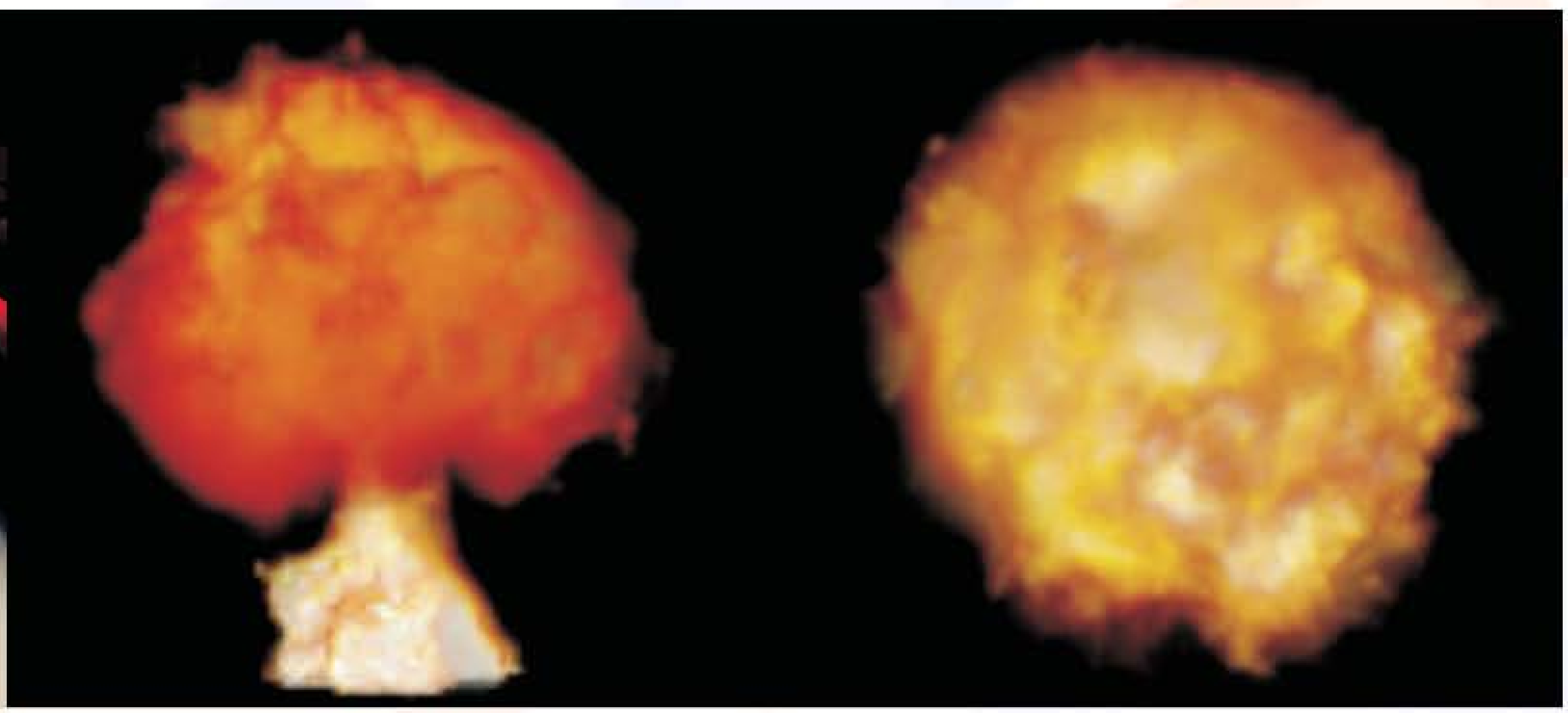
Real-Time Volume Graphics

[11] Game Developer's Guide to Volume Graphics

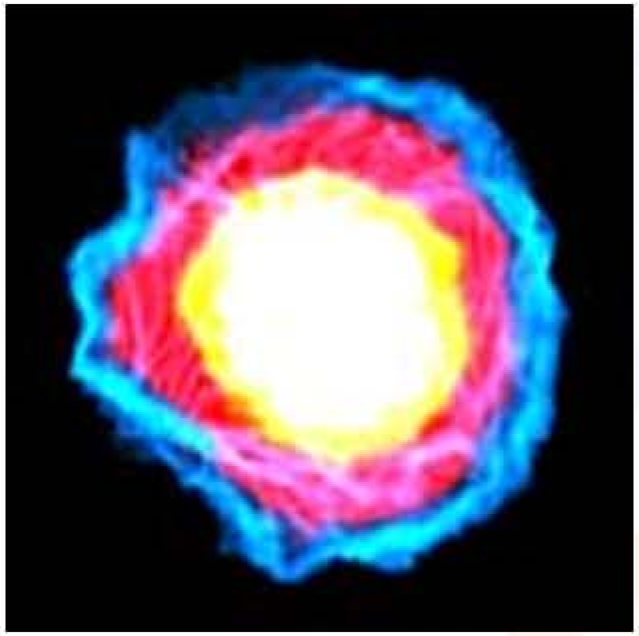
Lokovic and Veach



Krüger and Westermann

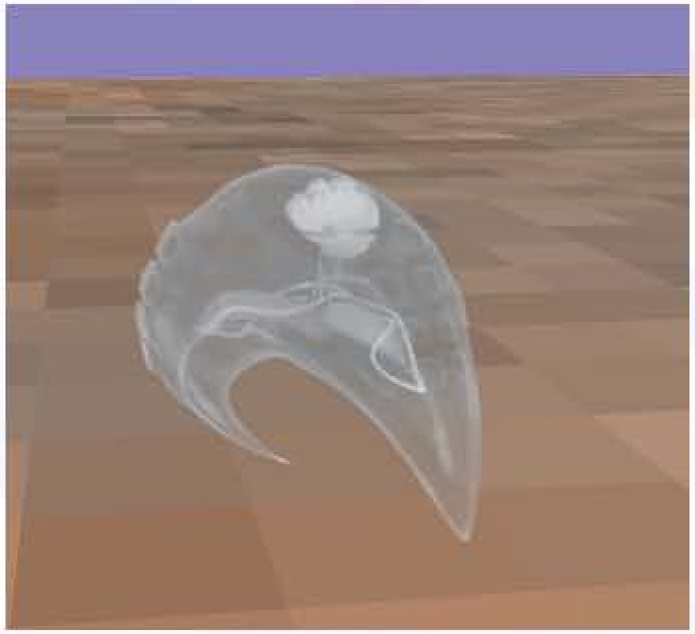
