Visual Simulation of Smoke

Ronald Fedkiw, Jos Stam and Henrik Wann Jensen

Stanford University & Alias|wavefront

Smoke in Computer Graphics

Ideally:

Looks Good + Fast

Non-Physical Models

Early CG models

Texture maps + simple primitives

Too much control…

Physical Models

Natural framework for fluid modeling

Reuse literature

Hard to solve !

Physical Smoke Models in CG

Incompressible

Yaeger'86 Two dimensions
Gamito'95

Foster'97 unstable
Stam'99 stable

Compressible

Yngve'00 explosions

Our New Model

Improve Stam'99 (Stable Fluids)

- Handle moving boundaries
- Reduce numerical dissipation
- Add high quality volume rendering

Method still fast but looks more "smoke-like"

Incompressible Euler Equations

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} + \mathbf{f}$$

self-advection

forces

$$\nabla \cdot \mathbf{u} = 0$$

incompressible

(Navier-Stokes without viscosity)

Additional Equations

smoke's density

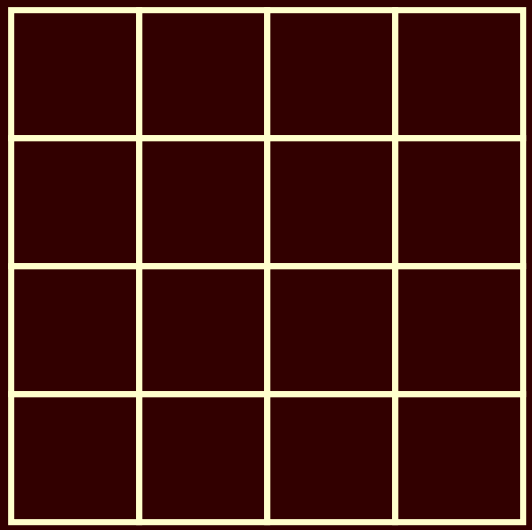
$$\frac{\partial \rho}{\partial t} = -(\mathbf{u} \cdot \nabla)\rho + S$$

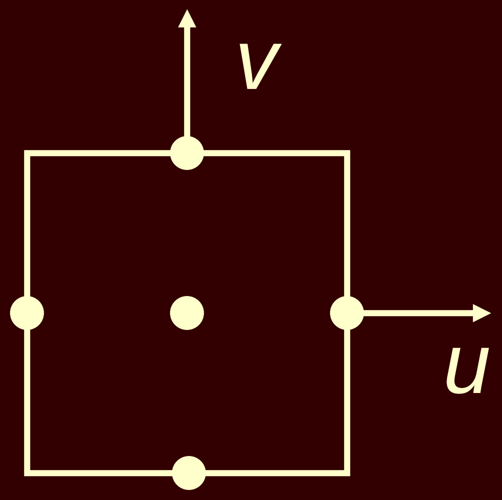
temperature

$$\frac{\partial T}{\partial t} = -(\mathbf{u} \cdot \nabla)T + H$$

$$\mathbf{f} = -\alpha \, \rho \, \mathbf{z} + \beta \, (T - T_{\text{amb}}) \, \mathbf{z}$$

