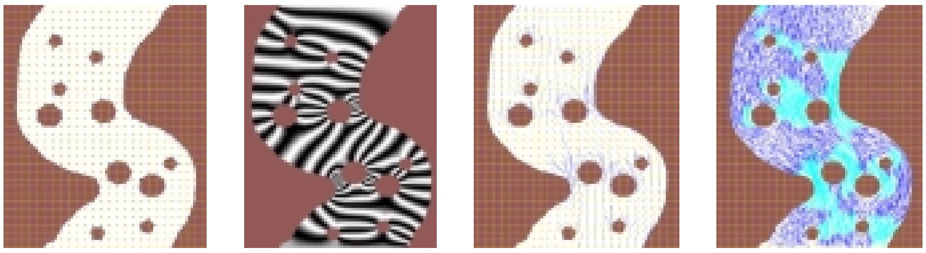

Fig. 3. *From left to right*: the brook painted by the user. The discretization is done using the 32×32 figured grid with a special care for the boundary (*i.e.,* extra nodes); the nodes are the small squares. The potential ϕ resulting of the system solving (interpolated). The corresponding velocity field, at the nodes, and interpolated (supercritical areas are marked in grey).

5.2 Discretization

We use a Finite Difference scheme on a quasi-regular grid, containing the regular grid nodes, plus nodes at the intersection of obstacles boundary with the grid lines (see Fig. 3.1 and Fig. 4). This allows for a better representation of boundary conditions at the price of a more complex discretization scheme: a grid cell can have 3, 4 or 5 corners. We rely on centered schemes for the discretized differential operators. Near the boundary some of the nodes are displaced (see Fig. 4), so we use a quasi-centered scheme with weights $\left(\frac{1}{h_1\frac{h_1+h_3}{2}}, \frac{1}{h_2\frac{h_2+h_4}{2}}, \frac{1}{h_3\frac{h_1+h_3}{2}}, \frac{1}{h_4\frac{h_2+h_4}{2}}, -(\frac{2}{h_1h_3} + \frac{2}{h_2h_4})\right)$ instead of $\left(\frac{1}{h^2}, \frac{1}{h^2}, \frac{1}{h^2}, \frac{1}{h^2}, -\frac{4}{h^2}\right)$ for the Laplacian (see Fig. 4 for notations).

The system solving is sensitive to boundary conditions discretization, so we have to settle them carefully. The Neumann condition $\nabla\phi.\overrightarrow{N} = 0$ tells that ϕ doesn't variate along the direction $\overrightarrow{N}$ so we translate it by linking the ϕ value at the boundary node to

Remark: the existence of a slow boundary layer implies that an iso$_{F_r=1}$ lies along the boundary when the nearby fluid is supercritical, which explains the starting of shockwaves in these locations.

[2] $rot(\overrightarrow{V}) = 0$ (irrotational flow) implies that we can derivate $\overrightarrow{V}$ from a potential ϕ, *i.e.,* $\overrightarrow{V} = grad\phi$. $div(\overrightarrow{V}) = 0$ is thus equivalent to $\Delta\phi = 0$.

[3] The Chézy formula is largely used in hydraulics; it allows estimation of the velocity of a river in functions of various practical global parameters.

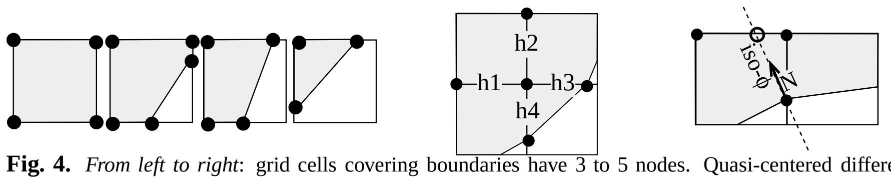

Fig. 4. *From left to right*: grid cells covering boundaries have 3 to 5 nodes. Quasi-centered differential operator discretization scheme. Discretization scheme for the Neumann boundary condition.

its value on the location obtained by projecting this node on the cell sides parallel to $\overrightarrow{N}$ (there are 2 candidate cells for each boundary node and at least 2 candidate borders per cell; see Fig. 4,right)). The value on the projected location also being defined by the interpolation of the 2 closest fluid nodes, this yield one equation per boundary node.

5.3 System solving and use of the solution

We solve this system using a standard solver and we compute $\overrightarrow{V}$ at each node. At the end of this precalculation we have a data structure encoding the quasi-regular grid, with ϕ, $\overrightarrow{V}$ and V values at each node (that we can store on disk). These values have to be interpolated when needed inside a given cell. For regular cells, this is a bilinear interpolation. For boundary cells it is less simple: bilinear interpolation can be used for trapezoid 4 corner cells and linear interpolation for 3 corner cells. Finding a continuous interpolation for 5 corner cells is far from trivial: for interpolating values we preferred cutting these cells into 2 parts of 4 corners.

6 Stationary Shockwaves

This section deals with the geometric construction of the stationary shockwaves. As explained in section 4, we have to find the ends of the iso$_{F_r=1}$ curves on the obstacles boundaries, follow them in order to find the locations where $\overrightarrow{V} \perp iso_{F_r=1}$ then to draw the 2 parts of each shockwave.

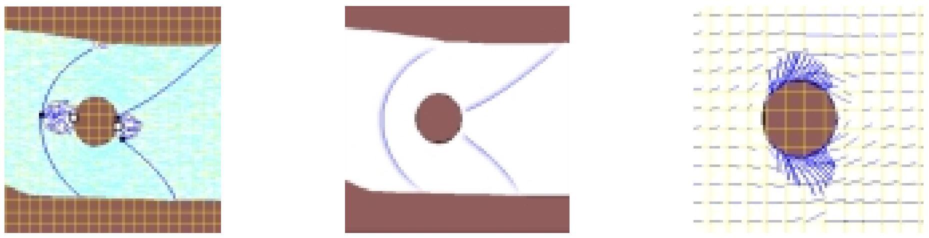

Fig. 5. *Left:* the various geometric constructions calculated in real-time by our simulator. The brook is designed by the user using a classical painter. The velocity field is solved on the 32×32 grid (the flow is coming from the left; the light grey vectors correspond to supercritical flow, the dark grey to subcritical areas). The stop points are figured (the 2 white squares). The curves iso$_{F_r=1}$ are traced in black. The starting locations of shockwaves found on the curves are marked as black squares. The upstream and downstream shockwaves (corresponding to four Froude lines) are shown in dark. *Middle:* the visible features obtained (*i.e.,* the shockwaves and their associated ripples). *Right:* ripples along the sides.

6.1 Finding the iso$_{F_r=1}$ curves extremities

Note that with ideal fluid assumption (no viscosity) the fluid velocity is 0 on the most upstream and downstream locations of the obstacles (called *stop points*). Since the areas $F_r < 1$ are in the vicinity this gives a clue to find the curve: an end should exist on each side of the stop point (see Fig. 6). Thus, we test every pair of boundary nodes until obtaining opposite conditions for them (*i.e.,* $F_r < 1$ for one node and $F_r > 1$

for the other). We estimate the exact end location (where $F_r = 1$) by interpolating. Inside the initial cell (the one containing the stop point) we have to consider this point as a node for interpolation, since the variations of V are non convex in this cell. This allows us to find a curve even in very high velocity fields, for which every grid node is supercritical: the extra 'nodes' corresponding to stop points are the only subcritical ones so the surrounding isocurve is extremely small (see Fig. 6). The fact it exists is sufficient to trigger a shockwave so it is important not to miss it. On the banks there are generally no stop points so we simply have to check every boundary segments.