Volumes in Games (1)

- Volumetric effects
- Participating media
- Semitransparent and flexible objects
- Distance volumes for displacement mapping
- ...



NVIDIA SDK

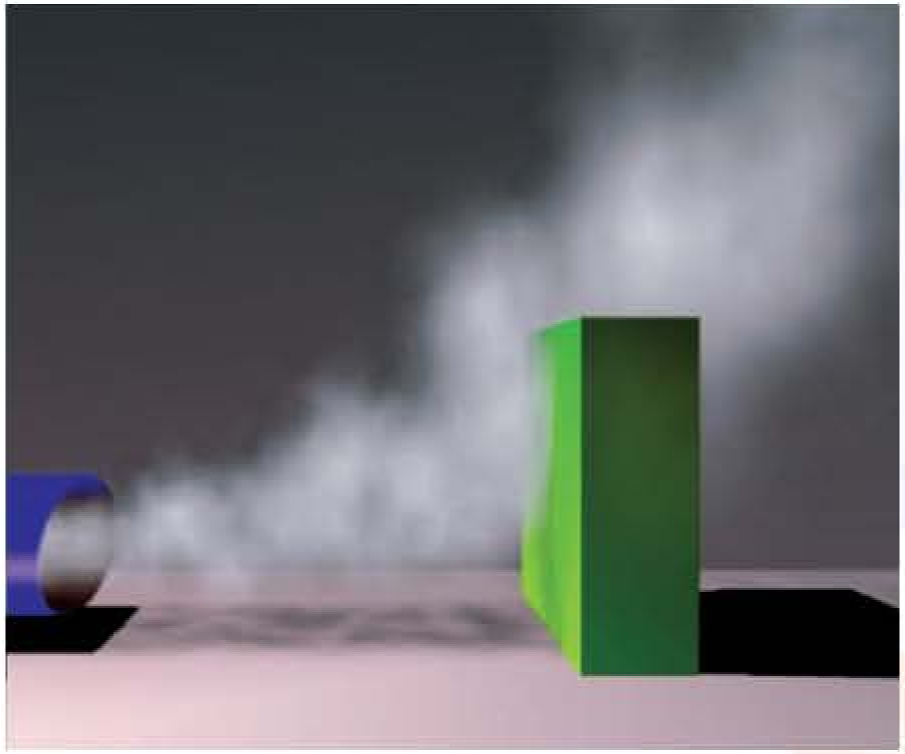
Dobashi et al.