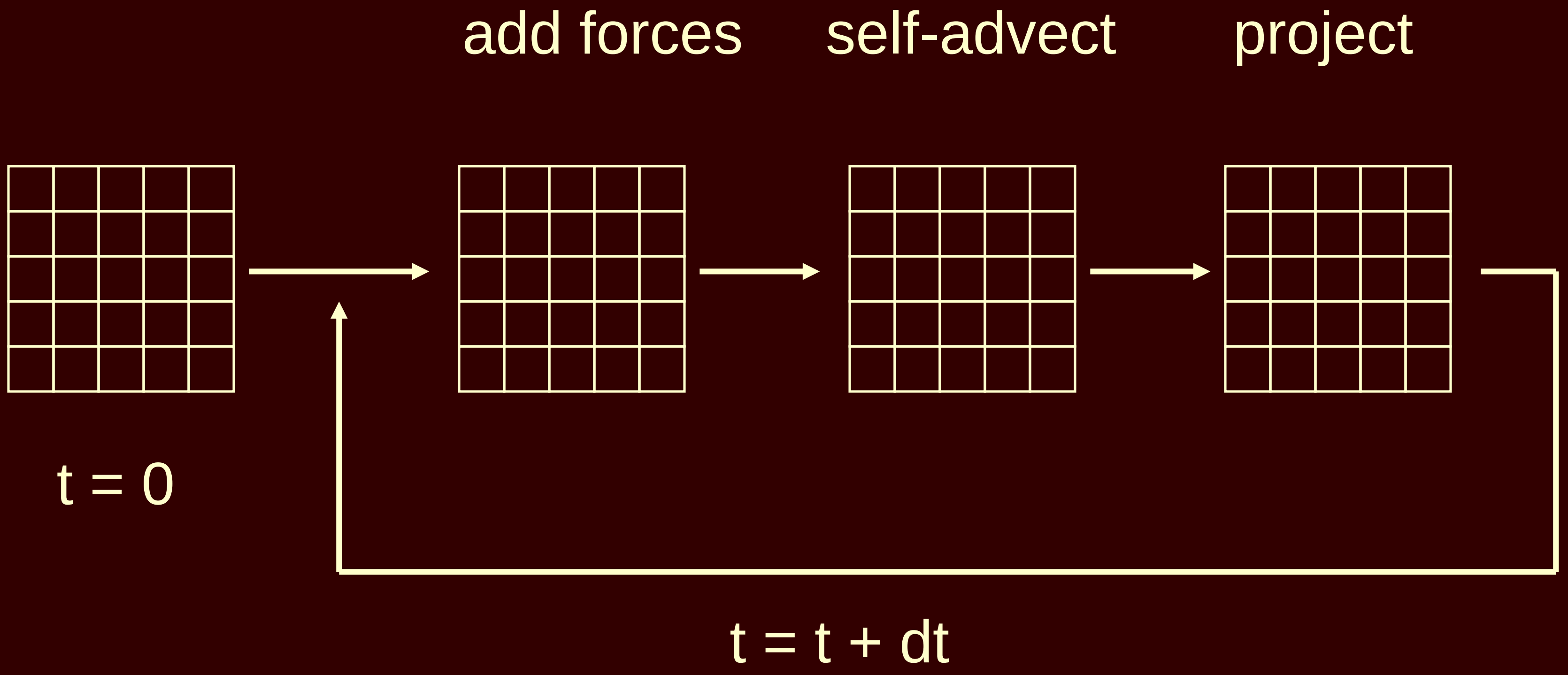
Discretization



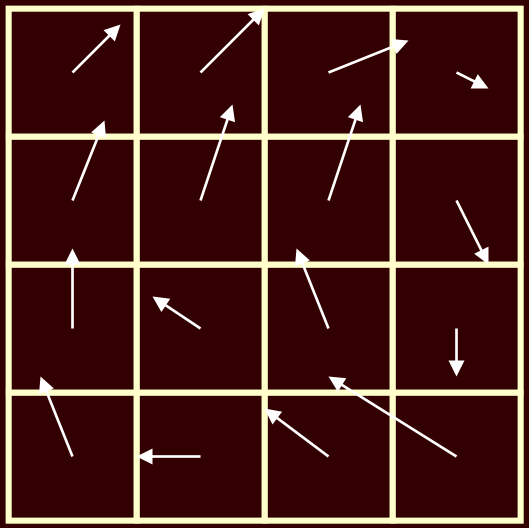


MAC (staggered) grid

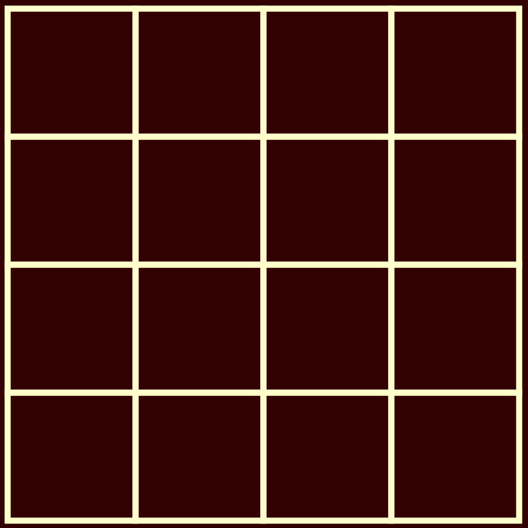
Algorithm



Self-Advection



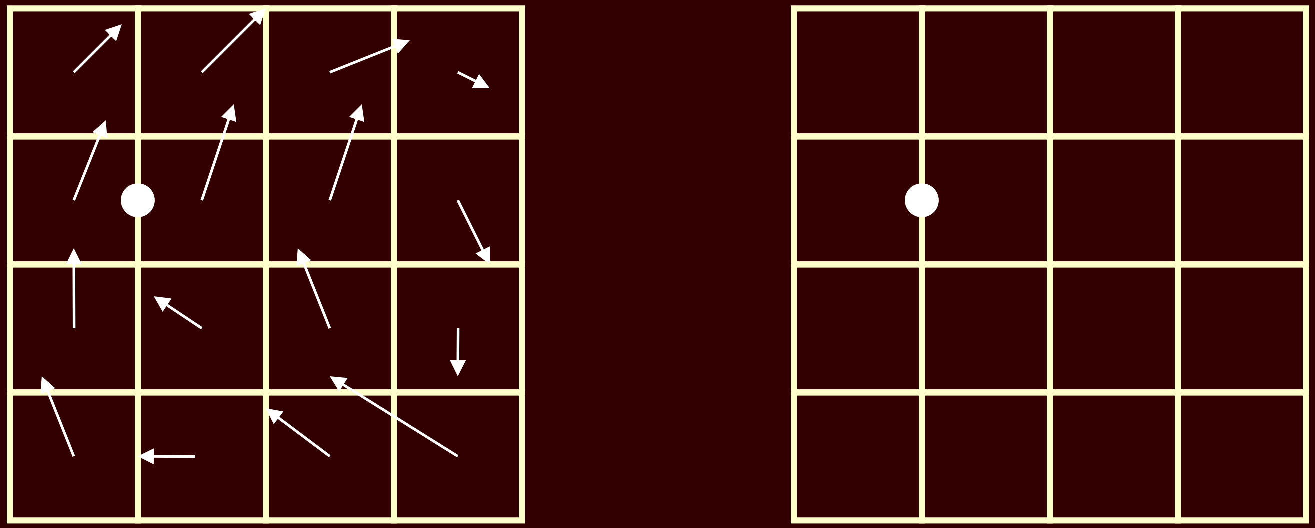
t



t+dt

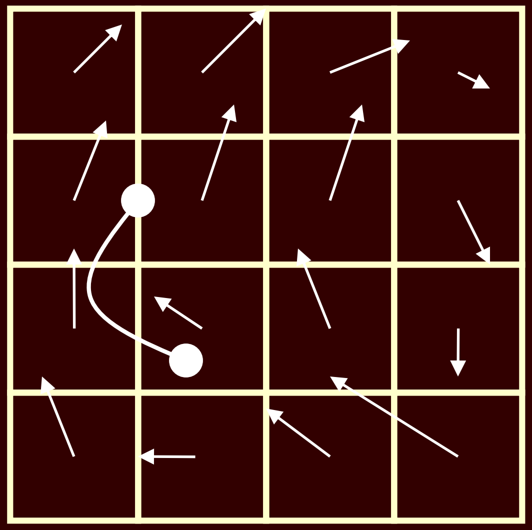
Semi-Lagrangian solver (Courant, Issacson & Rees 1952)