6.2 Finding the shockwave start point on the iso-curve

We follow the obtained curves in the same spirit as we follow the boundaries: we test the criterion by looking for cells for which $\overrightarrow{V_0}.\overrightarrow{l}$ and $\overrightarrow{V_1}.\overrightarrow{l}$ have opposite signs (with $\overrightarrow{l}$ the current curve segment and $\overrightarrow{V}$ the velocity at segment ends; see Fig. 6,right for notations) or segment ends for which $\overrightarrow{V}.\overrightarrow{l_0}$ and $\overrightarrow{V}.\overrightarrow{l_1}$ have opposite signs. The exact location is obtained again by interpolating. Note that for the isocurve downstream of an obstacle, this location is usually at the 2 ends of the curve (see Fig. 5). We have to deal with special cases for narrow $F_r < 1$ areas which can yield more than 2 intersections of the isocurve with some cells (see Fig. 6). Following the isocurves counter-clock wise (*i.e.*, the $F_r < 1$ area lies on the left), we can discriminate the right path. Note that some isocurves can connect to 2 obstacles and thus might be followed twice. A consequence is that since the curve wouldn't be closed in such case, we have to follow the isocurves from both ends. However this would mean following them twice in the case they are closed. So we mark the end cells of treated isocurves in order to avoid redundancy.

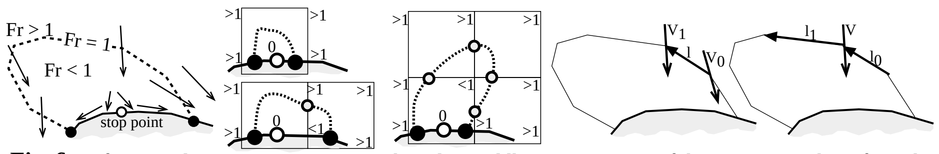

Fig. 6. *Left:* general situation upstream an obstacle. *Middle:* various cases of the iso$_{F_r=1}$ tracking from the Froude value on nodes. *Right:* search for the most upward point on the isocurve, where $V \perp isocurve$.

6.3 Tracing the shockwaves

At this point, we can launch the 2 Froude lines constituting the shockwave. We displace the starting point slightly upstream to avoid the line falling immediately in the subcritical area (where no Froude wave can exist). Then we interactively create segments having a slope α relatively to $\overrightarrow{V}$ and a given length dl, with α given in section 3 (see Fig. 2). This segment is obtained by $\frac{dl}{V}\cos\alpha\,\overrightarrow{V} + \frac{dl}{V}\sin\alpha\,\overrightarrow{V^{\perp}}$, with $\overrightarrow{V^{\perp}}$ oriented left to $\overrightarrow{V}$ for left lines and right to $\overrightarrow{V}$ right lines. Note that no trigonometric function need to be calculated as $\sin\alpha = \frac{c}{V}$ and $\cos\alpha = \sqrt{1 - (\frac{c}{V})^2}$. We stop the lines' construction if an obstacle or a subcritical area is hit. In reality, the energy of the shockwave probably dissipates before this. Since we couldn't find a physical criterion, we have to rely on an arbitrary distance criterion if we want to avoid creating too long shockwaves.

6.4 Drawing the ripples

As suggested in section 3, we draw the associated ripples at the same time as shockwaves. These consist of 4 parallel lines with a given offset and intensity fading with the distance upstream. The Froude ripples on the obstacle sides are generated in the same spirit as above: on the boundary areas in contact with supercritical flow we emit Froude lines with regular offsets.

7 Making the field Quasi-Stationary

To obtain the stationary solution we solved a Laplace equation with boundary conditions. Since it is linear, linear combination of solutions also obey the Laplace equation. In the same spirit as [27], we choose incompressible perturbations (*i.e.,* obeying the Laplace equation) having a small support (being null farther from a given radius, boundary conditions are thus automatically matched). We choose sources and vortices as perturbations to be added to the stationary velocity field, tuned to obey $\nabla \overrightarrow{V} = 0$ (sources can be seen as the above view of horizontal vortex rings). These perturbations correspond to upstream instabilities carried with the flow. We drop randomly, at the upstream end of the brook, particles following the flow associated with a perturbation (see Fig. 7). The velocity used for the visible features construction at a given location is then the sum of the stationary flow and of the various nearby perturbations (for optimization, we set in each cell a list of the perturbations covering it).

Note that, in addition to random perturbations, we can also create regular perturbations (*e.g.,* vortex pairs behind obstacles in order to create von Karman wakes) or perturbations associated to events (*e.g.,* extra obstacles, possibly moving).

8 Quasi-Stationary Shockwaves

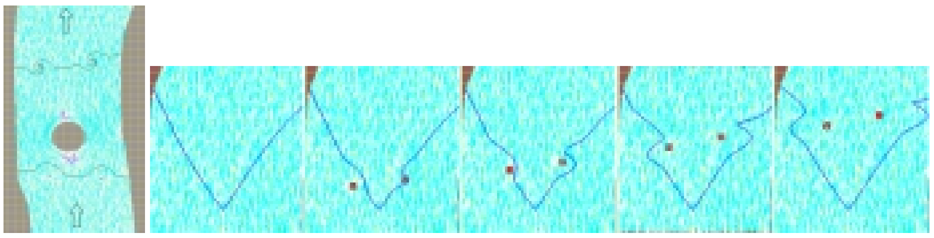

Fig. 7. *Left:* perturbations dropped in the flow in order to make it quasi-static. To visualize the effects, we launched 2 lines of passive particles (linked floaters). Top: vortices. Bottom: sources (*i.e.,* horizontal vortex rings). The perturbations are associated to particles carried with the flow, figured by the small squares. *Right:* Effects of the perturbations on Froude waves. Left particle: source. Right particle: vortex.

8.1 Consequences of quasi-stationarity on shockwaves

The shockwave geometric construction of section 6 is purely static. If we change the value of $\overrightarrow{V}$ at a given location the shockwave drawing would change instantaneously, which is not physically correct (and worse, not realistic). We have to take into account the speed of information transport along a Froude wave which is different to the flow velocity both in value and in direction. If the velocity changes at a given location, this locally changes the slope of the Froude wave, which yields an offset in the whole wave downstream, even if the velocity is unchanged there (see Fig. 7). There is thus 2 causes of change at a given location: local slope change due to local velocity change and change due to upstream change in the line location. Note that other events can perturbate the shockwave: a change in velocity can also change the shape of the curve iso$_{F_r=1}$ and thus the starting location of the shockwave. The appearance of a subcritical area on the path of a shockwave or the sweep of a shockwave up to an obstacle or a subcritical area, will stop its downstream propagation (but the downstream part separated by the interruption will persist, carried with the flow, as illustrated on the animations).

8.2 Structure and simulation of evolving shockwaves

Since we have to update the visible features instead of re-creating them at each time step, we have to store all the vertices of each Froude line. Each vertex is re-evaluated from the local and upstream conditions (*i.e.,* local velocity and previous vertex location). We have to take into account the speed of the information transport along a Froude wave, which is $V_F = \sqrt{V^2 - c^2}$: if the distance between 2 vertices along the line is dl, the time for the perturbation of the first vertex to reach the second is $\frac{dl}{V_F}$, *i.e.,* $\lambda = \frac{dl}{dt V_F}$ times steps. If $\lambda > 1$, more than 1 time step should be necessary to transmit the information to the next vertex. If $\lambda < 1$, the information should be transported farther than this vertex. This is classically simulated using relaxation: in the first case we only transmit $\frac{1}{\lambda}$ of the perturbation to the next vertex, *i.e.,* $P_2^{t+dt} = (1 - \frac{1}{\lambda})P_2^t + \frac{1}{\lambda}(P_1^t + dl\ \overrightarrow{slope}_1)$. In the second case, we immediately transmit the information to all the `int(`$\frac{1}{\lambda}$`)` next vertices, and we do a `frac(`$\frac{1}{\lambda}$`)` relaxation on the next one.

