Realistic and Controllable Fire Simulation

Philippe Beaudoin Sebastien Paquet Pierre Poulin

Target:

Use a set of techniques together to produce realistic-looking animations of burning objects.

Techniques:

i>A method for simulating spreading on polygonal meshes.

ii>Use individual flames as primitives to animate and render the fire.

(Flames, essentially are deformable chains of vertices rooted on the surface)

Advantages:

i>Simple, enabling rapid computation; giving more intuitive control over the simulation without compromising realism.

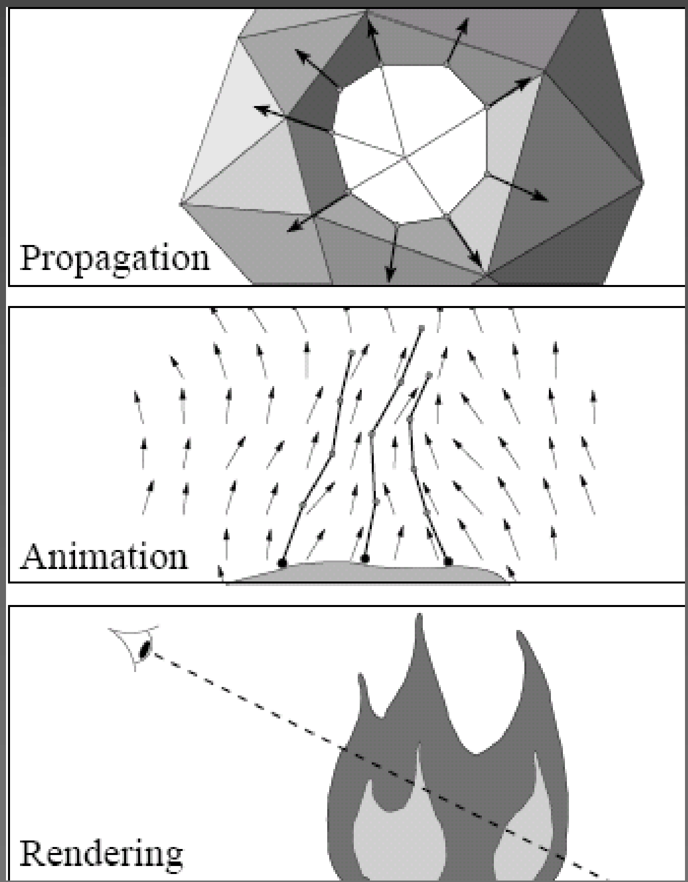
ii>Useful, scaling well, it's possible to animating phenomena from "simple candle-like flames to complex widespread fires".

iii>Of course could produce convincing visual behavior.

Realistic and Controllable Fire Simulation

Introduction

Simulating fire could be divided into three subproblems.



a.Fire propagation

b.Flame animation

c.Flame rendering

Realistic and Controllable Fire Simulation

Fire Propagation

Previous work: Modeling fire propagation by numerically solving the differential equations that govern the evolution of temperature, pressure, and velocity of the air surrounding the burning object. *(Chiba et al[1] and Takahashi et al[2])*

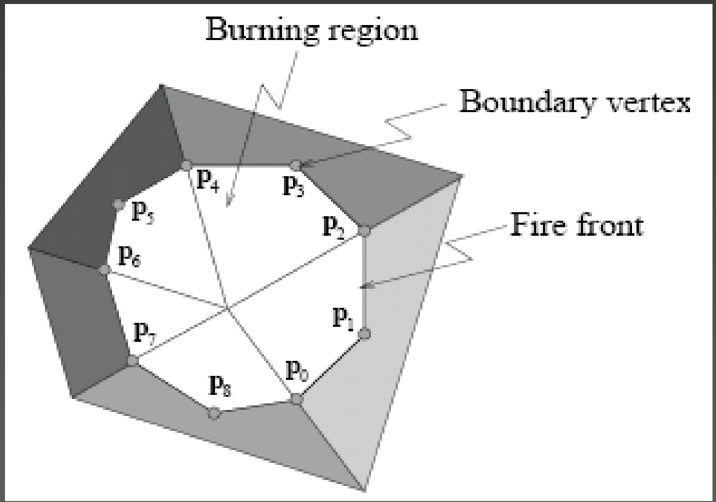
But huge computation and its complexity

Here, by observing the visual feature in fire propagation: the growth of the burning zone. Then find the driven parameters: **fuel density**, **oxygen supply**, **wind**, and **surface orientation relative to gravity**.

Method is based on one presented by *Perry and Picard[3]*

Realistic and Controllable Fire Simulation

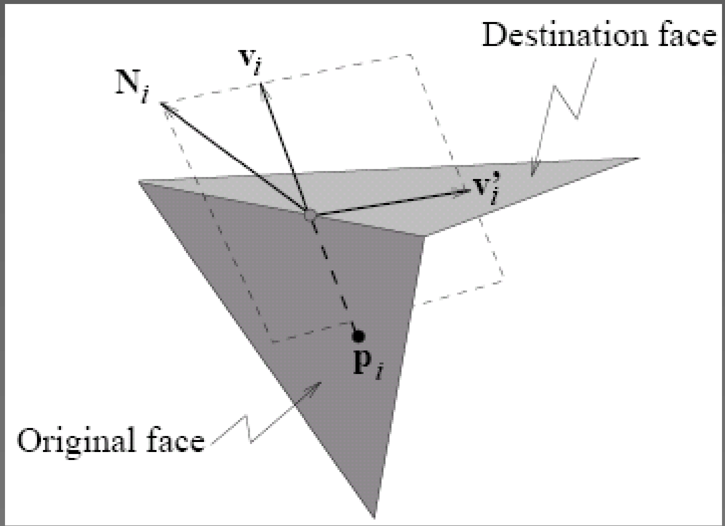
Fire Propagation



a. Boundaries representation

A boundary is represented by a closed curve on the surface of the object.

If two consecutive vertices are on different faces, then one of these vertices lies on an edge shared by those faces.



b. Displacing vertices

Denoting its position by pi and its velocity vi:

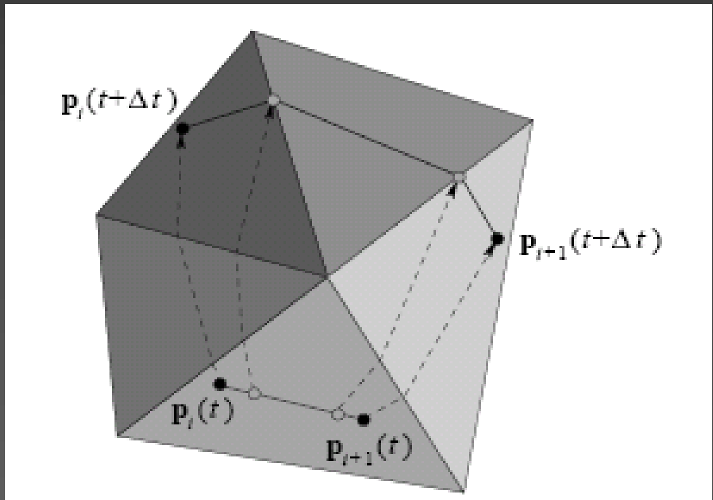
$$\mathbf{p}_i(t+\Delta t) = \mathbf{p}_i(t) + \mathbf{v}_i(t)\Delta t.$$

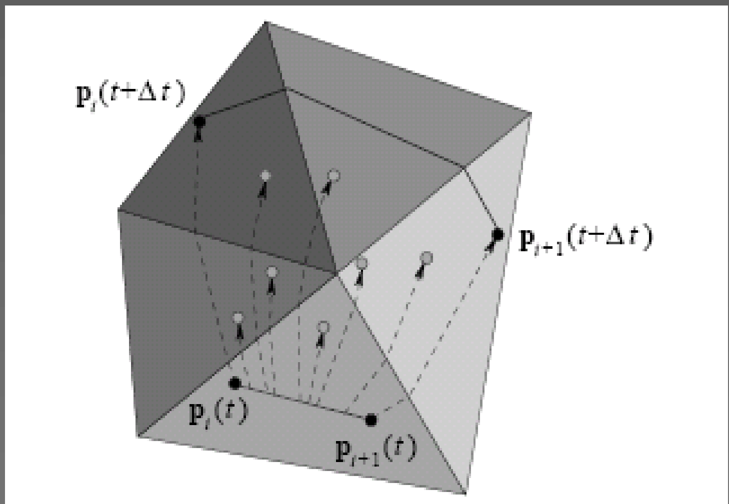
A vertex may leave a face by crossing an edge, now we have:

$$\mathbf{v}_i' = \eta(\mathrm{N}_i \times \mathbf{v}_i) \times \mathrm{N}'.$$

Realistic and Controllable Fire Simulation

Fire Propagation





c. Evolving the front

The boundary must expand with each time step.

i>Let each vertex of the boundary move according to its velocity vector, update it when neccesary.

ii>Adding new vertices if neccesary.

d. Nonuniform propagation speeds

The evolution of a burning surface is driven by locally defined parameters.