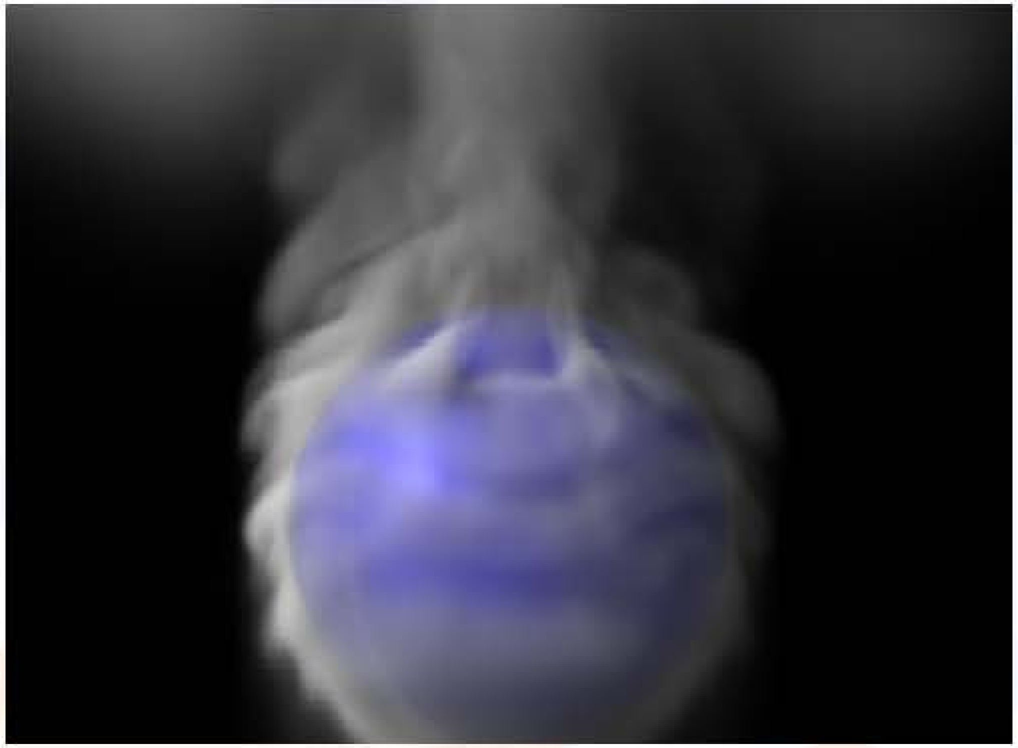
Christof Rezk-Salama

Volumes in Games (2)

- Simulation grids (smoke, fire, ...); level sets
- Pre-computed radiance transfer for volumes
- Irradiance volumes?
  usually not volume rendering



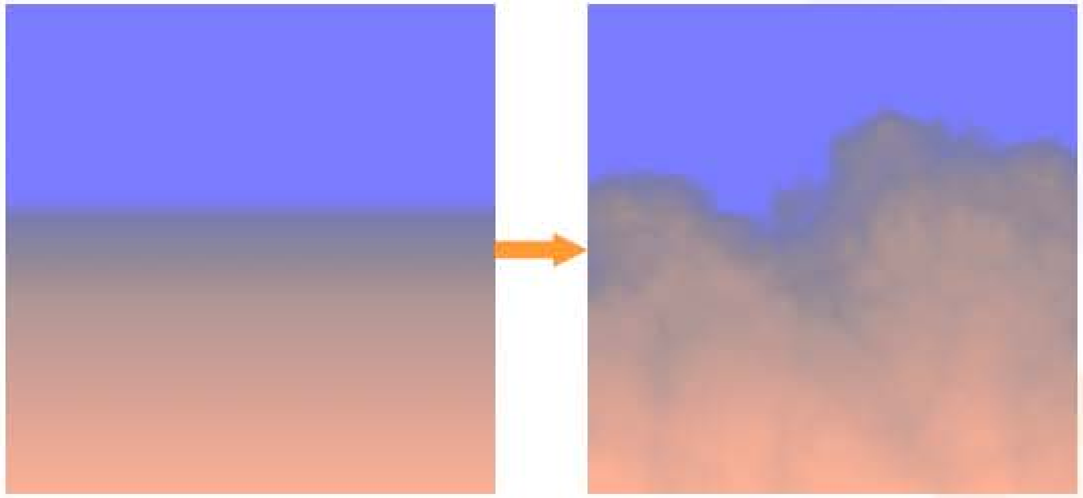
Wei et al.



Fedkiw et al.