To account for vertices entering in subcritical areas, we add an *invalid* flag to mark such vertices. The flag information is transmitted the same way. This avoids generation of separate vertex lists for orphan Froude line segments, knowing that the line will probably reappear in the near future. For newly created Froude lines, new vertices are created at the downstream end at a rate $\frac{1}{\lambda}$. To avoid visual discontinuities we fade the line intensity close to the open ends of a line segment.

9 Results

Our brook simulator runs in real time. Classically, the simulation time step dt is matched to the measured calculation duration during the previous steps in order to guarantee the true real-time.

The user provides a grey-level image of a brook, encoding in black the outside, and in grey the depth (see Fig. 8). This image can come either from a MNT, procedural tools, or a regular painter (we used *xpaint* for our tests). The user also provides the required grid resolution (32×32 or 64×64 in our tests), the size of the corresponding terrain in meter, the brook depth and the difference of altitude between the brook ends. In our tests, this was generally $32m \times 32m$ for the terrain, $80cm$ for the altitude decrease and $8cm$ for the brook depth.

Our implementation allows the storage of the precalculated stationary velocity field; to modify the Froude number (which is equivalent to changing the global velocity or the brook depth); to visualize the velocity field, the potential ϕ; to drop active (*i.e.,* perturbating) or passive particles in the flow (points, lines or circles); to put and displace 'needles' triggering shockwaves (see animations). Most of these features are for tests but several could be used to interface the simulation with objects or events.

There is no realistic rendering for the moment: the result consists of the drawing of the visible features. In section 10 we evoke possible ways of making realistic rendering.

We have implemented all the features treated in the paper (see Fig. 9 to 11 in Appendix). The limitations have also been mentioned: for the moment we use arbitrary offsets for the ripples, and end criteria to determine where the features should vanish are lacking, yielding too persistent Froude lines (see Fig. 10).

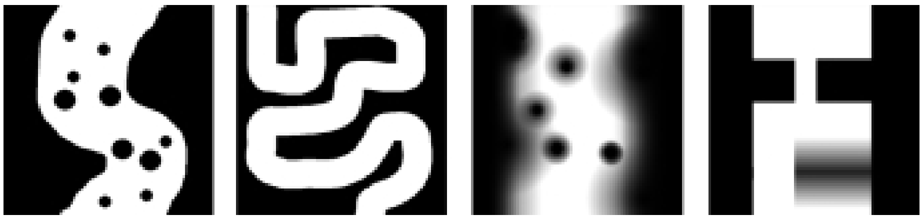

Fig. 8. Various brooks drawings.

10 Conclusion and Future Work

Our long term goal is the real-time detailed realistic rendering of brooks. We have achieved a first step: we have shown how to calculate animated geometric constructions of some important visible features existing on the water surface of brooks by combining simplified precalculated CFD and phenomenological hydraulics laws (accounting for shockwaves and ripples had never be done before in CG). The fact that we obtain geometric constructions (to be used as a skeleton defining the surface) and not a series of velocity and height fields, provides a resolution-independent features: the choice of visual resolution now depends purely on rendering criteria. Moreover, our primitives are very compact, and can be built in real-time by the simulator.
Remaining work to be done is twofold:

- realistic rendering, and more generally, 3D rendering: the main problem is to construct a surface from our geometric primitives. Since they represent waves (or parallel waves, for ripples) on a flat surface, we propose building mesh bands along the lines. Once the surface is defined, reflection and refraction maps should be sufficient to render the water aspect in a real-time framework.
- Improving the features and accounting for more effects: at first, we need criteria for fading and stopping the shockwaves and for tuning the wavelength of ripples. Beyond shockwaves and ripples, we would like to generate herringbones, von Karman wakes, hydraulics jumps, foam, etc... The first is very similar to the ones we treated, except that shockwaves don't occur in the vicinity of obstacles. The second could be achieved by launching regularly vortex particles and associating a visual effect to them, *e.g.,* smooth noise [26]. The third are subject to specialized literature [1] that could be adapted to our framework as well. The fourth are more difficult, but real observation shows that a threshold can separate states with or without air bubbles mixed to the water.

In conclusion, we hope to have illustrated how phenomenological approaches can very efficiently generate interesting features that would be rather expensive to obtain by numerical approaches. These 2 approaches can be combined to benefit from the strength of both: efficiency, resolution, controllability for the first and adaptability to spatial conditions of the second.

Acknowledgments:
We wish to thank Andreas Ruff and Pauline Jepp for proofreading this paper. We also would like to thank François Charru of IMFT, Mark Cramer of Virginia Polytechnic Institute, and Noel Clemens of University of Texas at Austin, for their valuable discussions on fluids and waves.

References

[1] M. Carlier. *Hydraulique générale et appliquée*. Eyrolles, 1980.

[2] P. Chassaing. *Mécanique des fluides. Eléments d'un premier parcours*. Cepadues éditions, 1997.

[3] M.S. Cramer. Gallery of fluid dynamics. http://www.eng.vt.edu/fluids/msc/gallery/gall.htm.

[4] David Ebert, Kent Musgrave, Darwyn Peachey, Ken Perlin, and Worley. *Texturing and Modeling: A Procedural Approach*. Academic Press, October 1994.

[5] R. Feynman. *Lectures on physics*. Addison-Weisley Publishing Compagny, 1977.

[6] Nick Foster and Demitri Metaxas. Realistic animation of liquids. In Wayne A. Davis and Richard Bartels, editors, *Graphics Interface '96*, pages 204–212. Canadian Information Processing Society, Canadian Human-Computer Communications Society, May 1996.

[7] Nick Foster and Dimitris Metaxas. Modeling the motion of a hot, turbulent gas. In Turner Whitted, editor, *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pages 181–188. ACM SIGGRAPH, Addison Wesley, August 1997.

[8] Alain Fournier and William T. Reeves. A simple model of ocean waves. In David C. Evans and Russell J. Athay, editors, *Computer Graphics (SIGGRAPH '86 Proceedings)*, volume 20, pages 75–84, August 1986.

[9] Manuel Noronha Gamito, Pedro Faria Lopes, and Mário Rui Gomes. Two-dimensional simulation of gaseous phenomena using vortex particles. In Dimitri Terzopoulos and Daniel Thalmann, editors, *Computer Animation and Simulation '95*, pages 2–15. Eurographics, Springer-Verlag, September 1995.

[10] Jean-Christophe Gonzato and Bertrand Le Saëc. A phenomenological model of coastal scenes based on physical considerations. In D. Thalmann and M. van de Panne, editors, *Computer Animation and Simulation '97*, Eurographics, pages 137–148. Springer-Verlag Wien New York, 1997.

[11] Michael Kass and Gavin Miller. Rapid, stable fluid dynamics for computer graphics. In Forest Baskett, editor, *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 49–57, August 1990.

[12] J. Lighthill. *Waves in fluids*. Cambridge University Press, 1978.

[13] G. A. Mastin, P. A. Watterberg, and J. F. Mareda. Fourier synthesis of ocean scenes. *IEEE Computer Graphics and Applications*, 7(3):16–23, March 1987.

[14] L.M. Milne-Thomson. *Theoretical Hydrodynamics*. MacMillan & Co LTD, 1968.

[15] Darwyn R. Peachey. Modeling waves and surf. In David C. Evans and Russell J. Athay, editors, *Computer Graphics (SIGGRAPH '86 Proceedings)*, volume 20, pages 65–74, August 1986.