Only allow modification to the magnitude of these velocities in order to avoid shrinking.

e. Generating points on the surface

Used to plant the flames

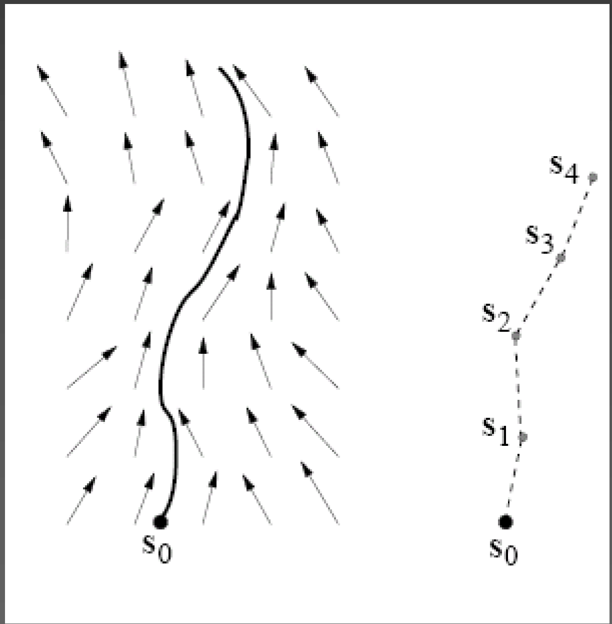
Realistic and Controllable Fire Simulation

Flame Genesis and Animation

- Picture a flame as a stream of incandescent gas which follows the air flow surrounding it.
- This stream is modeled as a chain of connected particles, which is **skeleton.**
- First particle is the root of the flame, attached to a point on the burning surface; the rest of the chain moves according to a turbulent, time-varying vector field which accounts for the dynamic behavior of the fire.
- Vector field is given by the user.

Realistic and Controllable Fire Simulation

Flame Genesis and Animation



a. Planting flames on the surface

The density of points generated on the surface can be adjusted to capture various effects visible in fire.

b. Defining the air velocity field

$$\mathbf{V}(\mathrm{x}, t) = \sum_{i=0}^{n} b_i(\mathrm{x}, t)\, \mathbf{V}_i(\mathrm{x}, t)\ .$$

c. Defining the flame skeleton

$$\mathrm{s}_i = \mathrm{s}_{i-1} + \frac{l(t)}{n} \mathbf{V}(\mathrm{s}_{i-1})\ .$$

d. Growing and shrinking the flames

Flames are assigned a life duration; use a clamped quadratic function of time starting and ending at zero. Peak length.

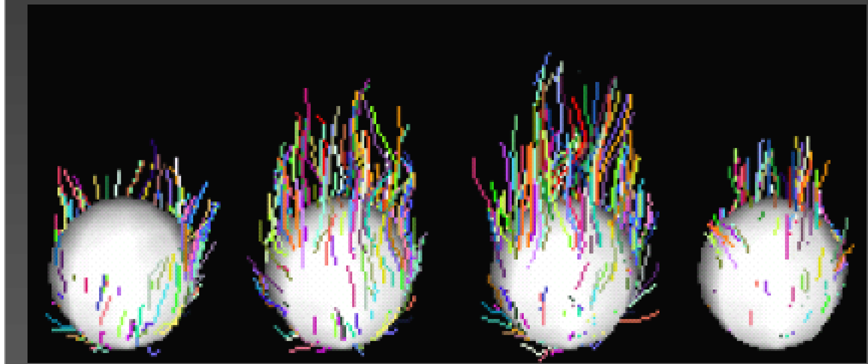
e. Detached flames

Highly turbulent fires often feature flames that take off from the surface and drift for a while before cooling down and vanishing.

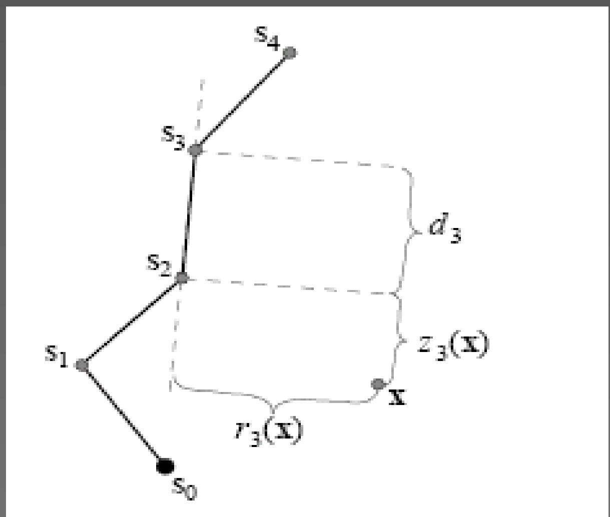
$$\mathrm{s}_0(t + \Delta t) = \mathrm{s}_0(t) + \Delta t\, \mathbf{V}(\mathrm{s}_0(t), t)\ .$$

Realistic and Controllable Fire Simulation

Flame Rendering



a.Modeling flames using implicit surfaces provides a convenient way to emulate the behavior: flames that close enough smoothly blend together while distant flames remain separate.

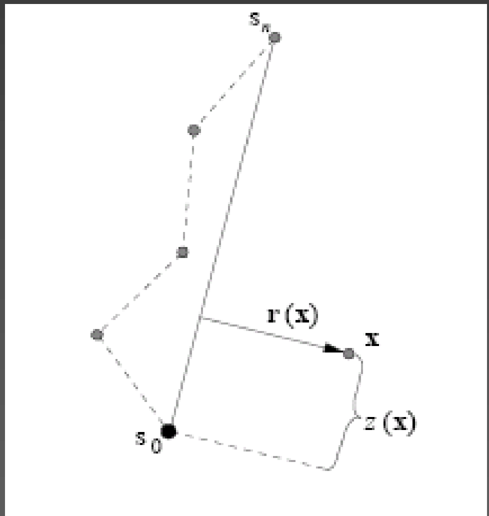


b. For a single flame, using the physical analogy of the electrical potential induced by a uniformly charged rod.

$$
\begin{aligned}
E_i(\mathbf{x}) &= \int_{p=0}^{d_i} \frac{1}{\sqrt{(p - z_i(\mathbf{x}))^2 + r_i(\mathbf{x})^2}} dp \\
&= \sinh^{-1}\left(\frac{z_i(\mathbf{x})}{r_i(\mathbf{x})}\right) - \sinh^{-1}\left(\frac{z_i(\mathbf{x}) - d_i}{r_i(\mathbf{x})}\right)
\end{aligned}
$$

Realistic and Controllable Fire Simulation

Flame Rendering



$$I_s(\mathrm{x}) = F\Big(E(\mathrm{x}'(\mathrm{x}))\Big) .$$

$$I(\mathrm{x}) = \sum_s I_s(\mathrm{x}) .$$

c. Isosurface obtained from previous one does not distinguish between the root and the top of the flame.

$$\mathrm{x}'(\mathrm{x}) = \mathrm{x} + \exp\left(\frac{2z(\mathrm{x})}{d} - 1\right)\mathbf{r}(\mathrm{x}) ,$$

d. E(x) falls off rather slowly with distance from the skeleton.

$$F(E) = \frac{v\,(e^E - 1)}{e^v - 1} .$$

Amplify the falloff of E for values smaller than the user-defined isovalue v.