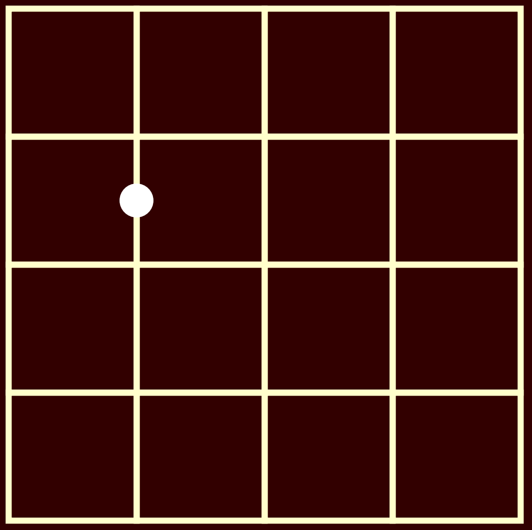
Self-Advection



For each u-component…

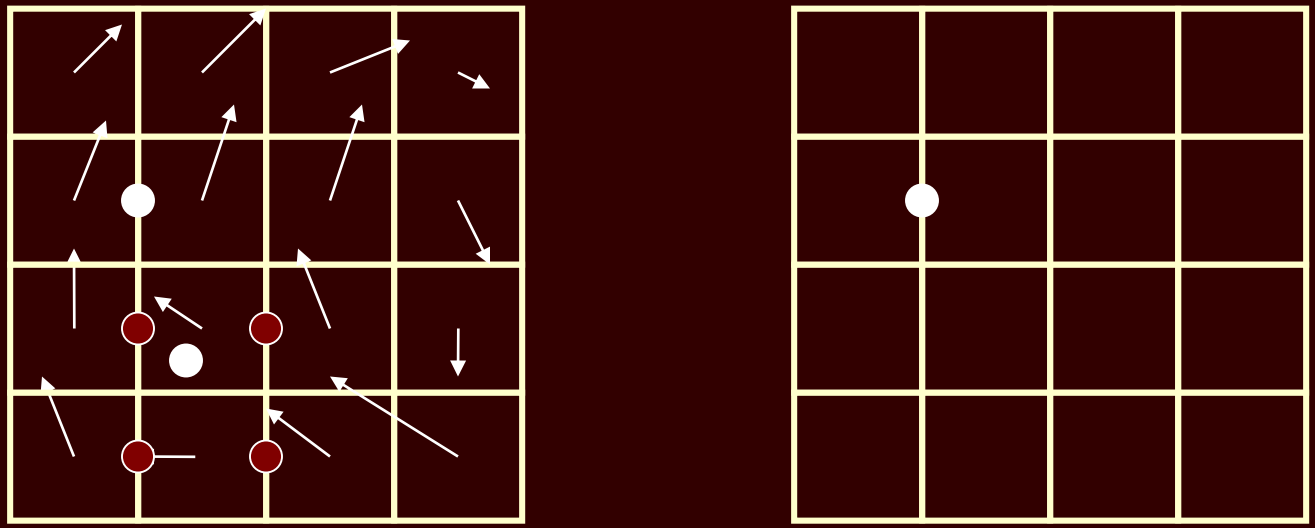
Self-Advection





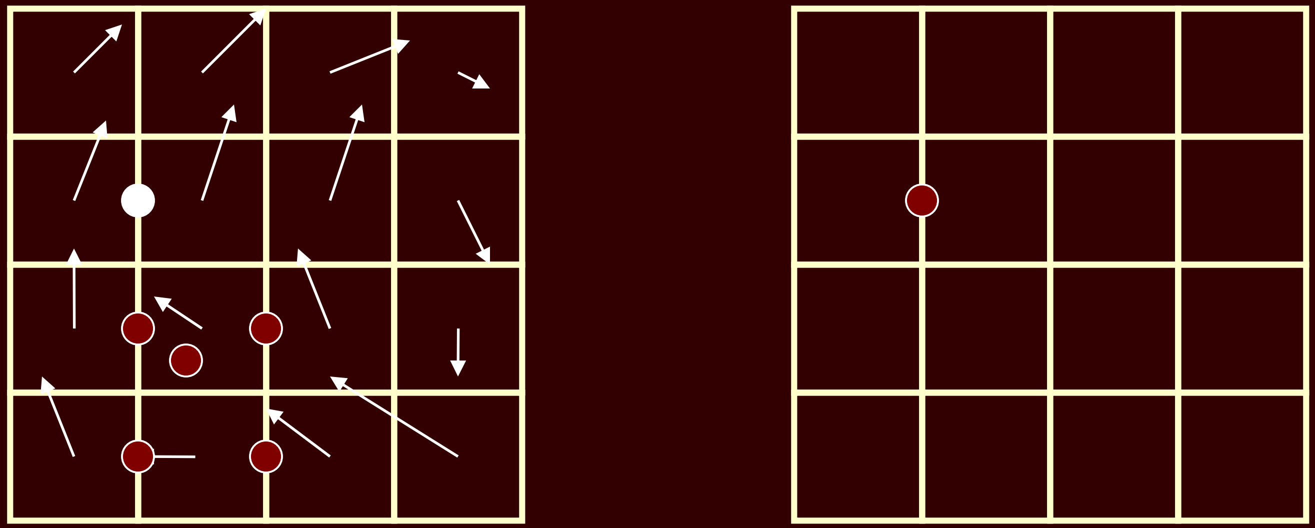
Trace backward through the field

Self-Advection



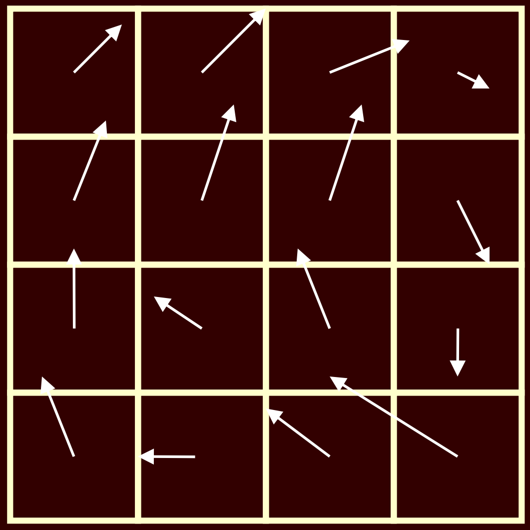
Interpolate from neighbors

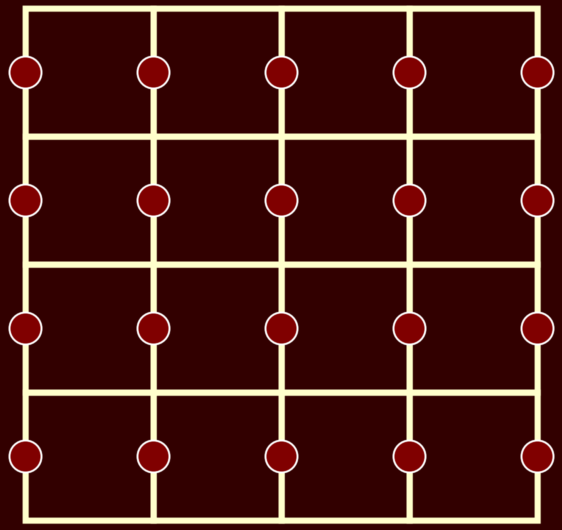
Self-Advection



Set interpolated value in new grid