[16] Ken Perlin. An image synthesizer. In B. A. Barsky, editor, *Computer Graphics (SIGGRAPH '85 Proceedings)*, volume 19(3), pages 287–296, July 1985.

[17] Ken Perlin and Fabrice Neyret. Flow noise: textural synthesis of animated flow using enhanced Perlin noise. In *SIGGRAPH 2001 Technical Sketches and Applications*, August 2001.

[18] Jean-Pierre Petit. *Le mur du silence*. Belin. http://www.chez.com/jppetit/mhd.html.

[19] M. Shinya and A. Fournier. Stochastic motion-motion under the influence of wind. *Computer Graphics Forum*, 11(3):119–128, 469, 1992.

[20] Japan society of mechanical engineers. *Visualized flow*. Pergamon Press, Oxford, 1988.

[21] Jos Stam. Stochastic dynamics: Simulating the effects of turbulence on flexible structures. *Computer Graphics Forum*, 16(3):159–164, August 1997. Proceedings of Eurographics '97.

[22] Jos Stam. Stable fluids. In Alyn Rockwood, editor, *Proceedings of the Conference on Computer Graphics (Siggraph99)*, pages 121–128, N.Y., August8–13 1999. ACM Press.

[23] Jos Stam and Eugene Fiume. Turbulent wind fields for gaseous phenomena. In *Proceedings of SIGGRAPH '93*, pages 369–376. ACM SIGGRAPH, 1993.

[24] Sebastien Thon, Jean-Michel Dischler, and Djamchid Ghazanfarpour. Ocean waves synthesis using a spectrum-based turbulence function. In *Computer Graphics International Proceeding*, 2000.

[25] Sebastien Thon, Jean-Michel Dischler, and Djamchid Ghazanfarpour. A simple model for visually realistic running waters. In *Eurographics UK*, 2000.

[26] Sebastien Thon and Djamchid Ghazanfarpour. A semi-physical model of running waters. In *Eurographics UK*, 2001.

[27] Jakub Wejchert and David Haumann. Animation aerodynamics. In Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 19–22, July 1991.

[28] Yingqing Xu, Cheng Su, Dongxu Qi, Hua Li, and Shenquan Liu. Physically based simulation of water currents and waves. *Computers & Graphics*, 21(3):277–280, May 1997.

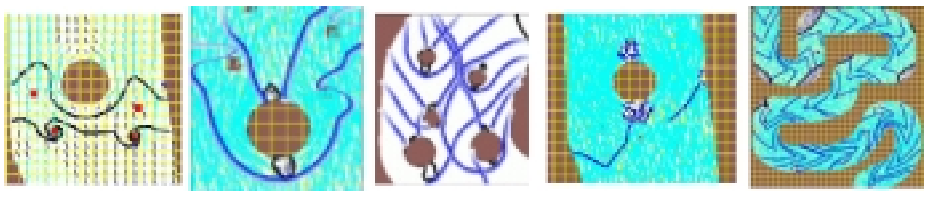
Fig. 9. Source and vortex perturbations make the brook quasi-stationary, showing evolving wave features (images are part of animations, which are available on our web site).

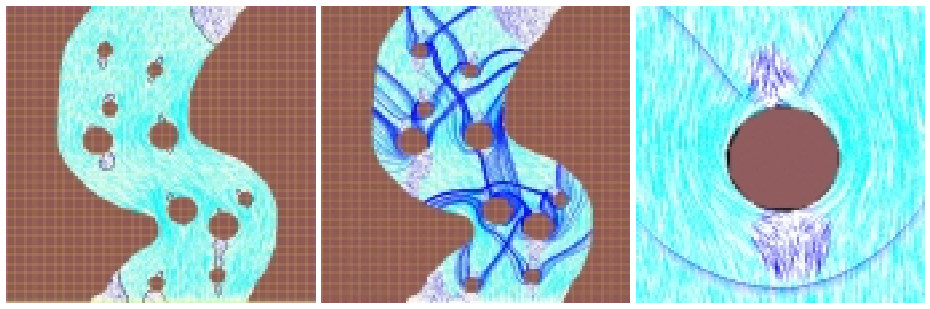
Fig. 10. *Left:* iso$_{F_r=1}$ curves built from the velocity field (the flow comes from the bottom of the image). *Middle:* ripples on obstacles sides (without any stop criterion). *Right:* close view of ripples upstream shockwaves.

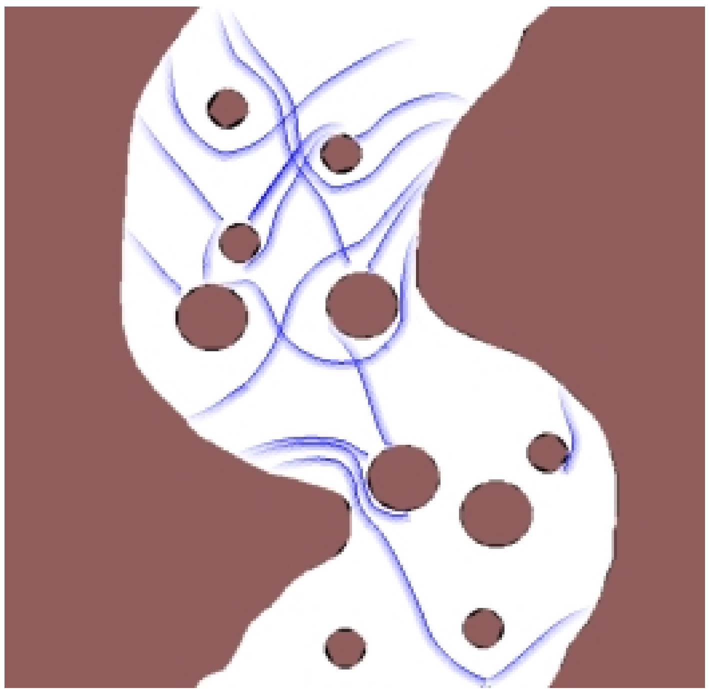
Fig. 11. Shockwaves built upstream and downstream obstacles.

Qualitative Simulation of Convective Cloud Formation and Evolution

Fabrice Neyret

DGP - University of Toronto
SF 4306-C, 10 Kings College Road
Toronto, Ontario, M5S 3G4, Canada
Fabrice.Neyret@inria.fr http://www.dgp.toronto.edu/people/neyret

Abstract.

Cloud simulation models are rare in computer graphics, although many rendering algorithms have been developed to evaluate the illumination and the color of gaseous phenomena. The laws of fluid mechanics used for physical simulation require a fine resolution in space and time, and solving the Navier-Stokes equation in 3D is in general quite costly. However, many heuristics, dealing with various scales, can be used to describe the evolution of the shape of convective clouds such as cumulus. These go beyond the classical equations governing the motion of each element of fluid volume. Physicists characterize the identity and behavior of phenomena such as bubbles, jets, instabilities, waves, convective cells, and vortices. Moreover, the shape of a convective cloud can be considered as a surface, so that we only require detailed information near the periphery of the cloud volume.

A guiding principle in our ongoing research is to take advantage of this type of high level knowledge available at various scales in order to obtain a simulation of convective clouds that may not be physically-accurate, but that will be perceptually convincing.

Keywords: clouds, simulation, natural phenomena

1 Introduction

Atmospheric convective phenomena such as cumulus clouds and billowing smoke are important natural phenomena that often occur in outdoor scenes. However, few animated clouds or smoke generators are available for computer graphics. The evolving fractal shapes of clouds are difficult to model, because what we see is the large scale result of complex non linear physical phenomena, combining the effects of fluid mechanics and thermodynamics. Moreover the numerical values of some of the real parameters required in numerical models (e.g. turbulent viscosity) are not readily known. Real boundary conditions such as heat and moisture generation on the floor are also not well understood. Furthermore, the Eulerian schemes used to solve the equations require a grid having at least the resolution of the smallest detail we wish to see, while the volume of the desired scene may be quite large. Thus, a physical cloud simulation using local equations is a task that is especially consuming in terms of memory and time !