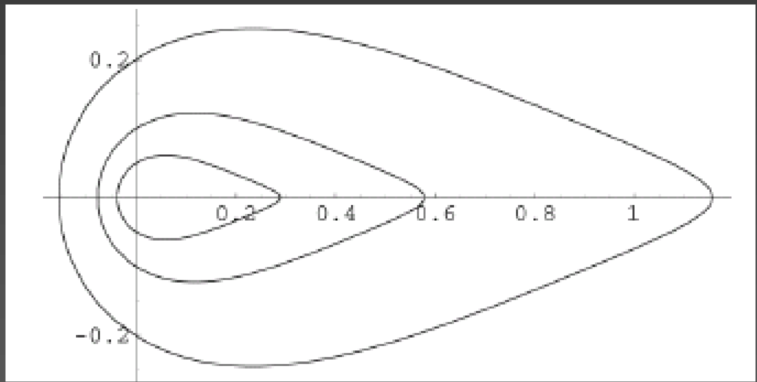
Also has the advantage of reducing the bounding volume for a flame.

Functions expresses the final contribution of skeleton s to the implicit function at point x

And all skeletons to the point x

Realistic and Controllable Fire Simulation

Flame Rendering



e.For a single flame, various colors appear in successive layers, each having a shape similar to the flame outline.

Temperature mainly depends on distance from the base of the flame.

When rendering, assign different colors to the layers in order to obtain a variety of fire effects.

f. Given a user-defined iso-value v, use the marching cubes technique to obtain the closed surface satisfying I(x) = v.

However, time consuming; limit the contribution of each skeleton to its neighborhood. Start from skeleton, stop when I is negligible.

g. The color and intensity of light reaching the eye following a given path is a function of the path lengths inside each layer.

Realistic and Controllable Fire Simulation

Pros and Cons

Pros:

a. It produces more convincing images of isolated flames as well as burning objects.

b. The relative simplicity of this model results in reduced time and memory requirements compared to other approaches.

c. These simplificatoins make the simulation more controllable, and it is easier and takes less time to obtain desired effects.

Cons:

a. Fire cannot reach objects that are disconnected from a burning object.

b. Many parameters are empirically set, exceptional results may happen.

Realistic and Controllable Fire Simulation

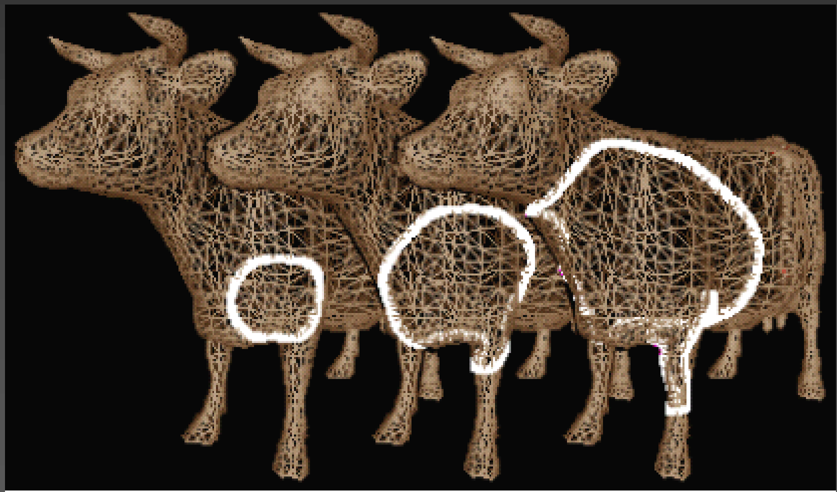


Figure 9: Front propagation.

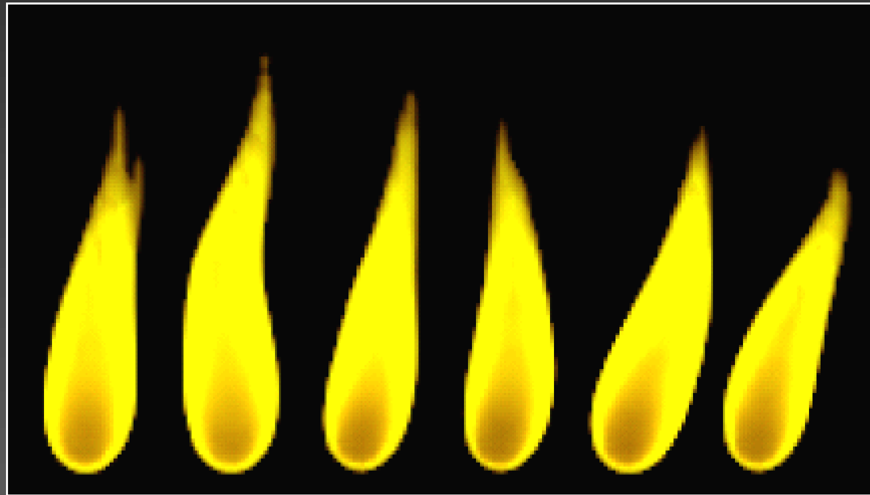


Figure 10: Animated candle flame.

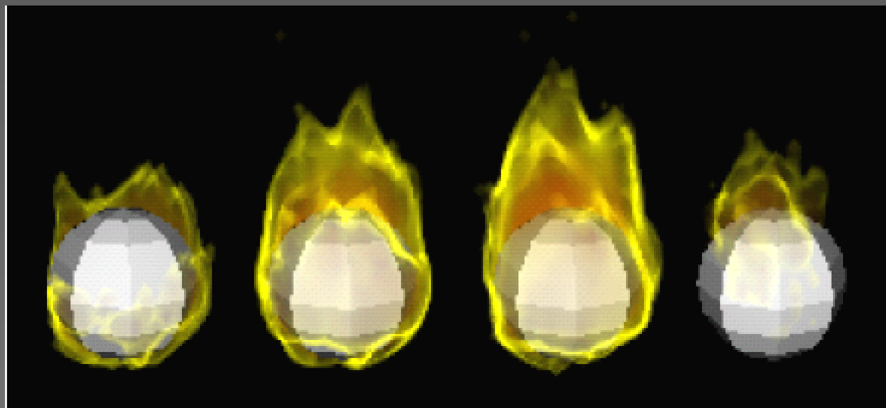


Figure 12: Burning sphere.

Voxel on Fire

Ye Zhao   Xiaoming Wei   Zhe Fan   Arie Kaufman   Hong Qin

Target:

Animate the fire propagation and the burning consumption of objects represented as volumetric data sets.

Method:

Use a volumetric fire propagation model based on an enhanced distance field.

The object voxels and the splats associated with the flame particles are rendered in the same pipeline so that the volume data with its external and internal structures can be displayed along with the fire.

Voxel on Fire

Introduction

Figure 1: Fire on a volumetric table. The underlying noncombustible metal frame is revealed once the wooden outer layer is consumed.

a. Splatting is used to render the flames resulting in a realistic visual effect. (*King et al[3]*)
b. Lattice Boltzmann Model(LBM) was introduced to model the interaction of the fire with wind, to generate the external air velocity field that affects the movement of the fire front.
c. Difference with the previous one, that propagation approach focused on simulating the fire front on surface triangles or polygons. Here the method is used on volumetric data sets.