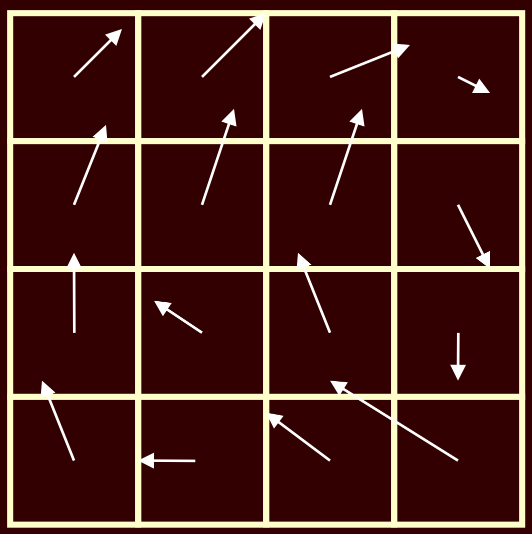
Self-Advection

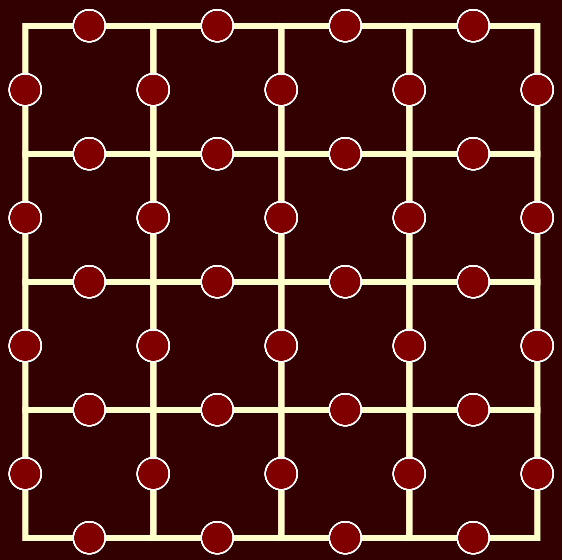




Repeat for all u-nodes

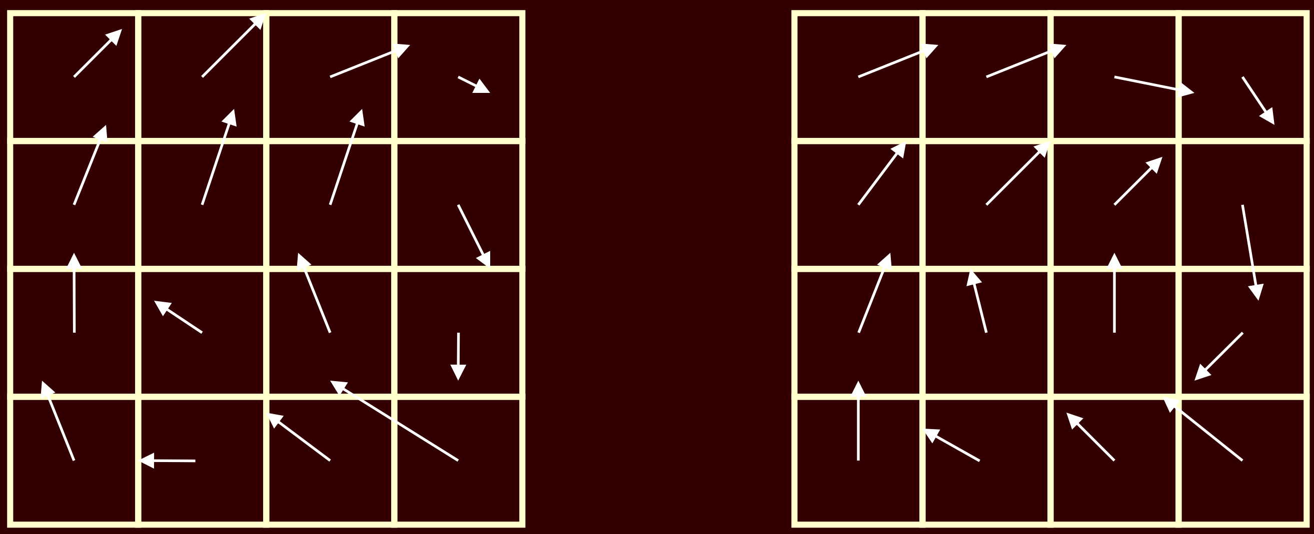
Self-Advection





Similar for v-nodes

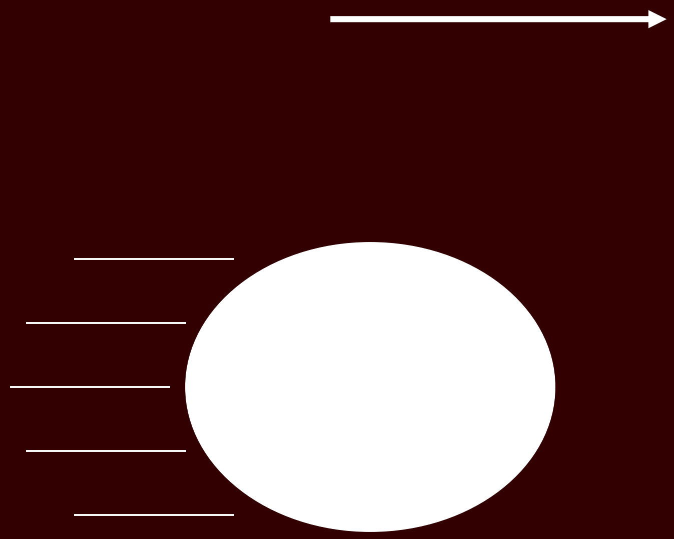
Self-Advection



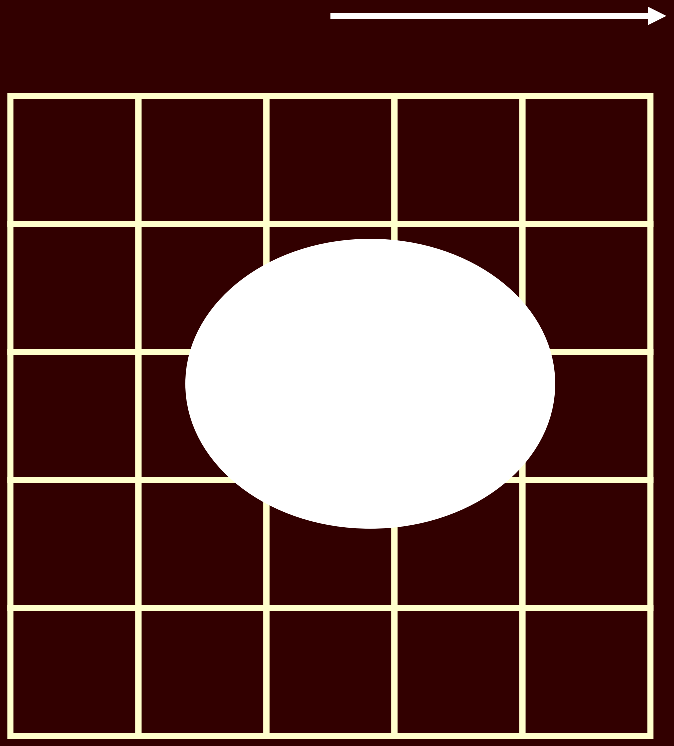
Vmax > Vmax

Advected velocity field

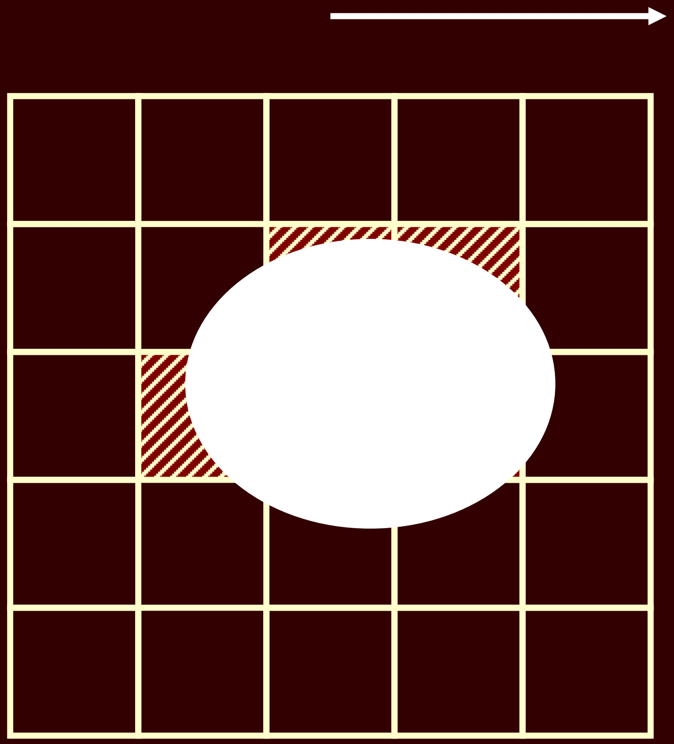
Moving Objects



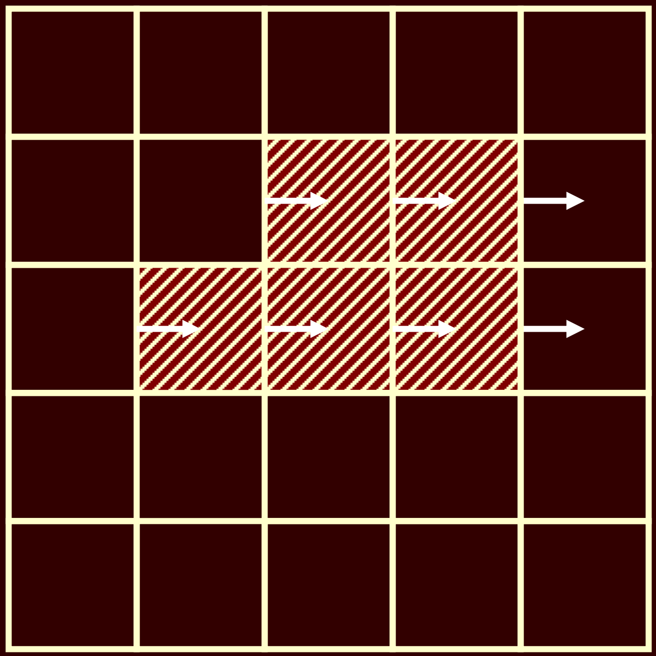