By using a macroscopic approach, atmospheric physicists have characterized many specific structures appearing in fluid phenomena, whose evolution, properties, and interactions can be measured. Some of these structures are as follows (some are shown in Figure 1):

- *Rayleigh-Taylor instability*: when a dense (cold) fluid layer lies on a light (hot) one, bulbs appear at the interface, rising to become 'atomic mushrooms', then becoming *bubbles* once they take off.
- *bubbles*: their ascending velocity, rate of expansion, and cooling through mixing can all be estimated.
- *jets* or *columns*: bubbles often 'drip up' along a vertical chimney. When a train of decelerating bubbles collides, the bubbles merge in a column having more intense behavior because the mixing with background air is less important.
- Once a bubble reaches the dew point, the water vapor it contains starts condensing and makes the rising bubble visible. The release of latent heat of condensation boosts the rising motion, and is the driving force for cumulus phenomena.
- Without mixing, a bubble's temperature decreases by 10 degrees per kilometer until reaching the dew point altitude, and then becomes 6 degrees per kilometer. This is different than the ambient air temperature, whose usual vertical thermal profile decreases by 6.4 degrees per kilometer. Near the floor, this variation is logarithmic, positive in summer and negative in winter. Convective motions can thus only start around summer.
- *Turrets*: bubbles are also born and rise on the top and from the sides of clouds, for the same reasons as near the floor, i.e. difference of density between two layers. The balance of forces tend to make them move in a direction that is halfway between the vertical and the cloud surface normal [14] (see Figure 4).
- *Kelvin-Helmholtz instability*: on the boundary of fluids having different velocities (e.g. a bubble top surface, see D on Figure 2), *waves* appear.
- These waves amplify and turn into *vortices*.
- On the top of a rising bubble (e.g. on a cloud surface), the balance between the mixing and the shearing in the surface layer produces waves with a characteristic size [5, 6] (about 1/10 of the bubble radius).
- *Vortices* tend to break into smaller vortices, up to the quasi-molecular size where the energy can be dissipated as heat. This transfer of energy among vortices of different scales is known as *Kolmogorov cascade*, and its power spectrum can be estimated.
- *Benard cells*: inside a fluid layer with a hot bottom surface and a cold top surface having a Rayleigh number in the correct range, regular convective cells appear. These usually are hexagonal in shape, with the fluid rising at the middle and sinking at the borders.
- Under specific wind conditions, the cells turn into ribbons, orthogonal or parallel to the wind direction depending of the wind velocity.

A classical survey about atmospheric structures can be found in [15]. More about convective clouds is explained in [8]. Advanced topics about plumes and thermals are presented in [1]. Interesting information concerning air motion below and around natural clouds can also be obtained from experienced paragliders [11].

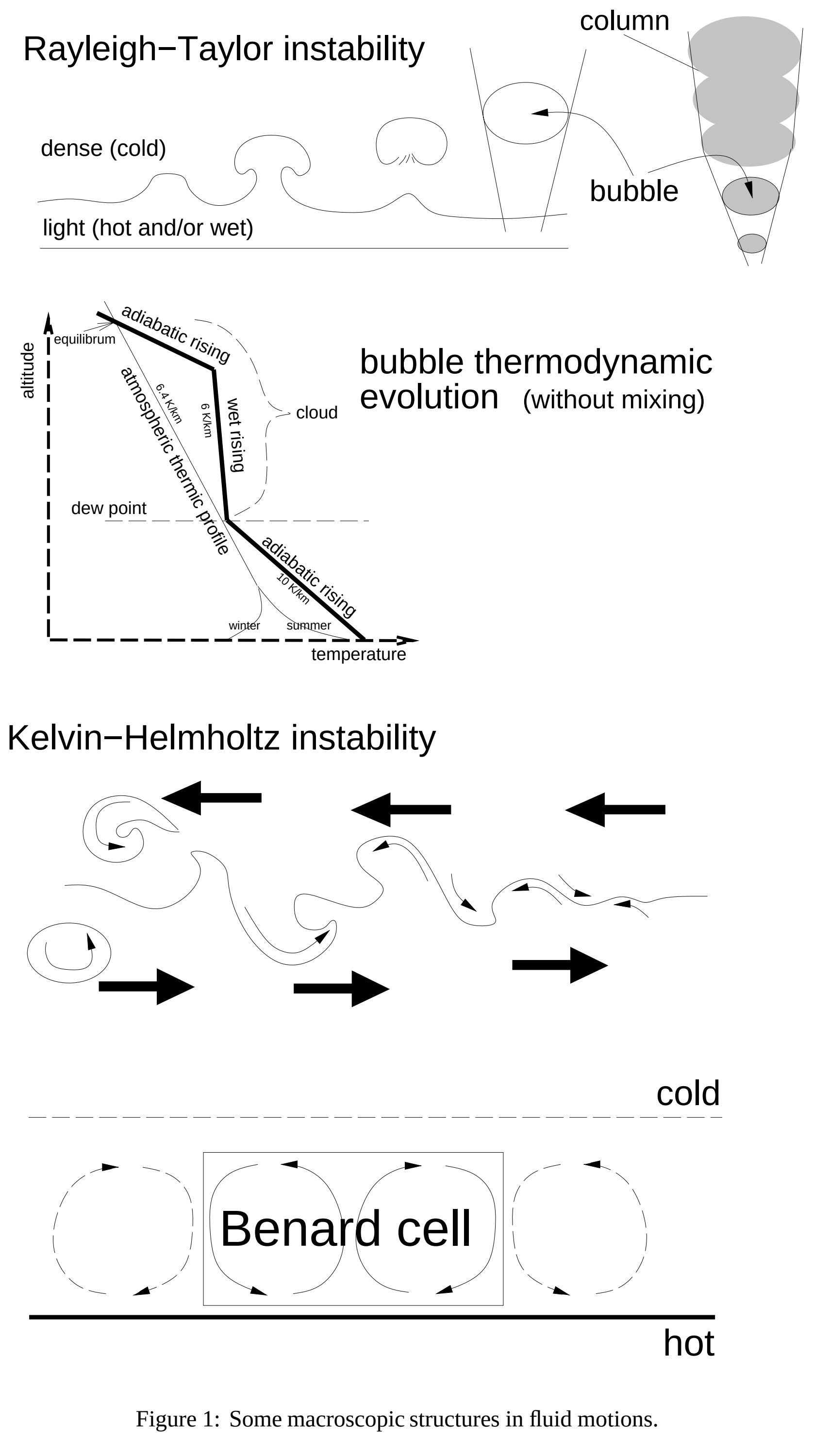

Figure 1: Some macroscopic structures in fluid motions.

We aim to incorporate most of this knowledge in a computer graphics model. In this paper, we take into account 3 phenomena at various scales (see Figure 2):

- hot spot generation on the ground, from which the bubbles rise (see section 3.1). We also hope to have the Benard circulation simulated at this stage.
- we used to simulate bubble rising and dew point reaching. However this model is too complicated (i.e. it contains too many equations), and probably useless. We don't describe it in this paper, rather assuming that the dew point altitude is known and constant, so that the travel between the floor and the cloud bottom doesn't needs to be simulated.
- bubble creation and evolution inside the cloud, and emerging as turrets on the top and the border (see section 3.2).
- waves and vortices convected at the surface of the turrets (see section 3.3).