Voxel on Fire

Distance Field

a. A distance field is a scalar field that specifies a distance to a shpe.

$$D(p) = sgn(p) \cdot min\{|p-q| : q \in S\}$$

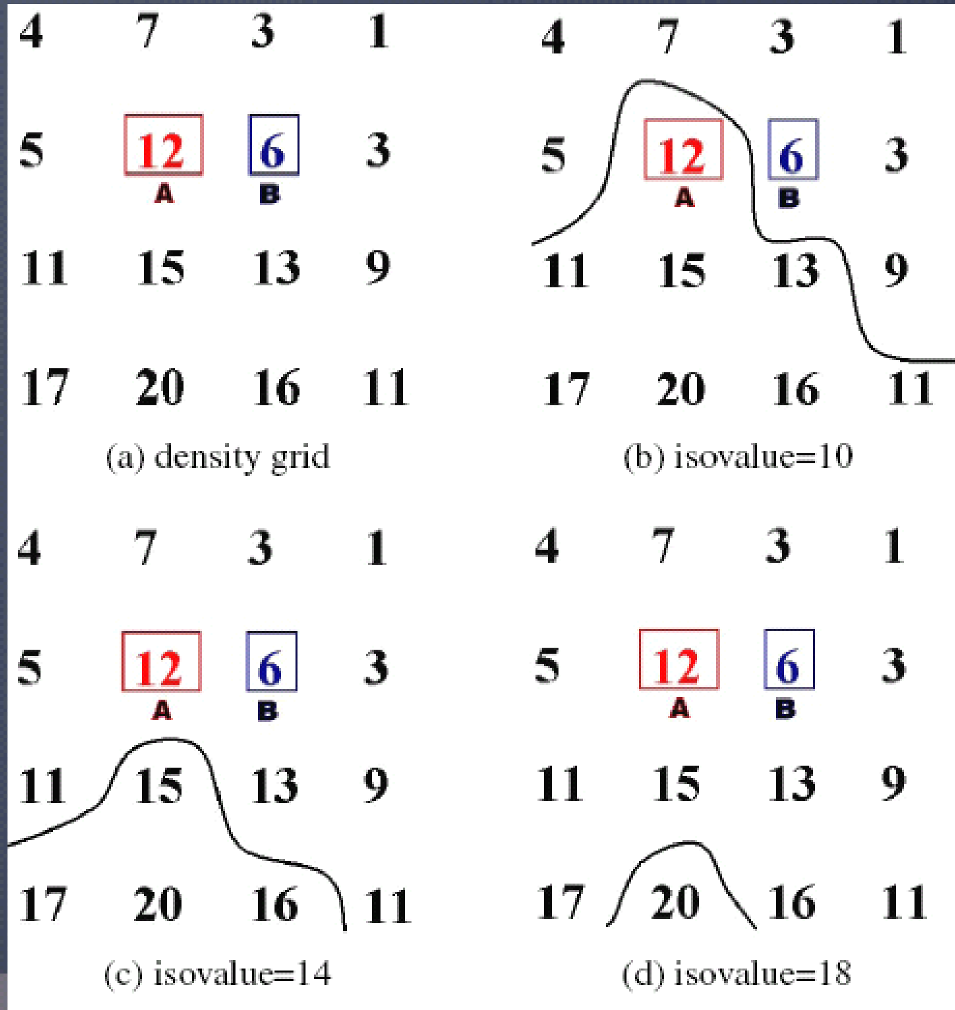
where sgn(p) = 1(-1) if p is inside(outside) of S.
b. In each step, fire front points move along a tangent direction of the object surface defined by external forces.
c. The method createing the distance field is based on one presented by *Breen et al[4]*
d. Shell Volume
i> Shell Volume is used to generated the distance field.
ii>The basic idea of using the shell volume: for given isovalue we can quickly determine whether a voxel of V is inside, outside or on the isosurface.
iii>Save three ranges
(min0, max0): min0 = min(dc, dm) and max0 = max(dc, dm).
(min1, max1): min1, defined as the smallest min0 of its lower density neighbors.
(min-1, max-1):min-1 is defined as the smallest min0 of its higher density neighbors.
for each voxel by comparing the density values in its 26 neighborhood.
ensure: max1 <= min0 and max0 <= min-1

Voxel on Fire

Distance Field

e. Comparing the isovalue with ranges.
INB; ONB; SNB.
f. Distance Field Generation.
INB, ONB: find the closest point in SNB
For A and B, decide the ranges
(min0, max0);
(min1, max1);
(min-1, max-1);

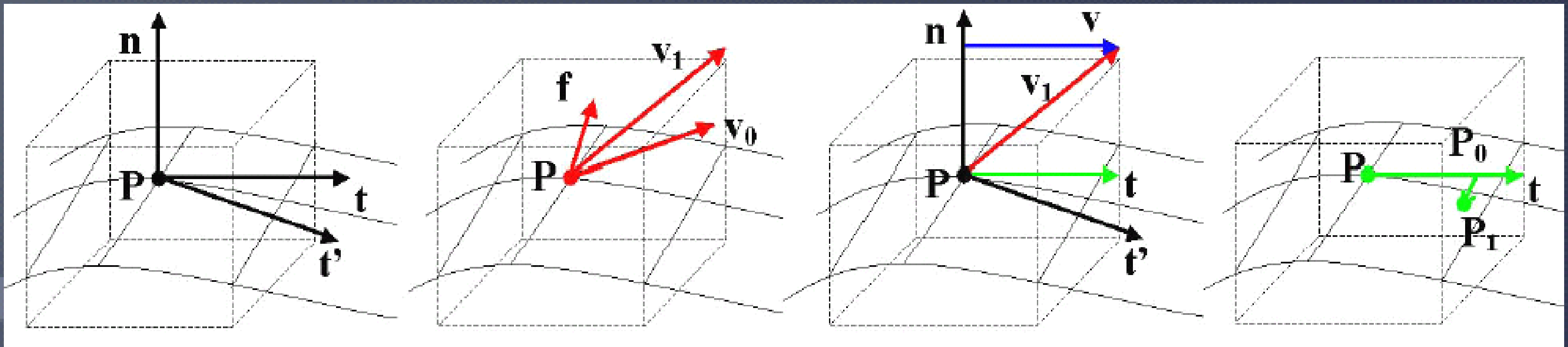
In figure d, the distance fields of A and B are calculated from propagation by the fast marching method[4].



Voxel on Fire

Fire Propagation

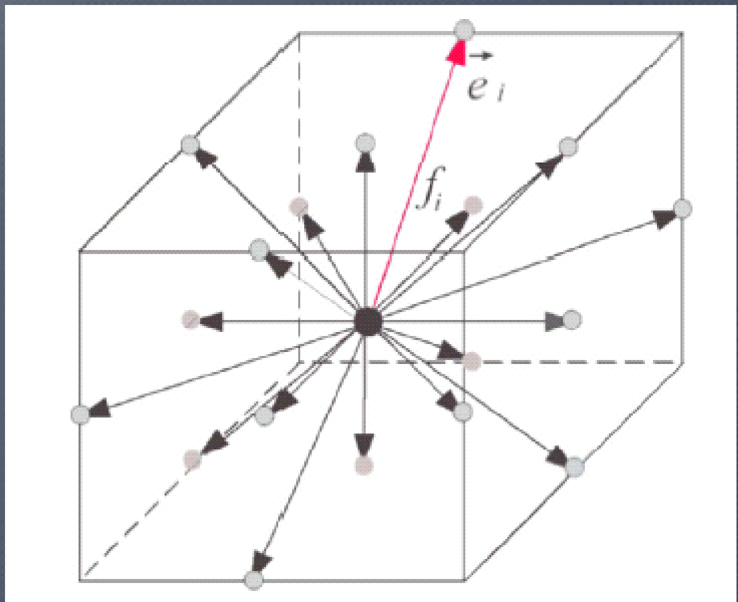
a. Fire front is represented by an evolving group of front points on a virtual isosurface.
b. Here model the fire propagation without generating an isosurface and no geometric entities of the surface are used directly.
c. Algorithm (every point; in each time step):
  i>Find the tangent plane of the virtual surface at the current position.
  ii>Calculate the forward velocity from current velocity and field velocity computed by LBM.
  iii>Adjust the forward velocity within the tangent plane.
  iv>Define the next position as the closest point on the virtual surface to the temporary target position using the distance field information.