Moving Objects



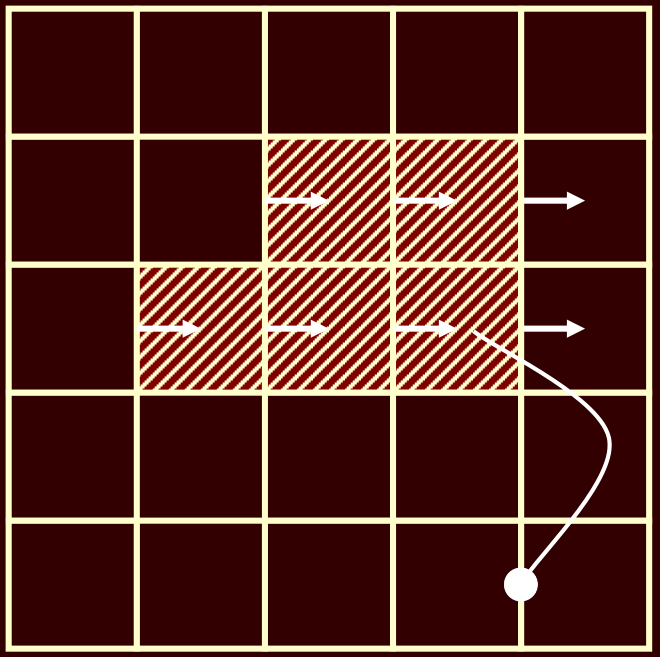
Moving Objects



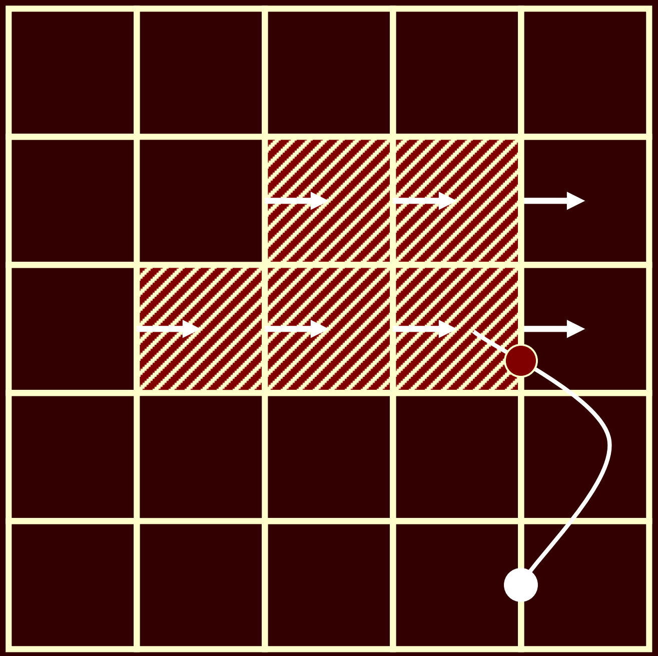
Moving Objects



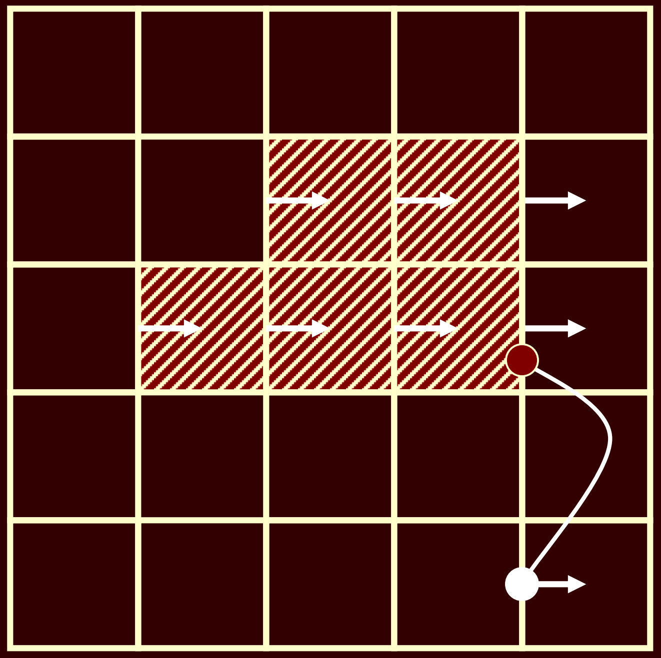
Moving Objects



Moving Objects



Moving Objects



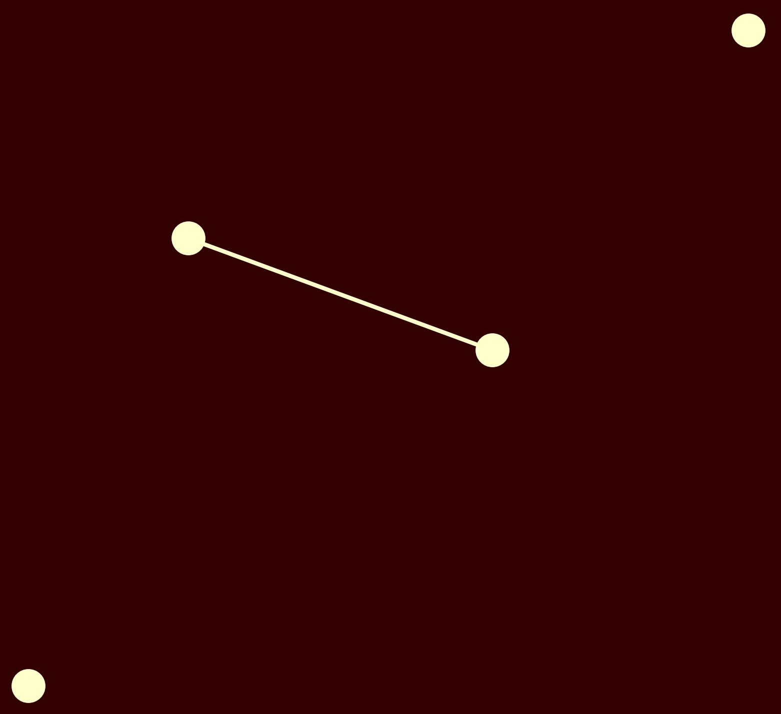
Numerical Dissipation

Stable Fluids dampens the flows

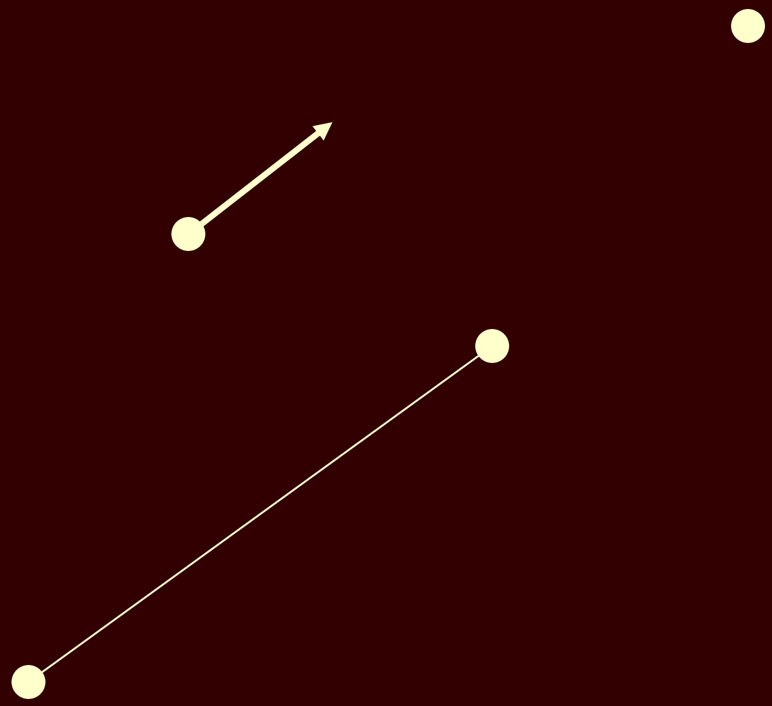
Improve using