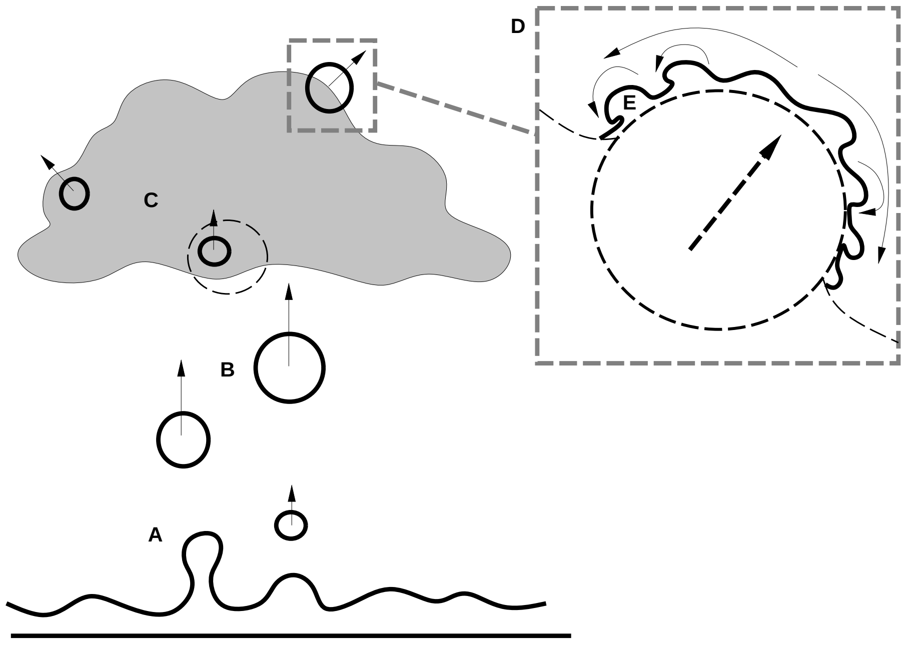

Figure 2: Our general scheme: bubbles birth (A) and rising (B), cloud bubbles motion (C), bubble substructures (D), waves becoming vortices (E).

2 Previous Work

Many physical simulation techniques exist, most of which are not used to produce images. Few physicists are really interested by the shape of the clouds. Instead, they are interested by the heat field, the velocity field, and the formation of rain drops. Simplifications are commonly assumed for efficiency, e.g. 2D vertical simulation, axi-symmetric simulation. In our work we are not interested by such costly simulations which are not particularly tuned for providing a shape.

In 1984, Kajiya proposed a model to achieve some clouds simulation [9]. These equations are very close to the physical models, so that the affordable resolution is very low (10x10x20).

Two papers simulate fluids in the context of computer graphics, and with an appropriately-high resolution [2, 10]. However, these deal only with the 2D case. Alas, not only there is a cost gap between 2D and 3D, but 2D and 3D fluids motions are qualitatively different in nature.

Several computer graphics models exist to produce cloud shapes using various kinds of procedural noise, possibly generating an explicit volumetric density field. Some fractal models exist as well. However, none of these models deal with the growth and animation of the cloud, which is the main goal of our work. We would like to consider that the shape is the consequence of the movment.

In [3, 4], Garner proposes an interesting primitive, the textured ellipsoid, covered with a procedural noise opacity which fades on the horizon of the shape so that one cannot see the elliptical border. In the 1985 paper, he describes how this ellipsoid can rise from the soil and grow, then becomes visible and stabilizes at a given altitude. While the images are detailed and interesting, the animation on the ellipsoid surface is simply obtained using an offset for the Perlin noise, which doesn't correspond to any real convection effect.

In [12, 13], Stam uses white noise filtering to generate a wind field with a mass conservation property and having defined statistical properties. Smoke, flames and cloud shapes are achieved by combining spheres warped by the wind field. The wind model provides for very fluids-like motions, but cannot generates real vortices, because these cannot be described by the second order statistical moments of the fluid velocity.

A new approach is thus required !

3 The proposed model

3.1 Bubble generation

In summer, the sun intensely radiates the surface of the earth. Depending of the type of soil (earth, water, concrete), orientation and vegetation (barren, grass fields, forest), the ground heats the bottom layer of air and provides moisture[1], so that the bottom layer becomes lighter than the neighboring air layers above. At the hottest points above the floor, the air begins to rise, and in doing so, sucks the hot air from the surrounding terrain. Depending on the conditions, this will lead to a single bubble, a train of bubbles, or even a column formation.

We simulate the bottom layer of air as a set of hot air parcels trapped on the ground, modeled as 2D particles. The ground heating due to the sun is modeled by creating new hot air parcels at random. One may use a texture indicating the kind of soil, thus controlling how much energy the ground releases at each point. The rising is due to the difference in air density ρ relative to the upper air layer, which occurs because of the difference in temperature. The buoyancy force per unit mass is $g(T - T^*)/T^*$, where T is the parcel temperature and T^* is the background temperature. However, the buoyancy force is opposed by friction, and the temperature stratification near the ground makes it difficult to define a 'background temperature'. We assume that the rising force is proportional to the heat gained $E = (T - T^*)$ provided by the sun, with an ascendancy coefficient to control the rising ability: the rising force is $f_r(i) = c_{asc}.m(i).E(i)$ for the air parcel i.

We also compute the attraction force created between the parcels by the vacuum effect of the displaced (rising) air. We assume a $1/d^2$ decrease law for this attraction: $f_a(P) = \sum_i \frac{P(i)-P}{\|P(i)-P\|^3}.f_r(i)$. A $1/d$ law would tend to favor a single hot spot attracting all hot air parcels. We also consider a viscosity coefficient to control the stability of the air flux along the surface. From these terms we compute the 2D air transport toward the hot spots. Parcels closer than a given distance are merged. When the rising force exceeds a threshold, which will occur at the hot spots, we consider that a bubble is born, to be given to the model managing the bubble transportation in air. The corresponding mass is subtracted from the remaining parcel of ground air.

The fact that the surrounding air is displaced towards a hot spot supposes that fresh air comes from above, which will decrease the chance for this same location to heat further, thus stabilizing the air transport (main hot spots tend to stay at the same place). We expect that this may emulate Benard cell behavior, but more studies should be done with respect to the resulting distribution of hot spots as a function of the value of the ascendancy and viscosity coefficients, and of the attraction law.

[1]Moist air is lighter than dry air, due to the molecular masses of H_2O and air (80% N_2, 20% O_2). Air density is also inversely proportional to temperature. We can merge the two parameters in one, the *virtual temperature*. Typically, 3% of vapor is roughly equivalent to 3 degrees of heat. Thus, in this whole section, T is the virtual temperature.

3.2 Cloud evolution

As explained in the introduction, we do not model within the scope on this paper the bubble rising from the ground to the dew point altitude. The bubble becomes visible at the dew point altitude because of condensation, thus marking the boundary forming the floor of the clouds. We assume that the bubble generated at the previous stage appears directly at the dew point altitude, above the initial point of formation. One may add some bias in order to take into account horizontal displacements attributable to wind.

Because of the quantitative changes in the mixing phenomenon and of the dynamics of the motion below and above the dew point altitude (the air is more hot, wet and turbulent inside the cloud), one cannot assume that a same well-defined bubble being born on the ground will rise up to the top of a cloud. The matter is redistributed. We consider a model where the cloud is fed from the earth's surface with bubbles, and where new bubbles are created inside the cloud depending on the local conditions[2].

In our model, the inside of the cloud is composed of static bubbles. The volumetric resolution is thus crudely represented, with sufficient accuracy for air transport to occur. We use *potential temperatures*, i.e. the temperature that an air parcel would have if it was at the sea level, rather than the real temperature, the latter being affected by the pressure decrease with altitude.