Voxel on Fire

Fire Flames

a. Fire front emits particles into the air space to form the flames. These particles move according to the wind velocity field surrounding the volume object.
b. The LBM is a numerical scheme for simulating viscous fluids using a regular lattice of cells and links.
c. The packet distributions are denoted as fi where i is a particular link with its velocity vector shown as ei. The macroscopic density and velocity u:



$$\rho = \sum_i f_i \qquad \mathbf{u} = \frac{1}{\rho}\sum_i f_i \mathbf{e}_i$$

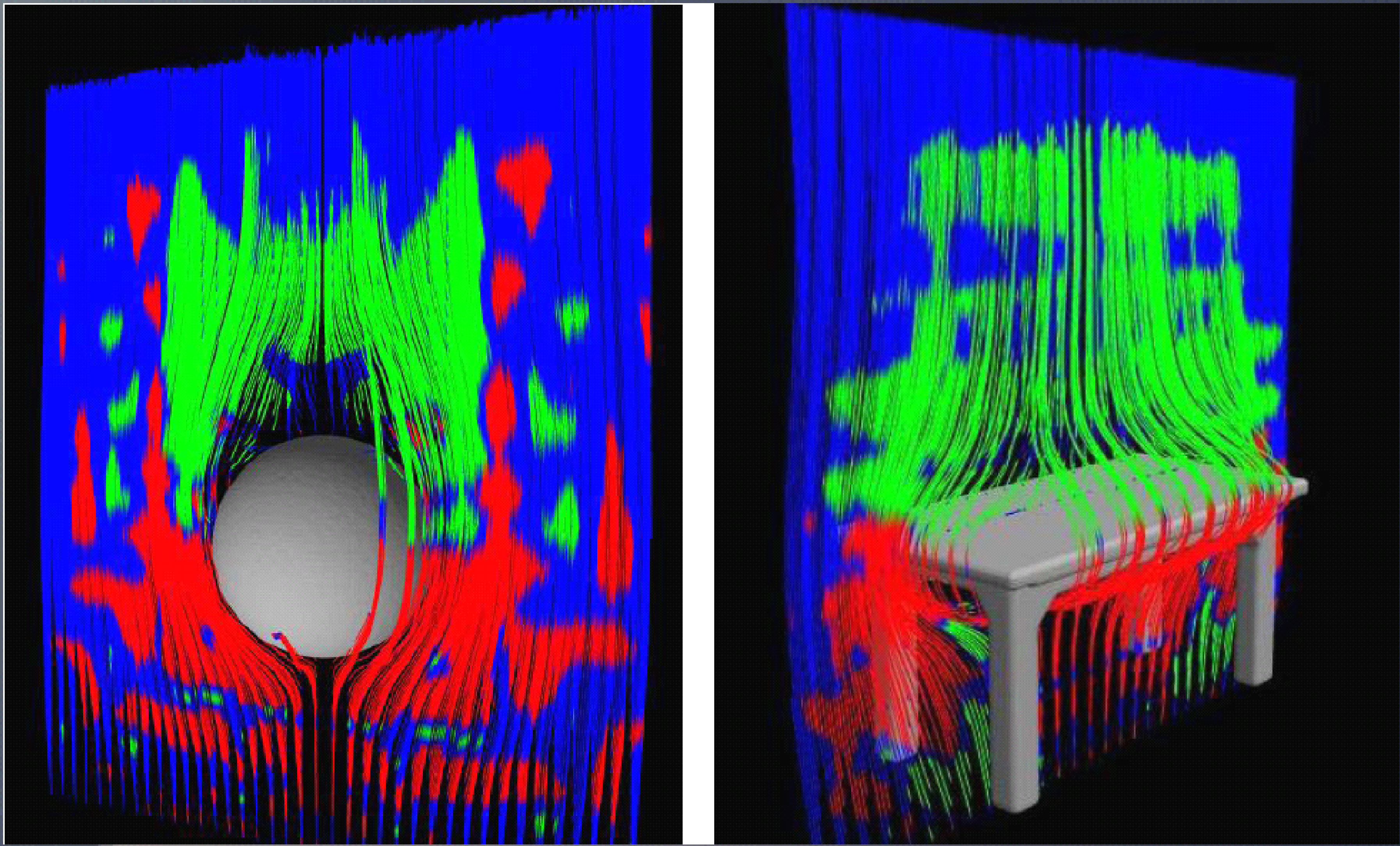
At each time step, every cell updates its packet distribution values based on colision and streaming rules.

$$collision: f_i^{new}(\mathbf{x},t) - f_i(\mathbf{x},t) = \Omega_i$$
$$streaming: f_i(\mathbf{x}+\mathbf{e}_i, t+1) = f_i^{new}(\mathbf{x},t)$$

The surfaces of the burning objects are resampled accurately as the intersection points on the links between fluid nodes and solid nodes.

Voxel on Fire

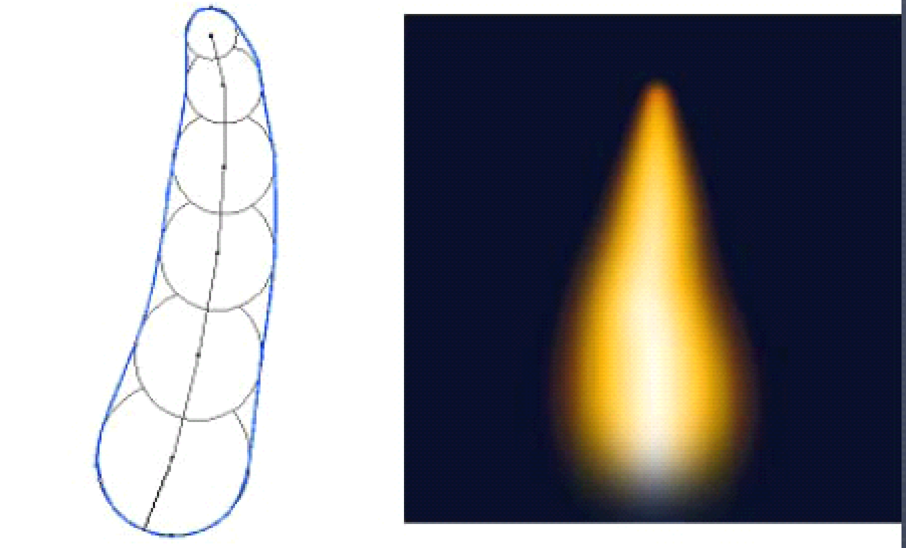
Fire Flames



Voxel on Fire

Flame Generation and rendering

a. In each simulation step, flame particles are emitted from the fire front points with an initial velocity defined by the current wind velocity and the fuel of its emitter.
b. Flame particles have a finite life span determined by the fuel value of that part of the object from which they are emerging.
c. This method works on the volume data, so could use volume rendering directly.