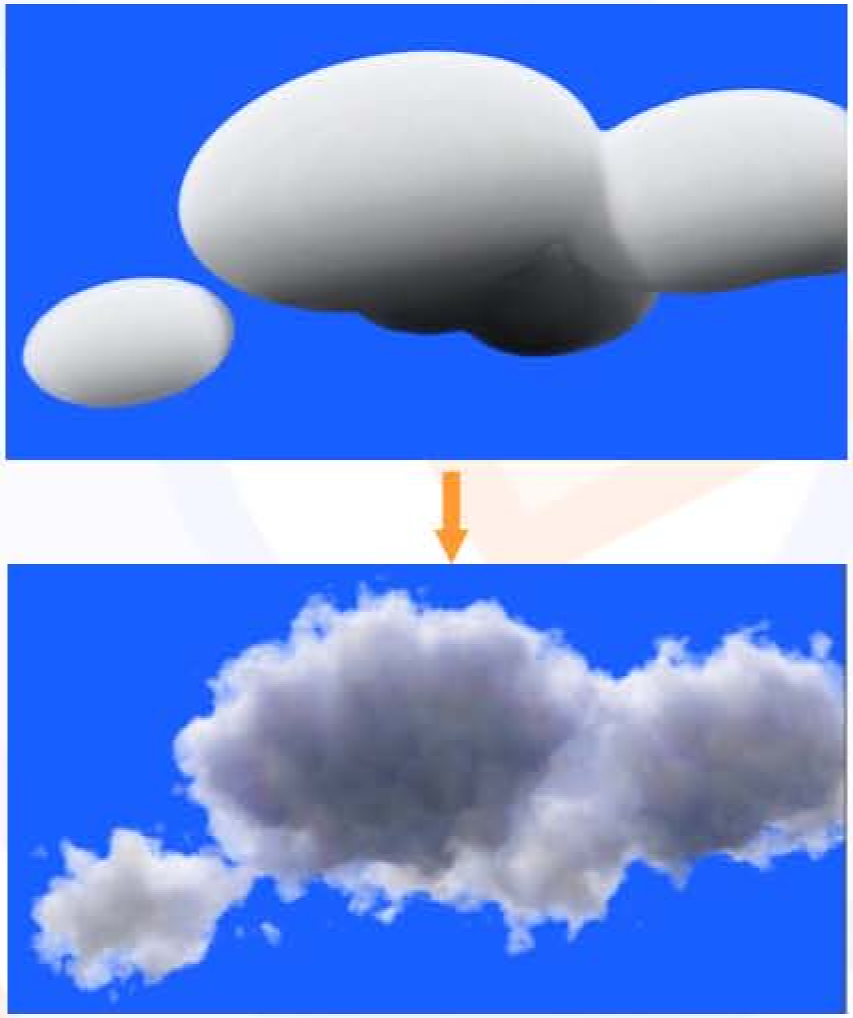
Procedural Volume Modeling

- Constructive volume modeling & animation
- Build volume from basic blocks
- Ken Perlin, David Ebert, Jim Blinn, ...



Perlin

Schpok et al.



Kniss et al.

Volume Rendering and Game Engines

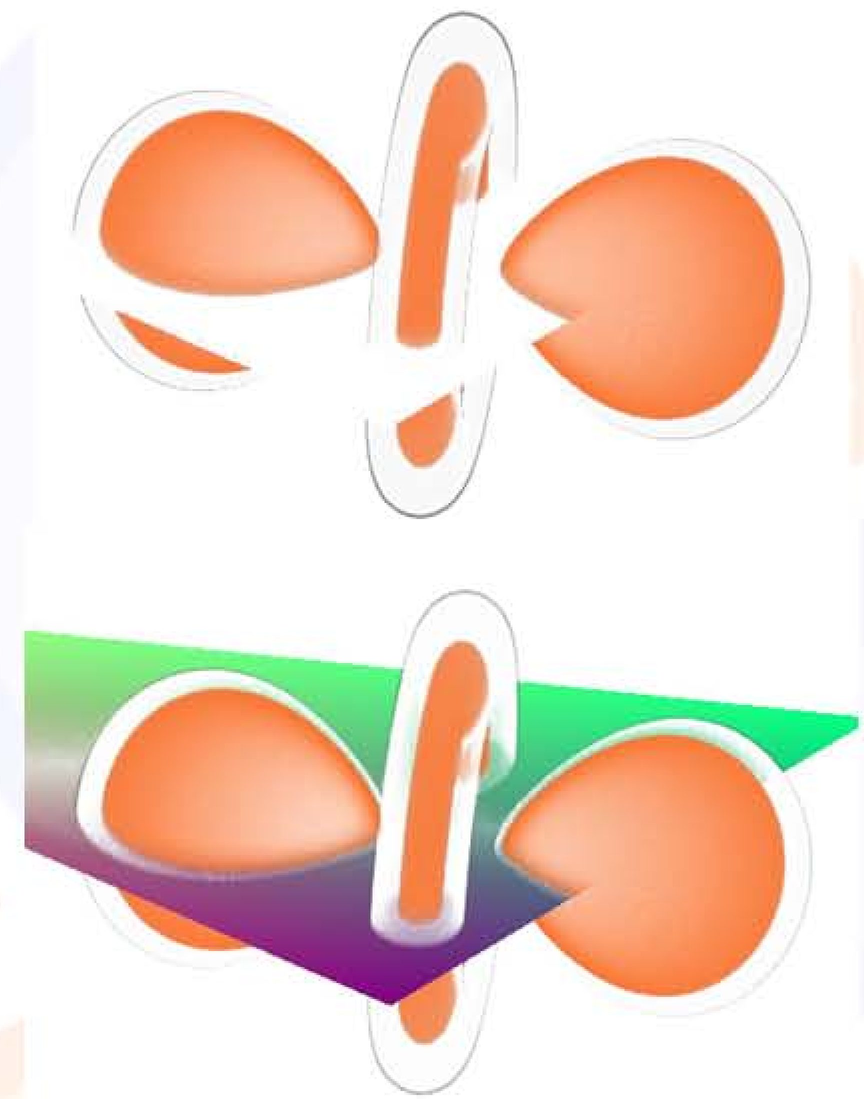
- Integration issues
  - Opaque scene geometry and volumes
  - Semitransparent scene geometry and volumes
  - Viewpoint inside the volume (e.g., fog, clouds)