The birth of a bubble inside or at the border of a cloud is due to a local temperature gradient (causing a Rayleigh-Taylor instability). Our computational model works as follows. We first look for the hottest neighborhood, assuming that the motion is due to newly arriving matter, and to accumulation of hot matter. We then compute the barycenter of this neighborhood using temperatures as weights, where the new bubble will be born. The new bubble takes the heat of each bubble in the neighborhood, according to a factor decreasing with the distance d, namely $\frac{1}{1+d/D_{\frac{1}{2}}}$, and in a proportion controlled by a global coefficient τ_{conv} (between 0 and 1). $D_{\frac{1}{2}}$ is a coefficient corresponding to the distance contributing at 50%. We chose it to be equal to the bubble radius so that the bubble recomposition is quite local.

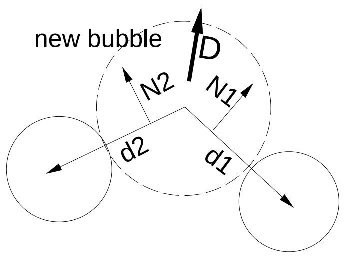

Figure 3: New bubble inside a cloud.

The direction of the bubble displacement depends on the local heat gradient, which is estimated by summing the normals $\vec{N_i}$ (taken in a vertical plane) to each bubble direction $\vec{d_i}$ from which the new bubble is taking heat: $\vec{N_i} = \vec{d_i} \times \vec{horiz}$, $\vec{D} = \frac{1}{nb} . \sum_1^{nb} (\vec{N_i}/||\vec{N_i}||)$, see Figure 3. Note that in a homogeneous temperature field,

[2]Some heuristico-physical models even consider 3 stages between the floor and the clouds, where the bubbles are rearranged, so that the bubbles reaching the cloud are not the same as the ones that left the floor. These models focus on heat and moisture exchange between successive layers.

the direction is null (no instability), while at the boundary of such a field, the direction follows the boundary normal. Beyond this local potential temperature gradient, there is a vertical gradient of real temperature due to the pressure gradient. This has no effect however, as it is directly counterbalanced by gravity. The density gradient is due to the pressure, which results from hydrostatic equilibrium: $\frac{\partial P}{\partial z} = -\rho.g$. We do need to add the important effect of latent heat released by condensation as long as a wet air parcel rises, as illustrated in Figure 4. This acts as a motor in the cumulus growing. For typical physical values, this is equivalent to adding a unit vertical vector to the computed direction, thus verifying the observation that turrets in natural clouds grow in a direction that is the halfway between the cloud normal and the vertical [14].

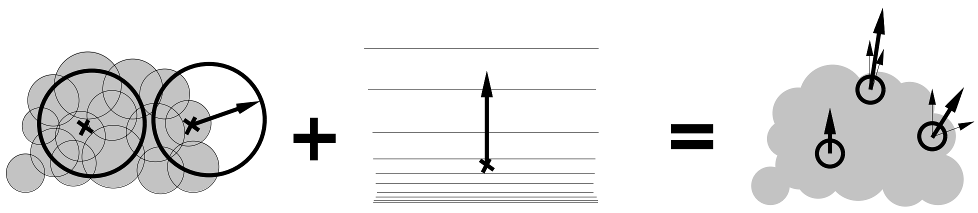
Figure 4: The combination of the potential temperature gradient and the latent heat release controls the bubbles motion inside the cloud, and the turrets formation on the border.

Once the bubble has moved a given distance in its chosen direction, one of several things can occur (see Figure 5):

- if the bubble is in a sparse area, we keep it in order to be used in further air transportation. It marks the occupation of this location by cloud matter.
- if the bubble appears on the top of the cloud, we keep it because it is visible. If this hides a neighbor, we can delete the neighbor.
- otherwise we can destroy it and distribute its mass and heat to its new neighbors.

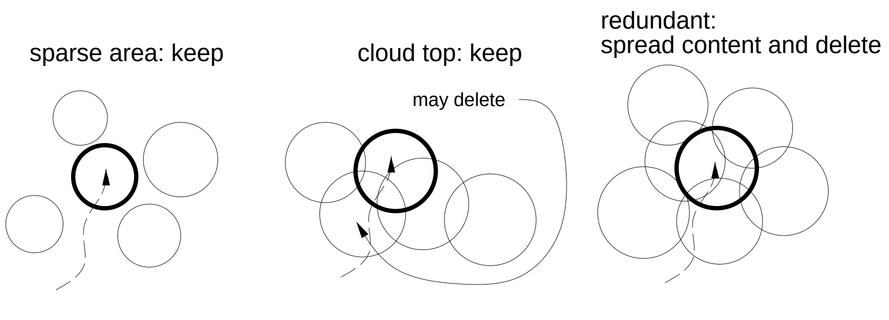

Figure 5: Destiny of the bubble after its move.

In Figure 6 we show the time evolution of the external envelope of bubbles, as visualized by placing a point at the center of each main vortex on a bubble surface. The purpose and use of vortices is described more thoroughly in the next subsection.

Figure 6: Growing 3D cumulus cloud.

3.3 Small scale shape

The surface of the cloud is materialized by structures smaller than the bubbles themselves, i.e. *waves* that become *main vortices* (see D on Figure 2), and subvortices. We assume a recursive structure, as illustrated in Figure 7. A bubble is considered as a sphere, onto which are convected the main vortices, which were initially waves having about 1/10 of the sphere radius [5, 6]. These vortices are also considered to be spherical, and subvortices are themselves advected upon their parent vortex surface in a recursive fashion.

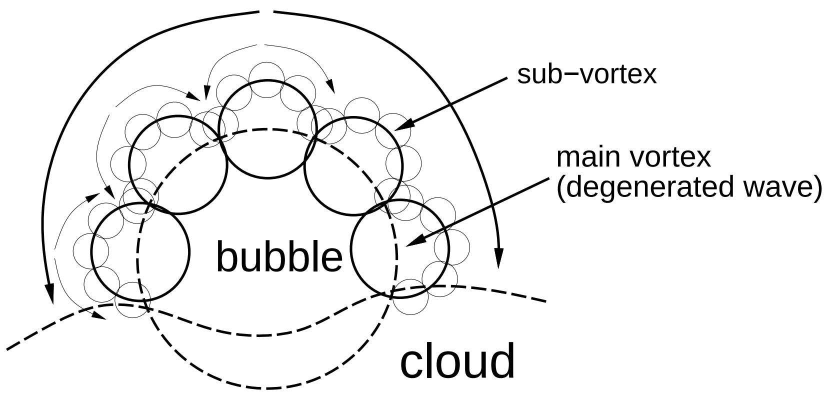

Figure 7: Bubble substructures: main vortices and sub-vortices, each being advected on its parent surface.

In our model, when a bubble reaches and crosses the cloud's top surface, it pushes back this surface, thus 'stealing' the matter (i.e. substructures) from the bubbles residing in the occupied place. These substructures are thus detached from their bubbles and re-attached to the rising one, and advected on its surface as shown in Figure 8. New matter (i.e. waves) appears on the top (in bold in the figure).

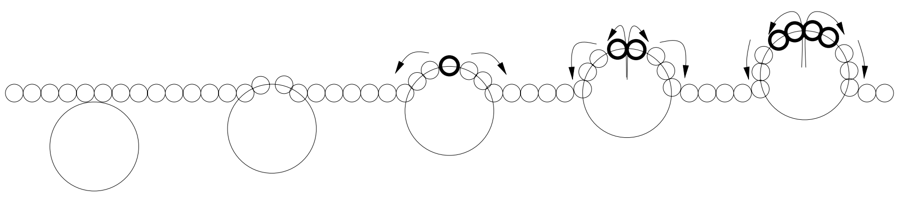

Figure 8: Matter advection.