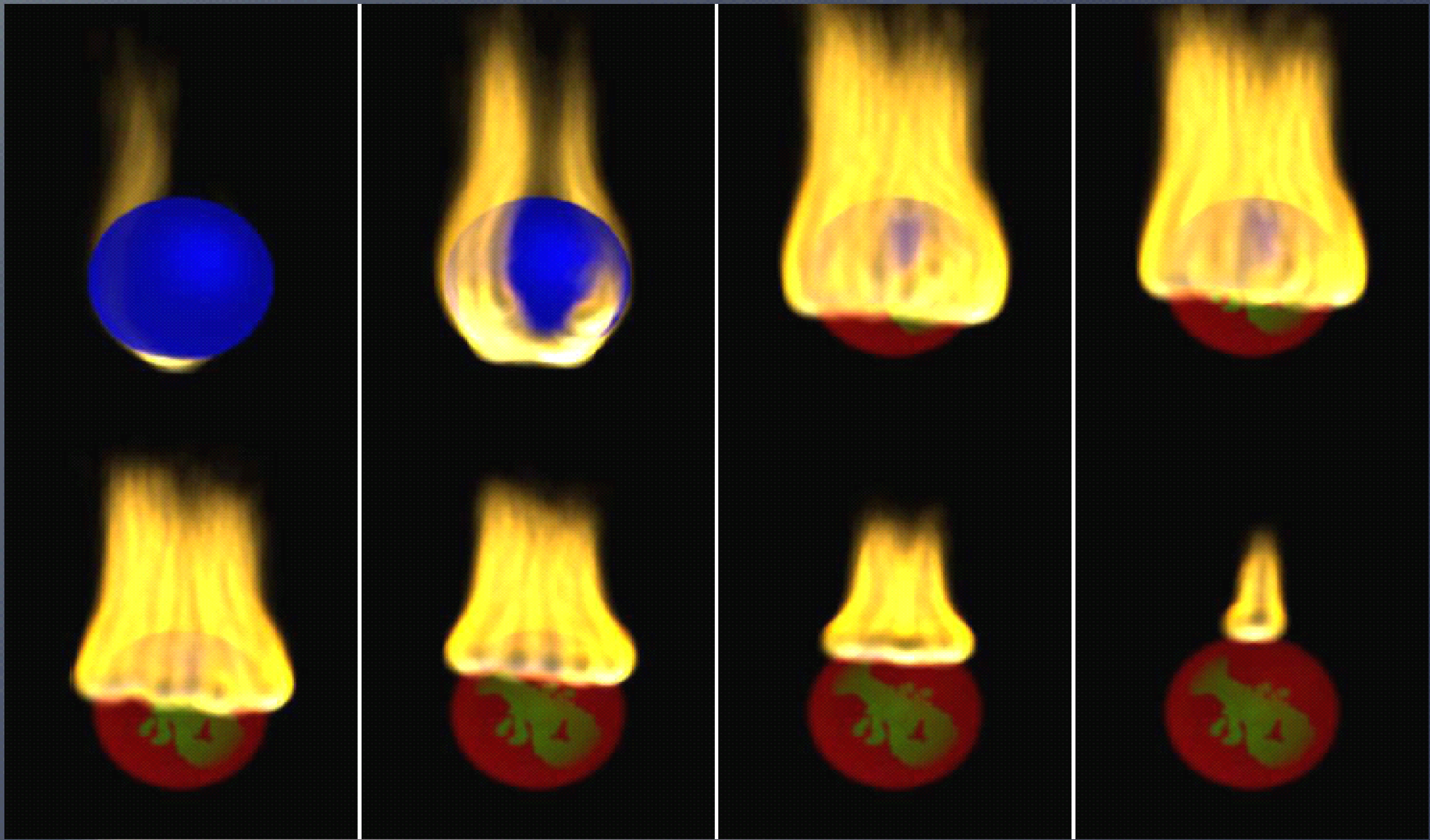
(a) (b)

d. Use splatting (Westover[5]) to render the voxels and flame particles together.
e. Assign different colors to different parts of the splats according to their distance from the center to produce a single flame in colored layers.

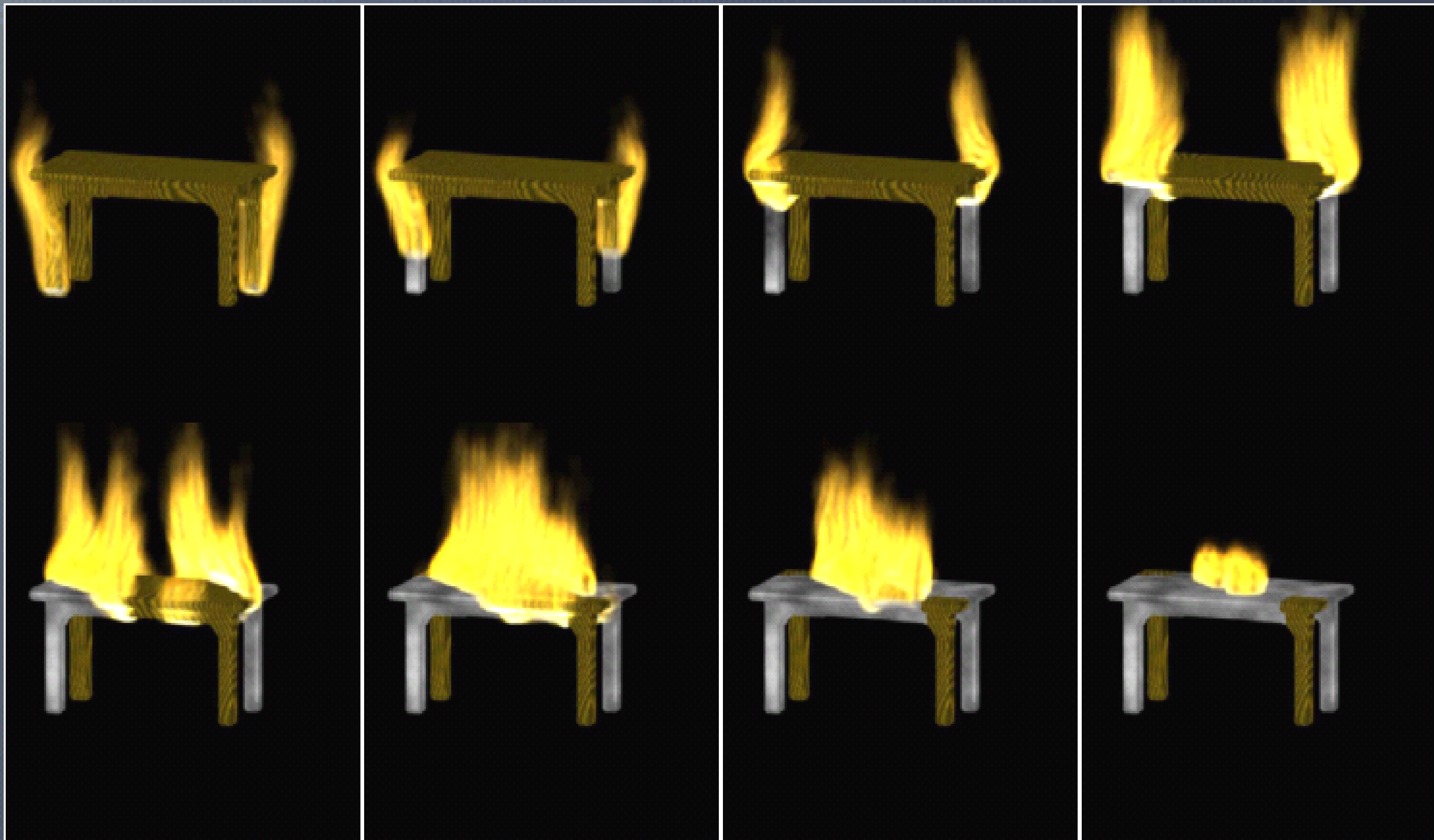
Voxel on Fire

Conclusion

Key: Distance field representation; shell volume; LBM; Splatting rendering method



Voxel on Fire



Stable Fluids

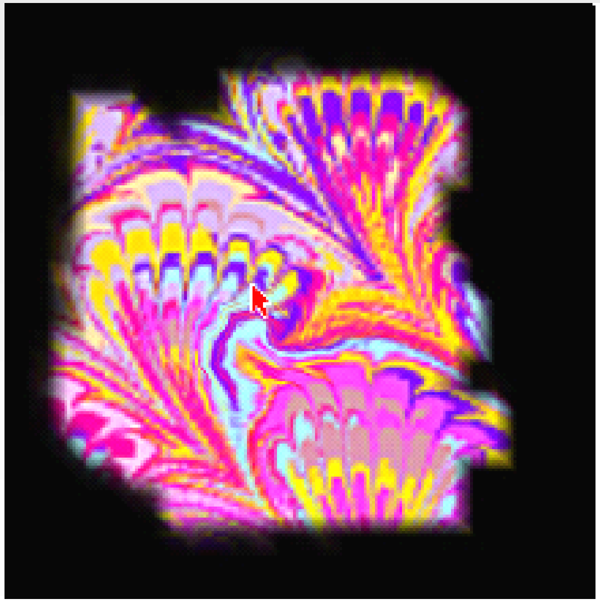
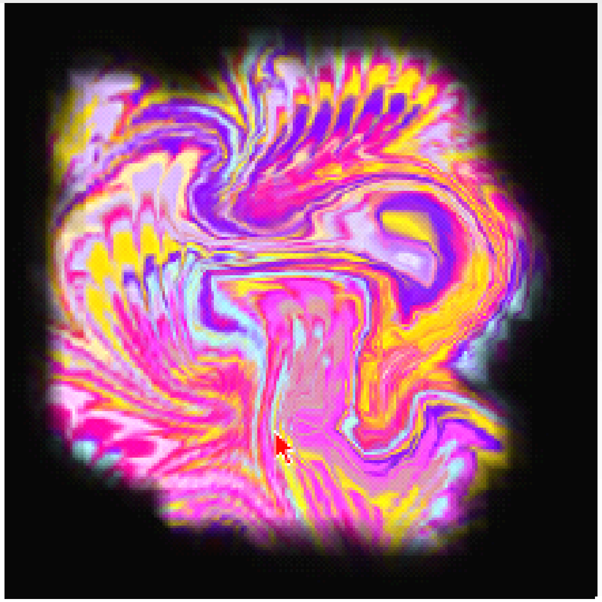
Jos Stam

- It propose an unconditionally stable model which still produces complex fluid-like flows, meanwhile it is very easy to implement.
- Navier-Stokes equations are a very good model for fluid flow, so the problem is to compute these equations.
- Previous work (*Foster and Metaxas[6]*) show the advantages of using the full three dimensional Navier-Stokes equations in creating fluid-like animation, their model is based on a finite differencing of the Navier-Stokes equations and an explicit time solver.
- The problem with explicit solvers is that the numerical scheme can become unstable for large time-steps. And instability leads to "blow-up". ("With the model we have described this(instability) can happen when the velocity of any part of the gas allows it to move further than Δr in a single timestep", Δr is the grid width)
- So author here presented a stable algorithm that solves the full Navier-Stokes equations.