  - Integration with lighting
  - Integration with shadows

Crysis / Crytek

No "Stand-Alone" Volume Rendering

- Integration with scene geometry
  - Correct visibility (volumes are semitransparent!)
  - Handle "room-filling" volumes
- Handle multiple volumes
- Integration with occlusion culling
- Integration with scene lighting
- Integration with HDR



Special Effects with Billboards (1)

- Billboards "cache" expensive effects
- Problem: clipping of billboards against geometry

Harris et al.

Special Effects with Billboards (2)

- Potential solutions
  - Take special care of billboard placement (e.g., cloud rendering of Mark Harris)
  - Fade out billboard according to z-distance to geometry (used, e.g., in Crysis/Crytek)
  - Use full volume rendering (still expensive, but improving rapidly)

Ingredients

- Slicing
- Ray-casting
- Local and global illumination
- Pre-integration
- Volume modeling and animation
- Performance optimizations

Integration with Scene Geometry

- Opaque scene geometry
- Semitransparent scene geometry
- Viewpoint inside the volume
- Visibility ordering for multiple volumes
- Occlusion culling

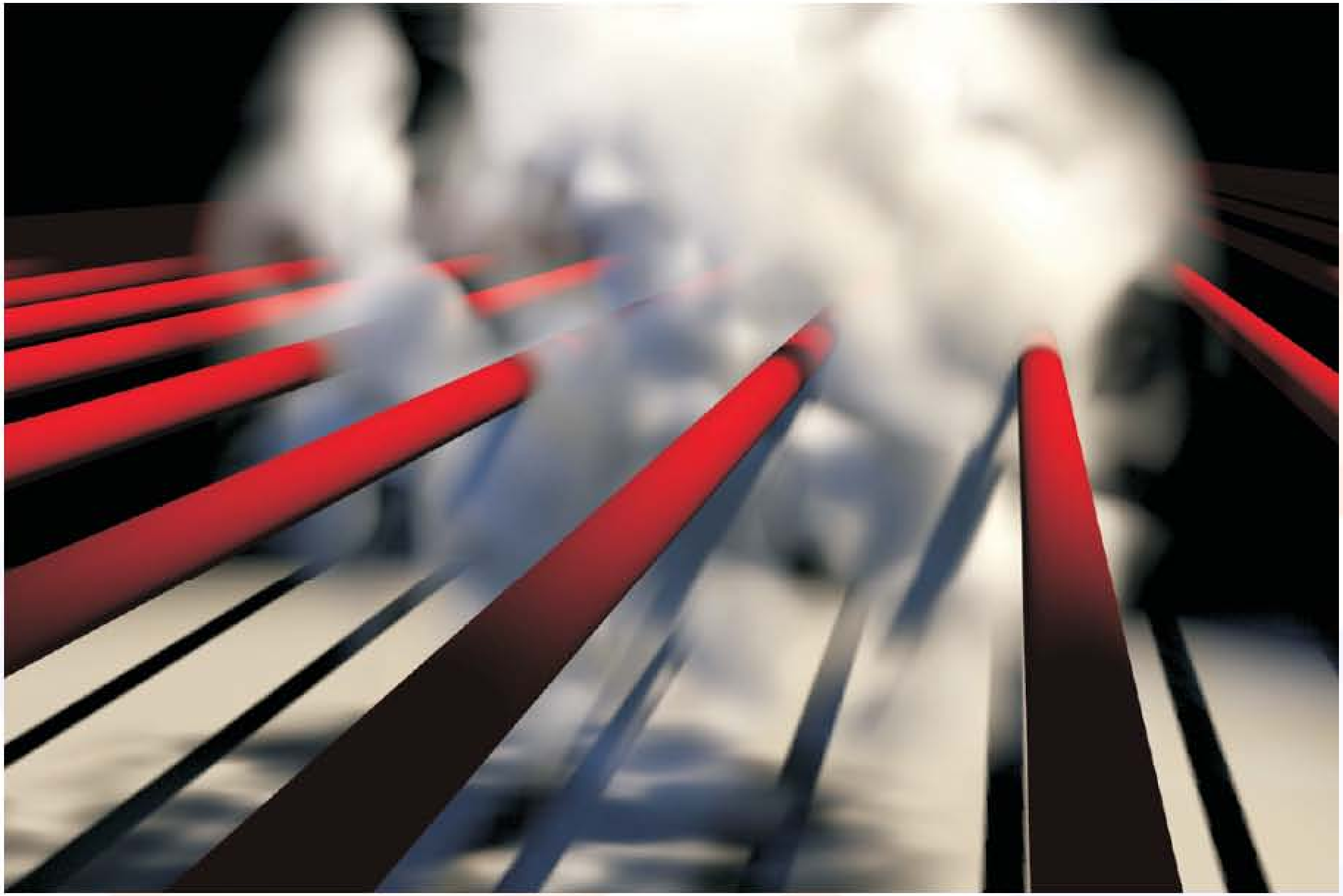
Shadows from Detailed Geometry

- Alpha coverage results in "semi-transparent" pixels
- Percentage of light that is occluded

Lokovic and Veach

Deep Shadow Maps (1)

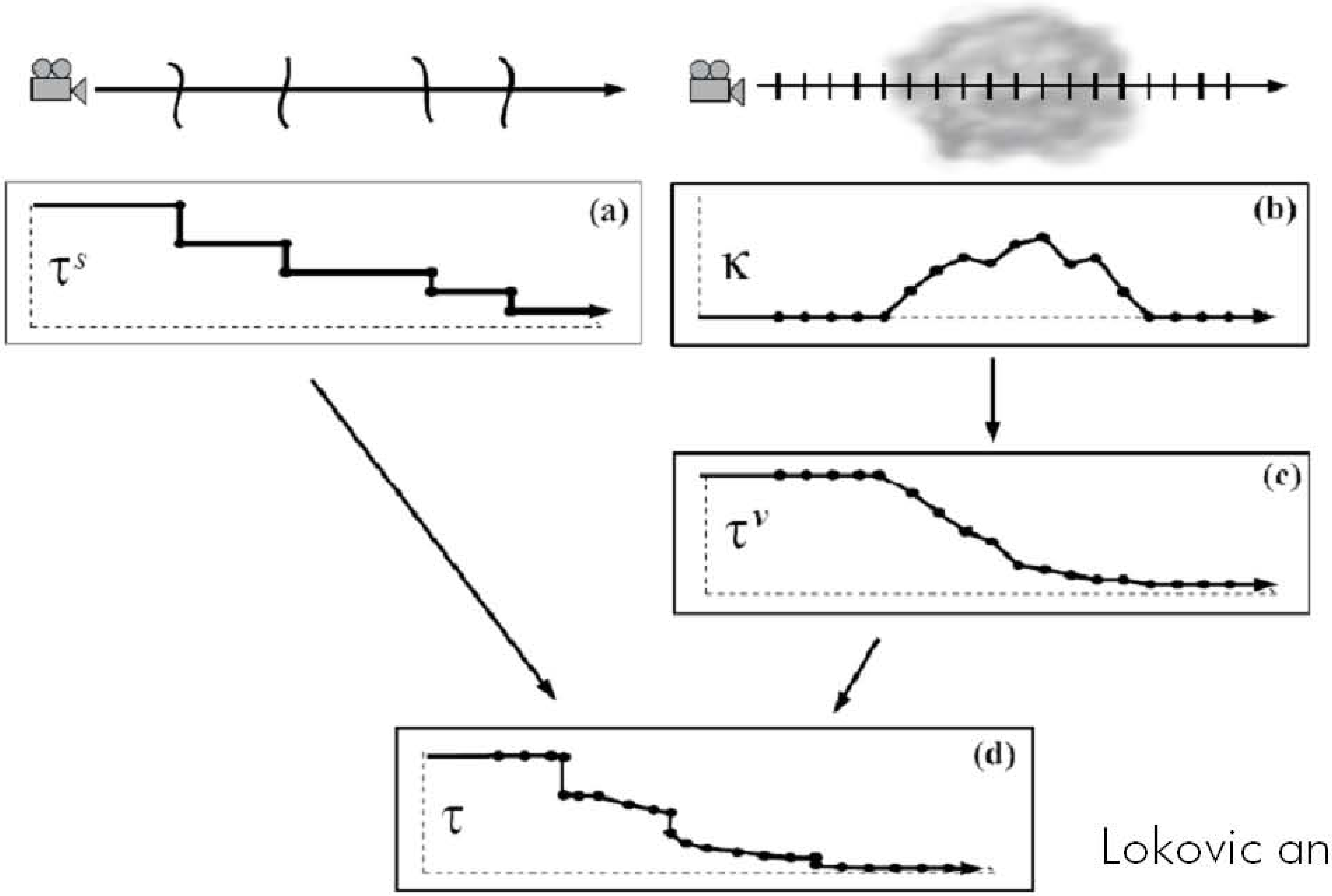
- Unify shadows from geometry and volumes