The matter that appears on the top is advected to the sides where it is then stationary with respect to the background, while the bubble is rising. The rotation around the bubble compensates for the bubble rising, so that the advection acts much like a vertical tank track. If a single bubble rises as a turret, the matter is dropped when it reaches the largest location of the bubble, thus forming a cylindrical column. This is shown in Figure 9.

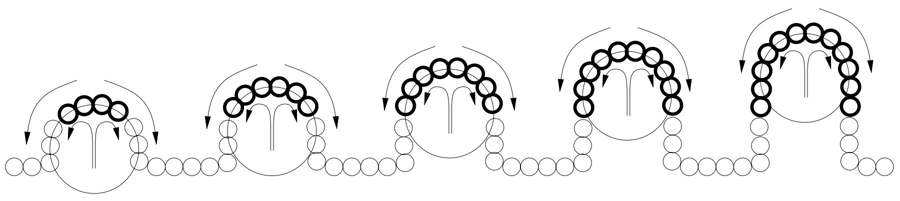

Figure 9: Turret formation.

This structure is also suitable for the representation of billowing smoke, where each individual bubble remains visible as it rises, because the dust replaces the water, and because the smoke column is thin and sparse oppositely to a chubby cumulus. For billowing smoke, the motions are more violent in the vortices at the surface of the bubble, and we consider the vorticity acquired by the small structures, that makes them roll. This remains the subject of future work, however.

In Figure 10, we show an OpenGL rendering of a single bubble with its substructures. These are frames taken from two animations of a single smoke bubble. Shadows on the left image are computed using a shadowmap. The spheres on the right image are textured.

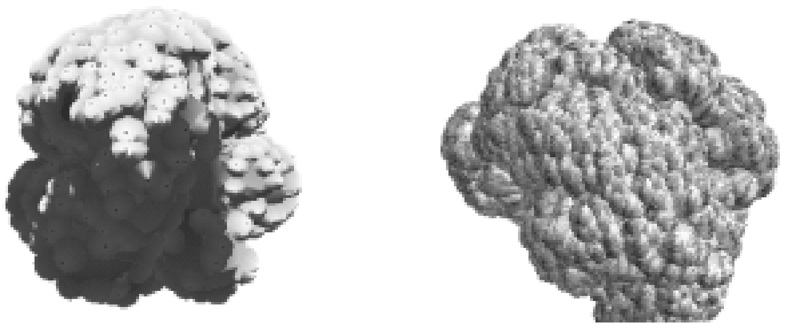
Figure 10: Two bubbles and their substructures rolling on their surfaces.

4 Conclusion

The ongoing work we have presented in this paper proposes to qualitatively simulate the growth and animation of convective clouds. This is achieved by simulating and combining atmospheric structures of various scales. Bubbles are the largest scale structure we use and are responsible for displacements of 3D air parcels in our model. We combine several models to deal with bubbles birth near the ground, bubbles rising up to a cloud, and bubbles moving inside a cloud. Smaller scales are used to add visual complexity for motion and shape on the cloud surface. The smaller-scale substructures are attached to each bubble lying on the cloud periphery, and themselves have substructures. A substructure is advected on the surface of its parent substructure. The largest level of substructure corresponds to waves on a bubble surface that degenerate into main vortices, while finer substructures correspond to the breaking of these vortices into smaller vortices. The general scheme is illustrated in Figure 2.

Our model and implementation are still at an early stage. The various scales have been implemented individually; we have yet to combine them in a single simulation. The following issues also need to be addressed. There are no special provisions for the initial birth and death of a cloud, while the assumption that a cloud is a surface at these instants is not very realistic. The Benard cell structure is not specifically encoded, so there is no guarantee that clouds stay distant from each other. The bubble motion inside the cloud is effected one bubble at a time, while several bubble motion events should be treated simultaneously. Lastly, we have specified the animation of the cloud, but we do not specifically deal with the rendering. The simplest approach to rendering consists of drawing the smaller spherical substructures, possibly with textures. The addition of a global 'skin' upon these structures may also improve the visual aspect of the cloud surface.

Two extensions arise at this point. It would be interesting to deal with larger scale cloud structures, in order to simulate the shape and the evolution of the sky in outdoor scenes. Another issue is that most of the model can also apply to billowing smoke, where the dynamics of advection is more intense, and were substructures play a more important role.

[3] **Acknowledgments**

Specials thanks are due to Michiel Van de Panne for his fruitful discussions, his help with 3D interfaces, and his patient rereading of the paper.
We gratefully acknowledge the financial support of our research by ITRC (Ontario) and NSERC (Canada).

References

1. Emanuel and K. A. *Atmospheric Convection*. Oxford University Press, 1994.
2. Manuel Noronha Gamito, Pedro Faria Lopes, and Mário Rui Gomes. Two-dimensional simulation of gaseous phenomena using vortex particles. In Dimitri Terzopoulos and Daniel Thalmann, editors, *Computer Animation and Simulation '95*, pages 2–15. Eurographics, Springer-Verlag, September 1995. ISBN 3-211-82738-2.
3. Geoffrey Y. Gardner. Simulation of natural scenes using textured quadric surfaces. In Hank Christiansen, editor, *Computer Graphics (SIGGRAPH '84 Proceedings)*, volume 18, pages 11–20, July 1984.
4. Geoffrey Y. Gardner. Visual simulation of clouds. In B. A. Barsky, editor, *Computer Graphics (SIGGRAPH '85 Proceedings)*, volume 19, pages 297–303, July 1985.
5. W. W. Grabowski and T. L. Clark. Cloud-environment interface instability: Rising thermal calculations in two spatial dimensions. *Journal of the Atmospheric Sciences*, 48:527–546, 1991.
6. W. W. Grabowski and T. L. Clark. Cloud-environment interface instability, part II: Extension to three spatial dimensions. *Journal of the Atmospheric Sciences*, 50:555–573, 1993.
7. W. W. Grabowski and T. L. Clark. Cloud-environment interface instability, part III: Direct influence of environmental shear. *Journal of the Atmospheric Sciences*, 50:3821–3828, 1993.
8. Robert A. Houze Jr. *Cloud Dynamics*. Academic Press.
9. James T. Kajiya and Brian P. Von Herzen. Ray tracing volume densities. In Hank Christiansen, editor, *Computer Graphics (SIGGRAPH '84 Proceedings)*, volume 18, pages 165–174, July 1984.
10. A. Luciani, A. Habibi, A. Vapillon, and Y. Duroc. A physical model of turbulent fluids. In Dimitri Terzopoulos and Daniel Thalmann, editors, *Computer Animation and Simulation '95*, pages 16–29. Eurographics, Springer-Verlag, September 1995. ISBN 3-211-82738-2.
11. Jean Orloff. *personal communications*.
12. Jos Stam and Eugene Fiume. Turbulent wind fields for gaseous phenomena. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 369–376, August 1993.
13. Jos Stam and Eugene Fiume. Depicting fire and other gaseous phenomena using diffusion processes. In Robert Cook, editor, *SIGGRAPH 95 Conference Proceedings*, Annual Conference Series, pages 129–136. ACM SIGGRAPH, Addison Wesley, August 1995. held in Los Angeles, California, 06-11 August 1995.
14. Y. Takaya. The motion of uneven structure of convective clouds. *Journal of the Atmospheric Sciences*, 50:574–587, 1993.
15. Wallace, J. M., and P. V. Hobbs. *Atmospheric Science: An Introductory Survey*. Academic Press, 1977.