Stable Fluids

Characteristics

- Use both Lagrangian and implicit methods to solve the equation.
- Bad or Good? It suffers from too much "numerical dissipation.
- Apply the solver to update both the flow and the motion of densities within the flow.
- Can be combined with solid textures.

Stable Fluids

Navier-Stokes equations

$$\nabla \cdot \mathbf{u} = 0 \qquad (1)$$

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

- ν: kinematic viscosity of the fluid; ρ: density; f: external force;
- p: pressure field; u: velocity field. (Density and temperature are nearly constant)
- Consider two types of boundary conditions: *periodic* boundary conditions and *fixed* boundary conditions.
- Helmholtz-Hodge Decomposition: $\mathbf{w} = \mathbf{u} + \nabla q,$ w is a vector field, u has zero divergence and q is a scalar field.
- Define an operator P which projects any vector field w onto its divergence free part u = Pw.

$$\mathbf{u} = \mathbf{P}\mathbf{w} = \mathbf{w} - \nabla q. \rightarrow \frac{\partial \mathbf{u}}{\partial t} = \mathbf{P}\left(-(\mathbf{u} \cdot \nabla)\mathbf{u} + \nu \nabla^2 \mathbf{u} + \mathbf{f}\right),$$

Stable Fluids

Method of Solution

Basic equation: $$\frac{\partial \mathbf{u}}{\partial t} = \mathbf{P}\left(-(\mathbf{u}\cdot\nabla)\mathbf{u} + \nu\nabla^2\mathbf{u} + \mathbf{f}\right),$$

a. Start from w0(x) = u(x, t), resolve it over the time span Δt in four steps.

$$\mathbf{w}_0(\mathbf{x}) \overbrace{\longrightarrow}^{\text{add force}} \mathbf{w}_1(\mathbf{x}) \overbrace{\longrightarrow}^{\text{advect}} \mathbf{w}_2(\mathbf{x}) \overbrace{\longrightarrow}^{\text{diffuse}} \mathbf{w}_3(\mathbf{x}) \overbrace{\longrightarrow}^{\text{project}} \mathbf{w}_4(\mathbf{x}).$$

i> First step, it's the easiest one: $\mathbf{w}_1(\mathbf{x}) = \mathbf{w}_0(\mathbf{x}) + \Delta t\ \mathbf{f}(\mathbf{x}, t)$

$$\mathbf{w}_1(\mathbf{x}) = \mathbf{w}_0(\mathbf{x}) + \Delta t\ \mathbf{f}(\mathbf{x}, t)$$

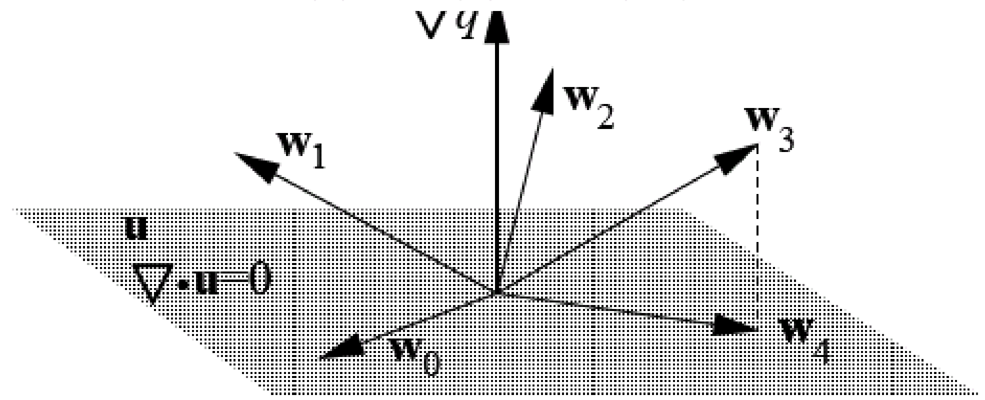


Figure 1: One simulation step of our solver is composed of steps. The first three steps may take the field out of the space of divergent free fields. The last projection step ensures that the field is divergent free after the entire simulation step.

Stable Fluids

Method of Solution

ii>Second step: advection.

This term makes the Navier-Stokes equations non-linear, previous method(*Foster and Metaxas[6]*) resolve this component using finite differencing.

Method stated here is based on a technique to solve PDE known as *method of characteristics.*

$$\mathbf{w}_2(\mathbf{x}) = \mathbf{w}_1(\mathbf{p}(\mathbf{x}, -\Delta t)).$$

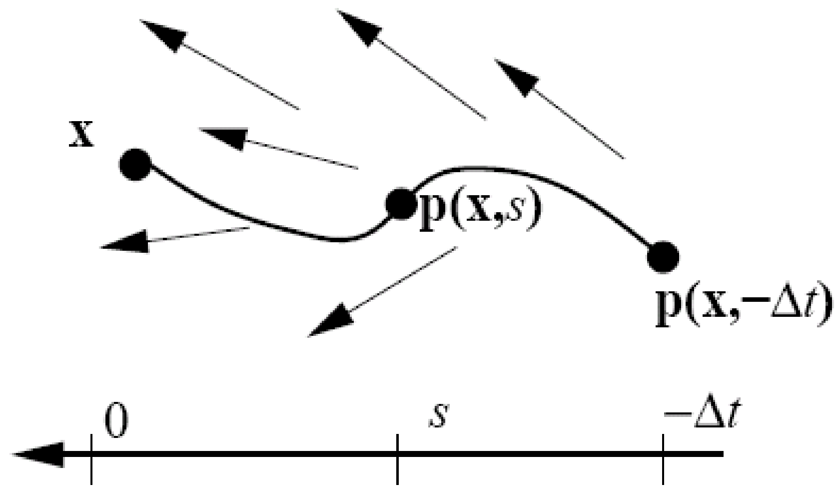


Figure 2: To solve for the advection part, we trace each point of the field backward in time. The new velocity at $\mathbf{x}$ is therefore the velocity that the particle had a time Δt ago at the old location $\mathbf{p}(\mathbf{x}, -\Delta t)$.