Lokovic and Veach

Deep Shadow Maps (2)

- Geometry and volumes combine easily



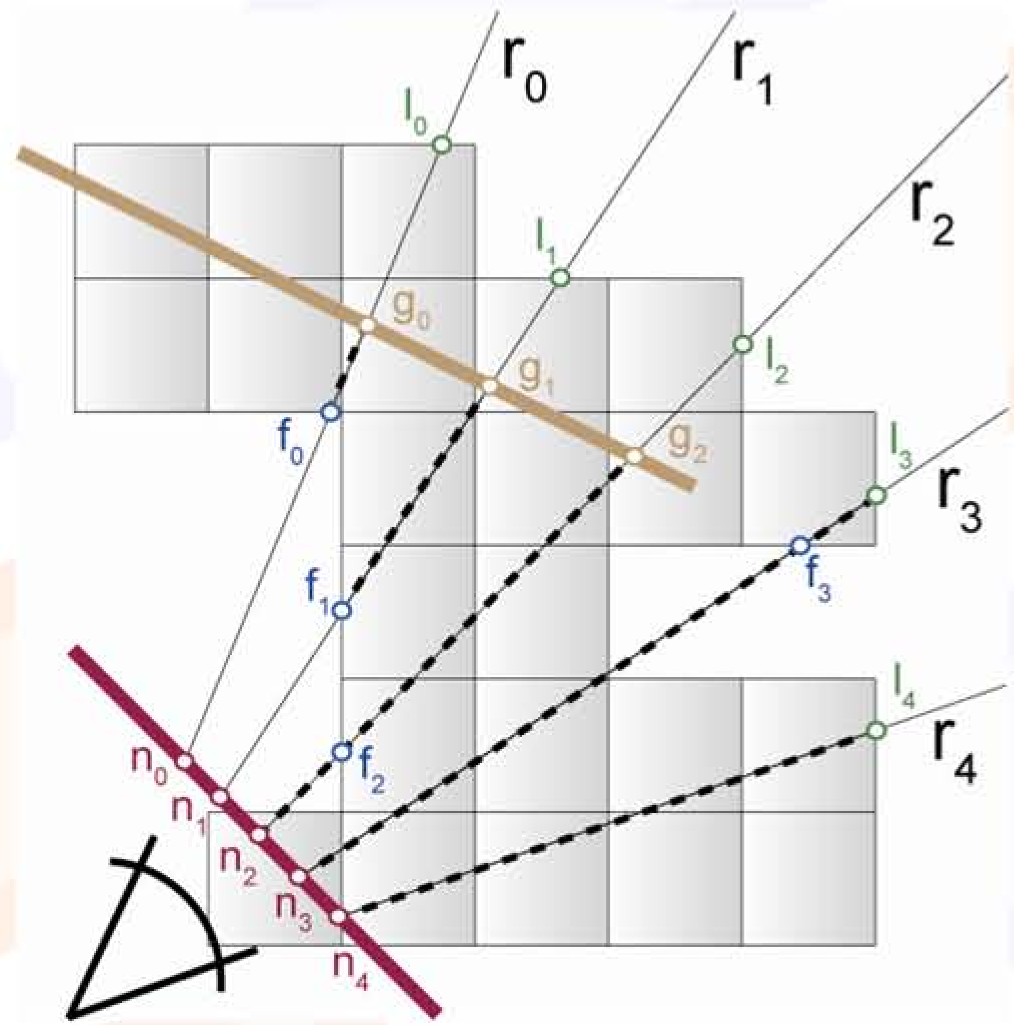
Lokovic and Veach

Opaque Scene Geometry (1)