- monotonic cubic interpolation
- “Vorticity Confinement” force

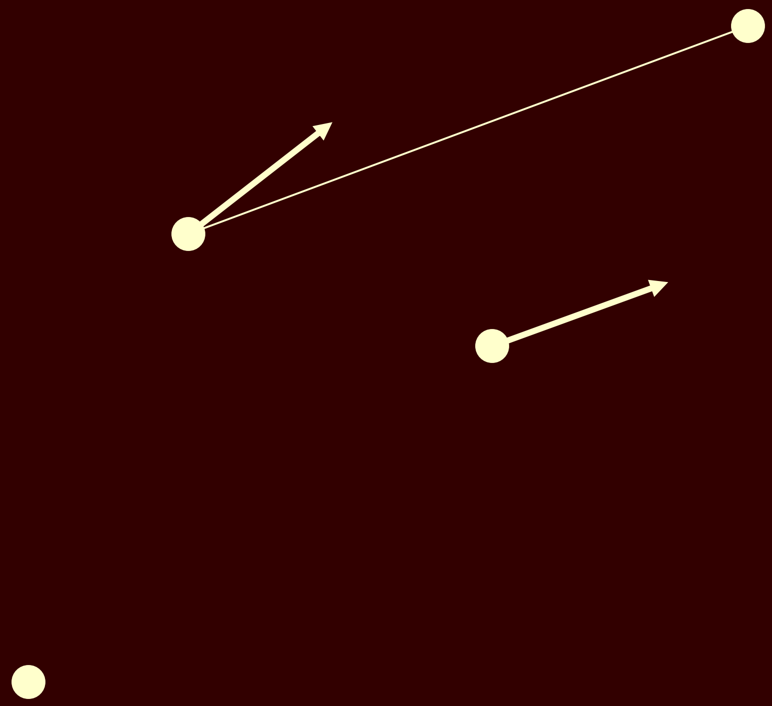
Cubic Interpolation



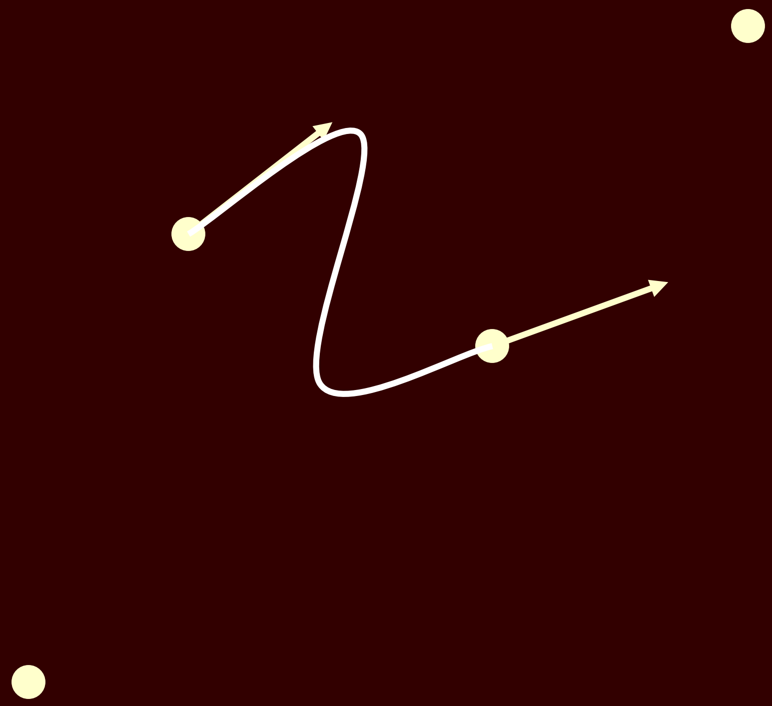
Cubic Interpolation



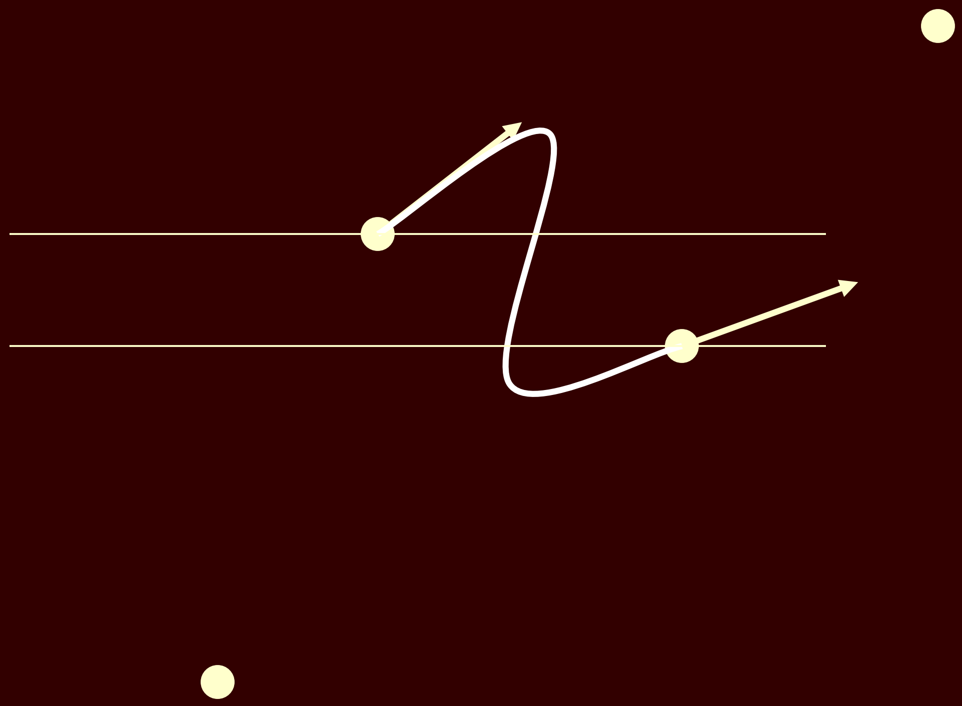
Cubic Interpolation



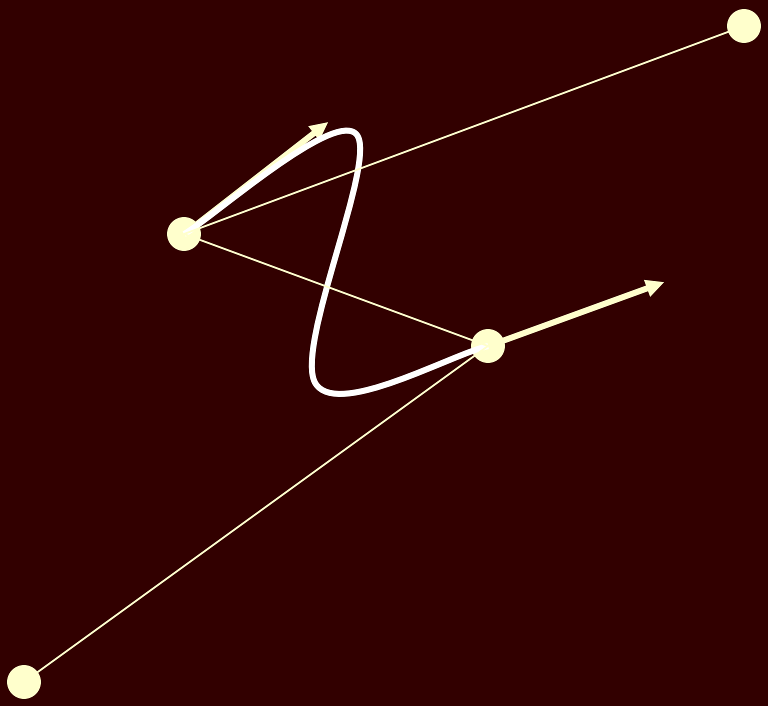
Cubic Interpolation



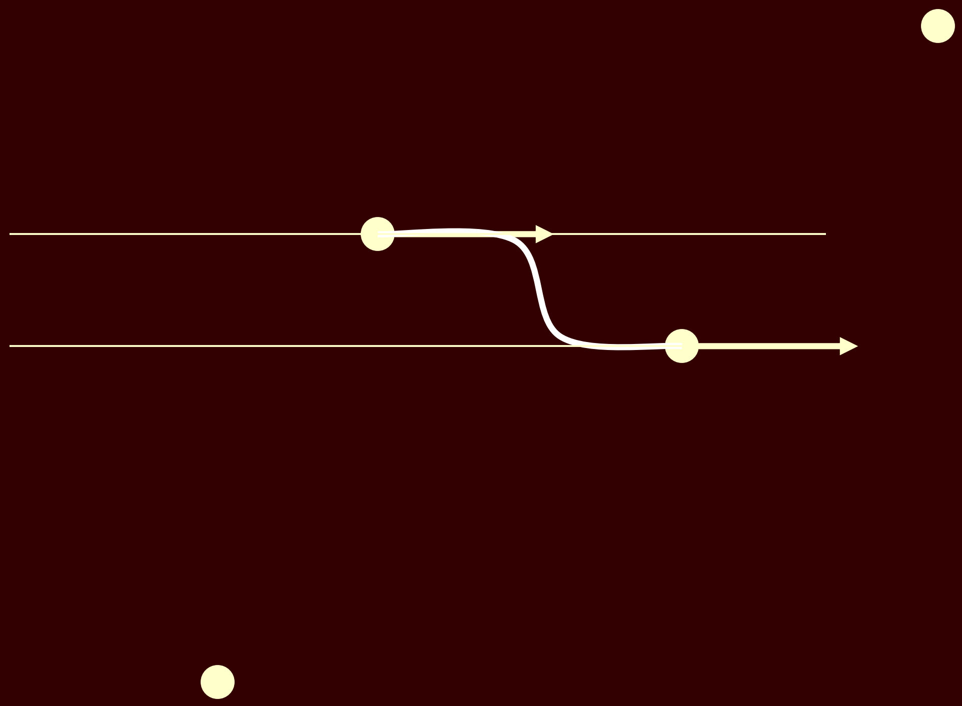
Cubic Interpolation



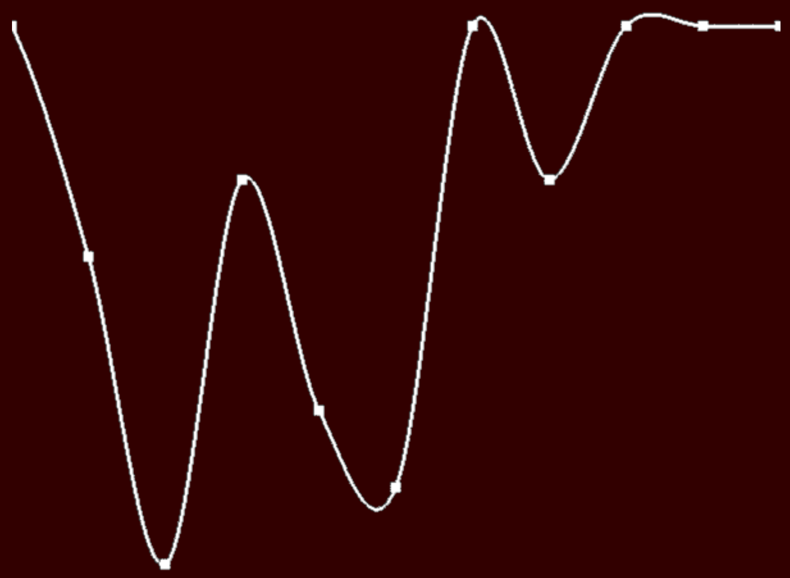