Stable Fluids

Method of Solution

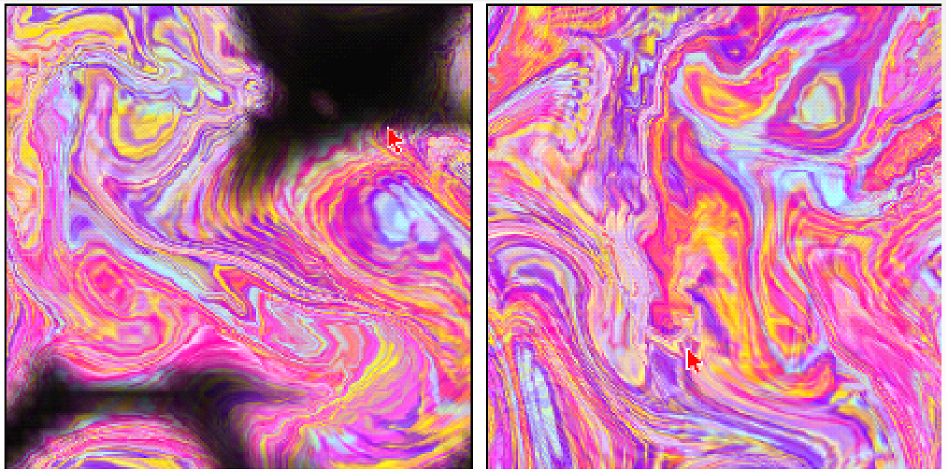
iii>Third step: viscosity.

$$\frac{\partial \mathbf{w}_2}{\partial t} = \nu \nabla^2 \mathbf{w}_2. \quad \longrightarrow \quad \left(\mathbf{I} - \nu \Delta t \nabla^2\right) \mathbf{w}_3(\mathbf{x}) = \mathbf{w}_2(\mathbf{x}),$$

iv>Fourth step: projection.

$$\nabla^2 q = \nabla \cdot \mathbf{w}_3 \qquad \mathbf{w}_4 = \mathbf{w}_3 - \nabla q.$$

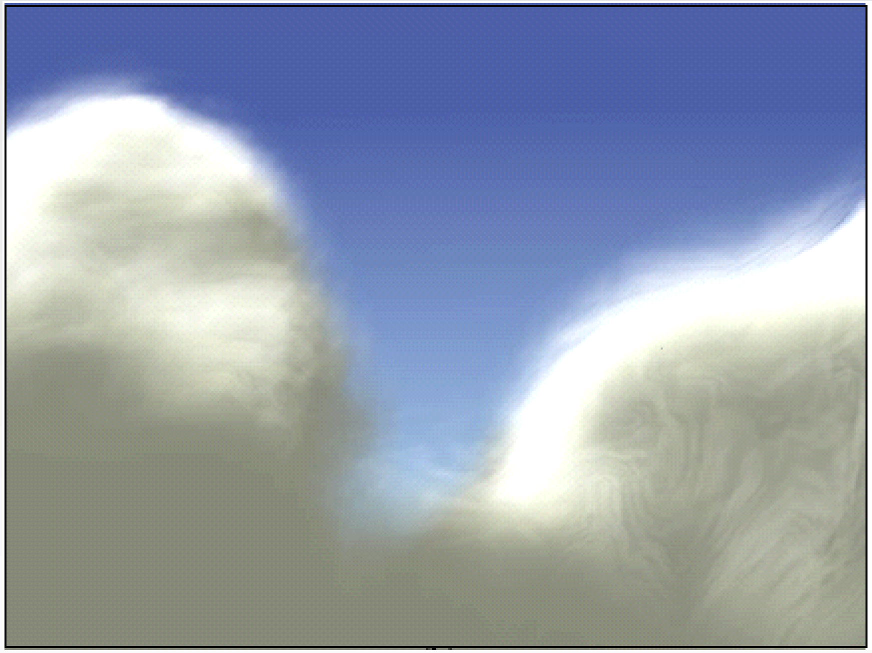
Poisson equation.

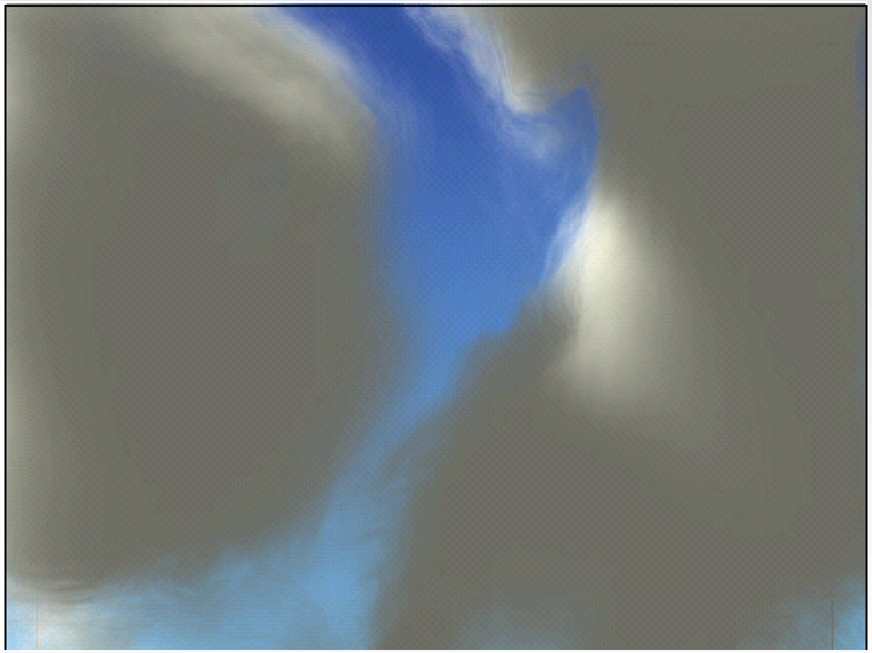


Stable Fluids

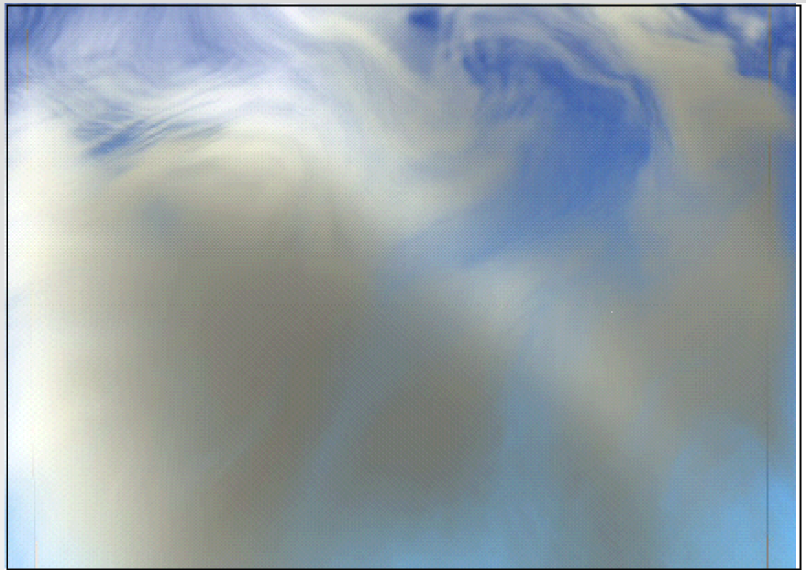
Result

a. We can animate a non-reactive substance which is injected into the fluid using a similar method.

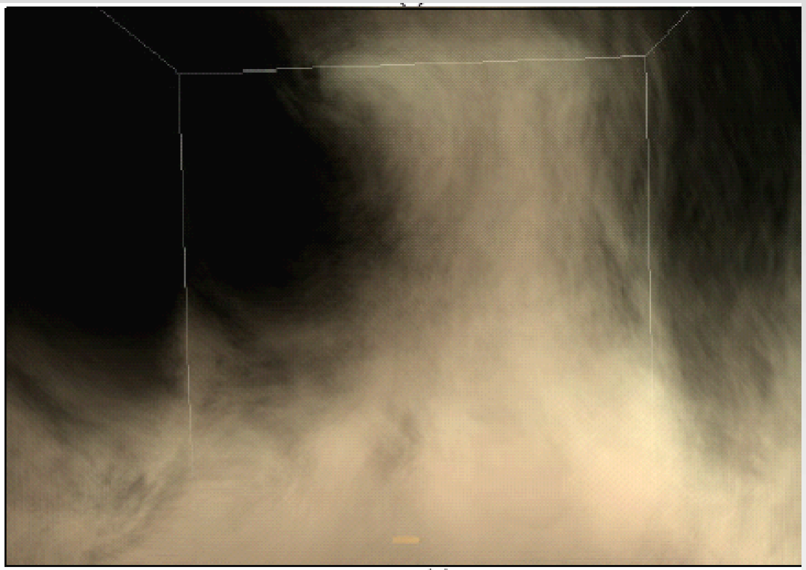




Stable Fluids



(d)



(e)