Cubic Interpolation



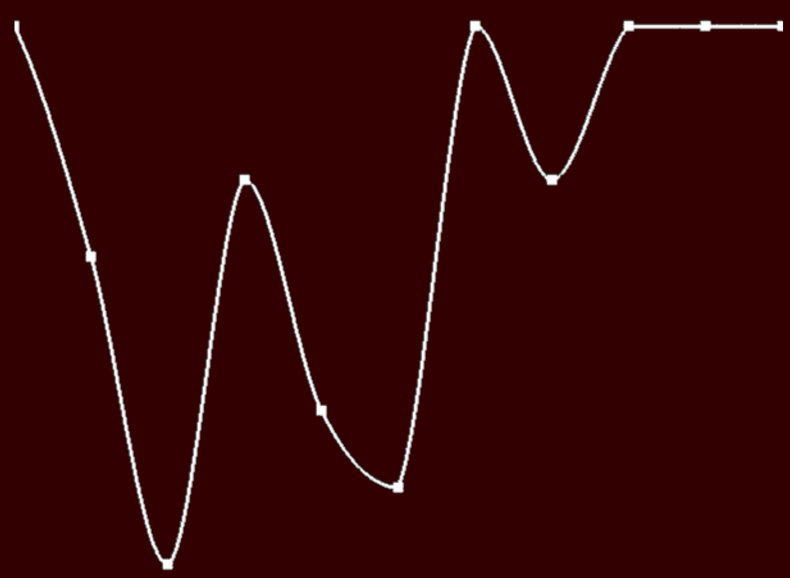
Cubic Interpolation



Cubic Interpolation

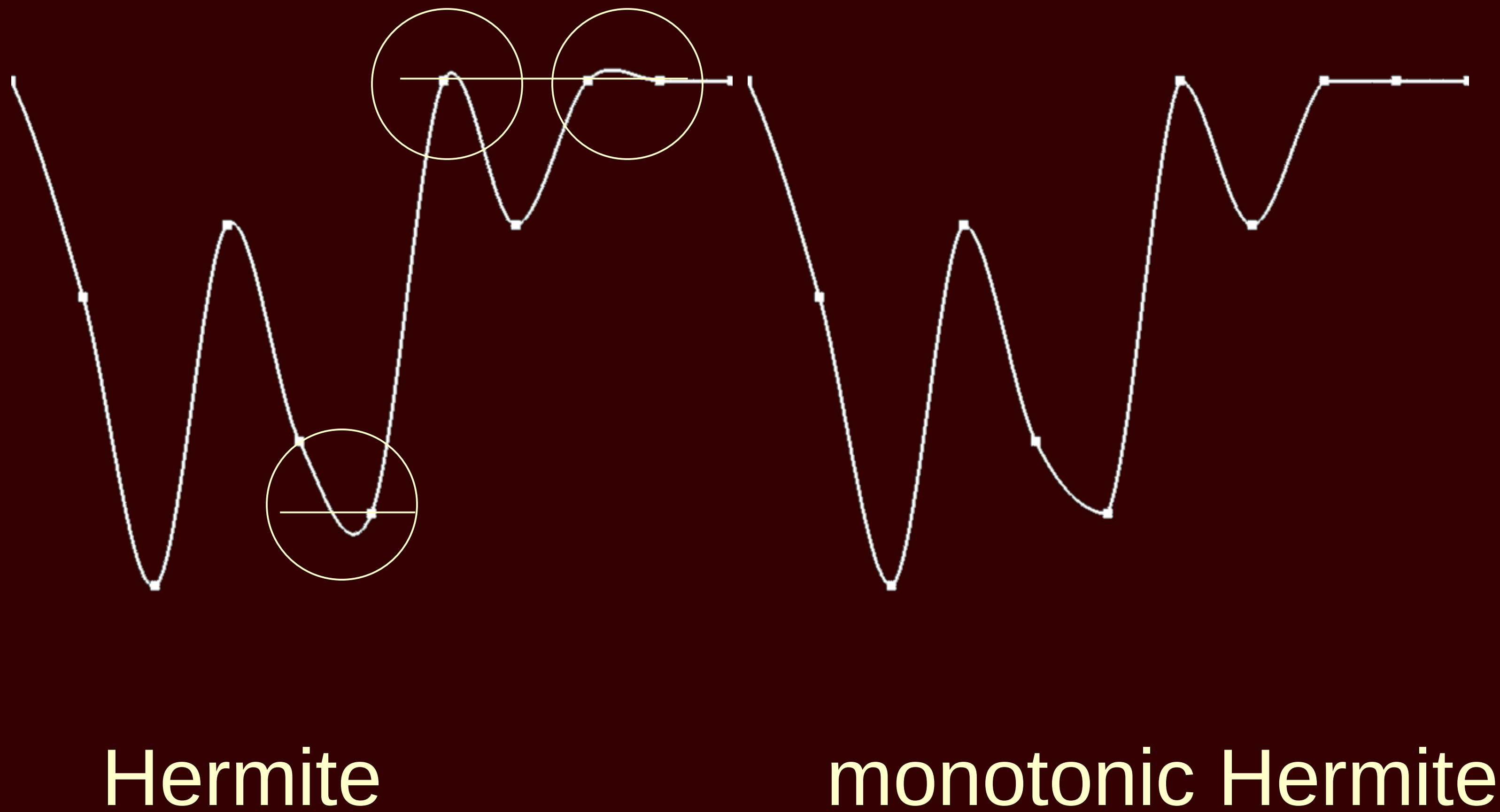


Hermite

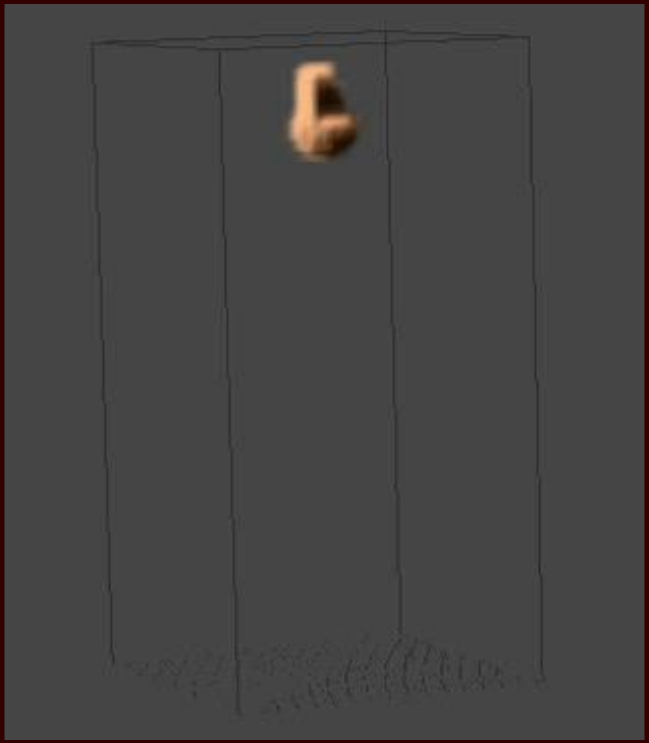


monotonic Hermite

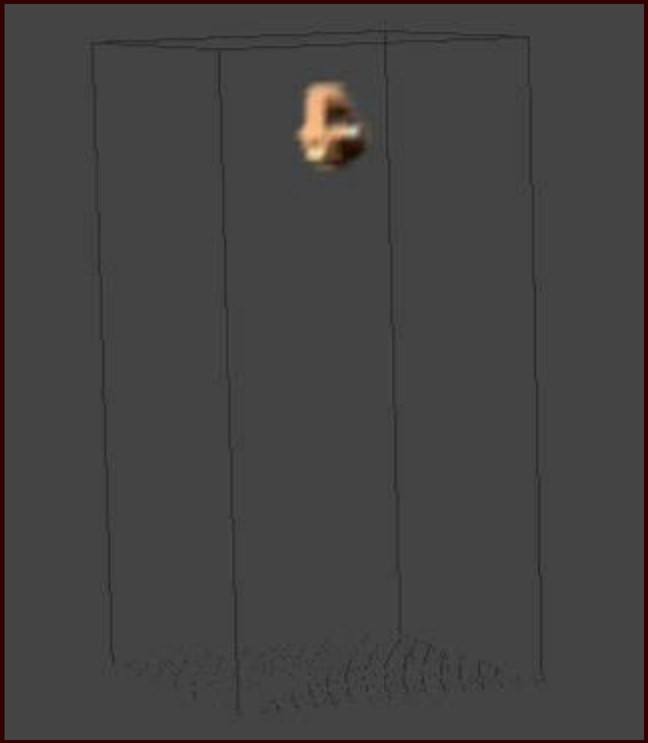
Cubic Interpolation



Cubic Interpolation



Linear



monotonic Hermite

Vorticity Confinement

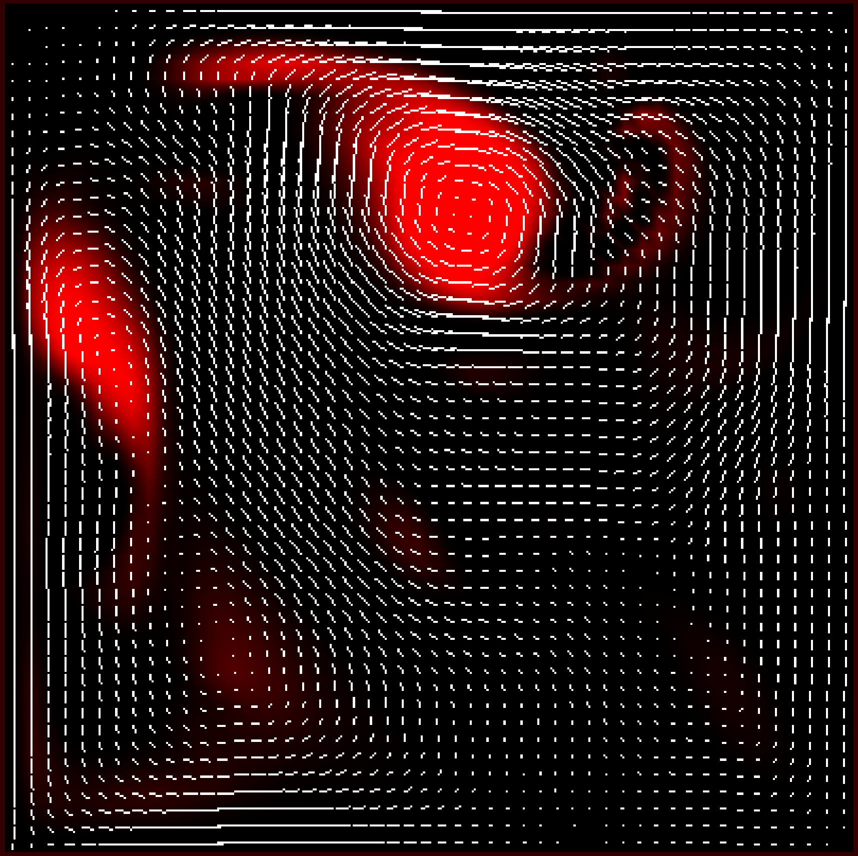
Basic idea:

Add energy lost as an external force

We use “Vorticity Confinement” force

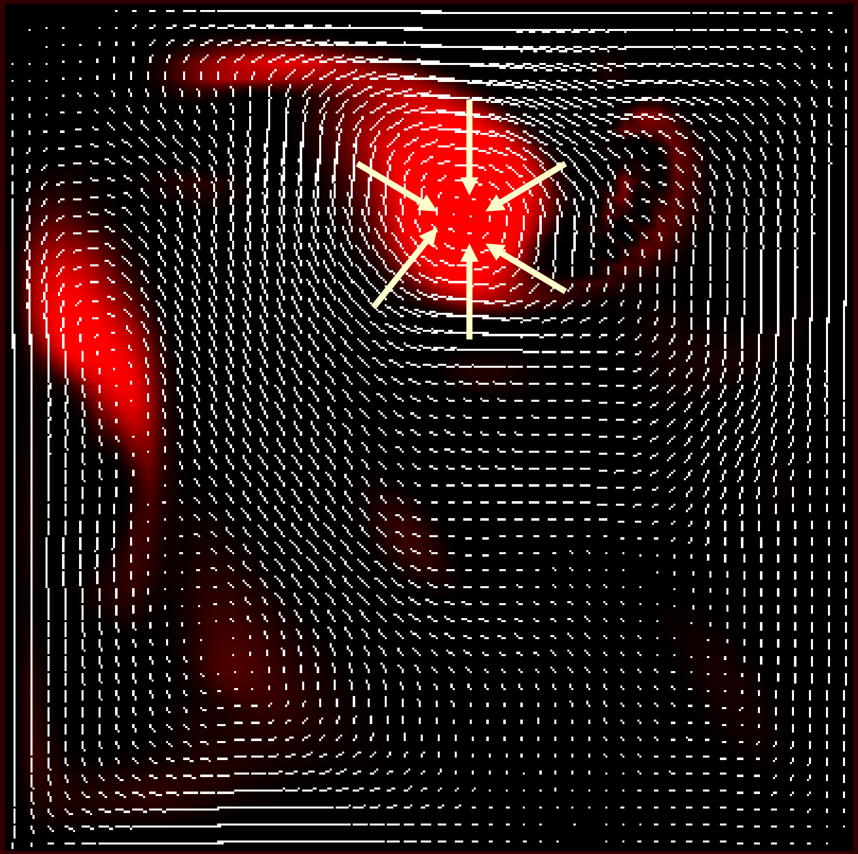
invented by John Steinhoff ~10 years ago

Vorticity Confinement



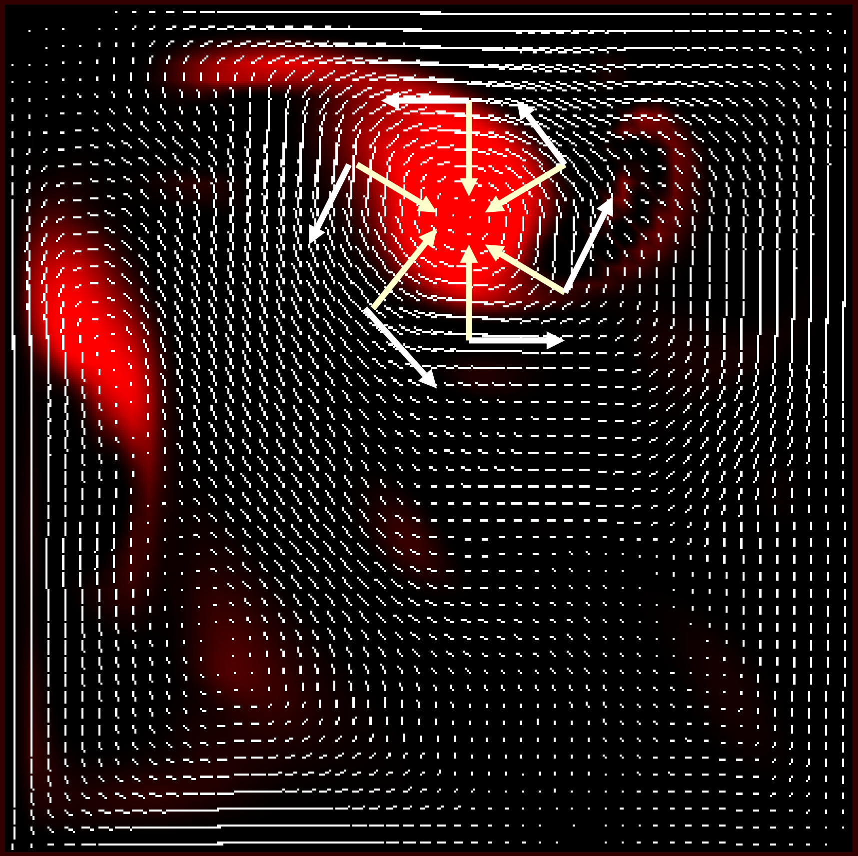
$$\omega = \nabla \times \mathbf{u}$$

Vorticity Confinement



$$\mathbf{N} = \frac{\eta}{|\eta|} \quad \eta = \nabla|\omega|$$

Vorticity Confinement



$$\mathbf{f} = \epsilon\, h\, (\mathbf{N} \times \omega)$$

Vorticity Confinement

Show demo

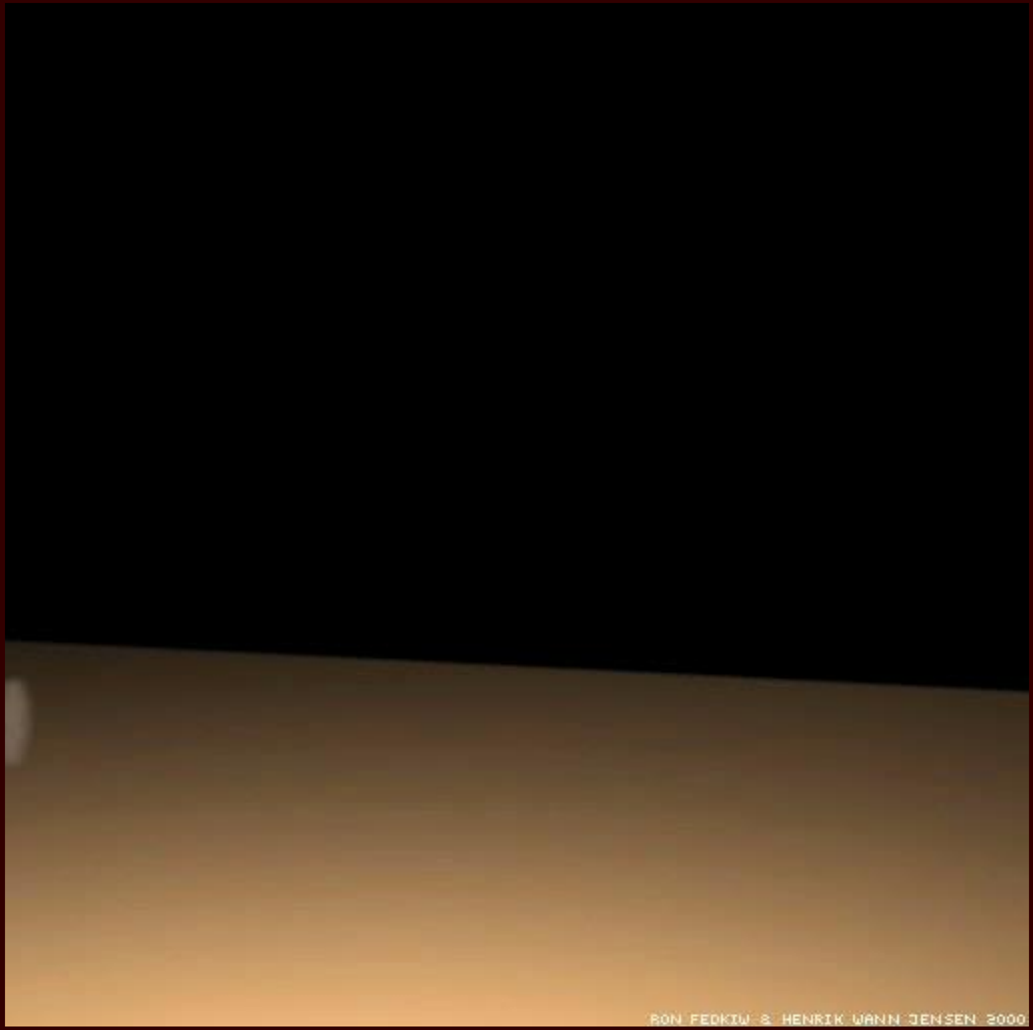
Intel PIII 1GHz + *n*Vidia GeForce2 Go

Results

100x100x40

30 sec.

20-45 min.



Results

90x135x90

75 sec.

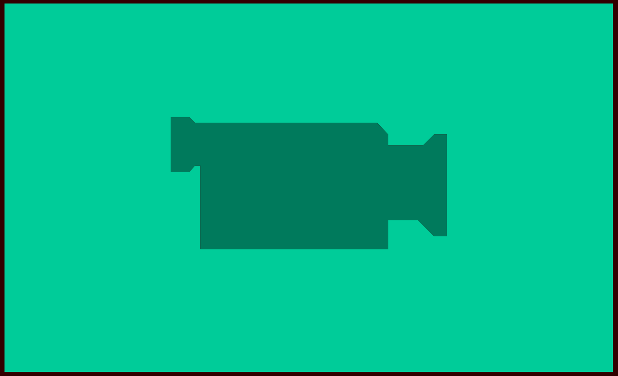
20-45 min.

Results

90x135x90

75 sec.

20-45 min.



PocketPC Demo

Show demo

StrongARM 200MHz + no GPU

Future Work

- Adaptive Grids
- Control
- Other confinement-like forces
- Fire (where there is smoke there is…)
- …