- Rasterize scene geometry into depth buffer
- Volume ray-caster stops rays at these depths
- Ray-cast on top of geometry or blend afterward

Opaque Scene Geometry (2)

- Back-project scene depth into volume space [0,1]
- Use these volume space coordinates to stop rays
- Works for arbitrarily complicated scenes



Opaque Scene Geometry (3)

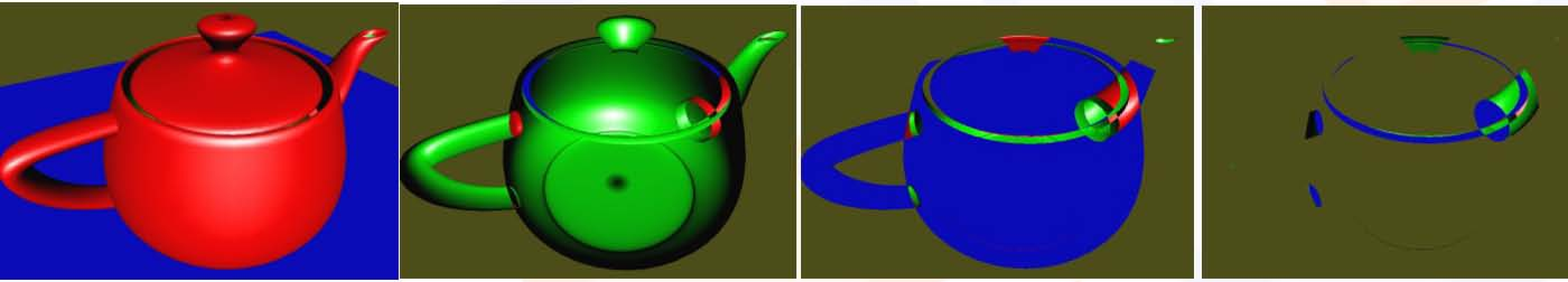
```
float4 main(float2 window_position:  TEXCOORD0,
            uniform sampler2D depth_texture,
            uniform float4x4 ModelViewProjInverse) :  COLOR
{
    // compute the homogeneous view-space position
    // window_position is in [0,1]^2 and depth in [0,1]
    float4 hviewpos;
    hviewpos.xy = window_position;
    hviewpos.z  = tex2D(depth_texture, window_position);
    hviewpos.w  = 1.0;
    // we need this to be in [-1,1]^3 clip space
    hviewpos = hviewpos * 2.0 - 1.0;

    // back-project to homogeneous volume space
    float4 hvolpos = mul(ModelViewProjInverse, hviewpos);

    // return normalized volume-space position
    return (hvolpos / hvolpos.w);
}
```

Transparent Scene Geometry

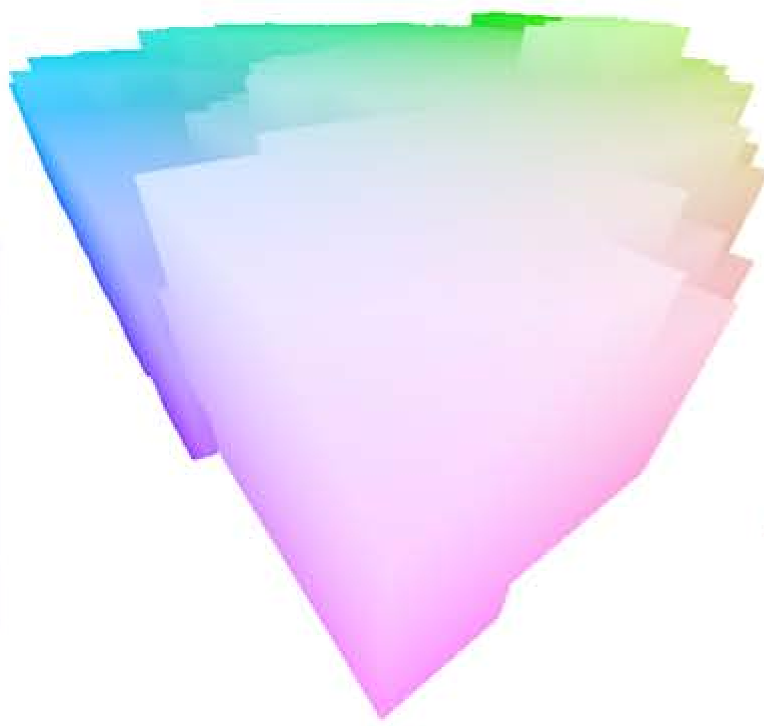
- Render in depth layers (depth peeling)
- Ray-cast for each layer and handle layer as "opaque"
- Very rasterization-intensive
- Only real time for small number of layers

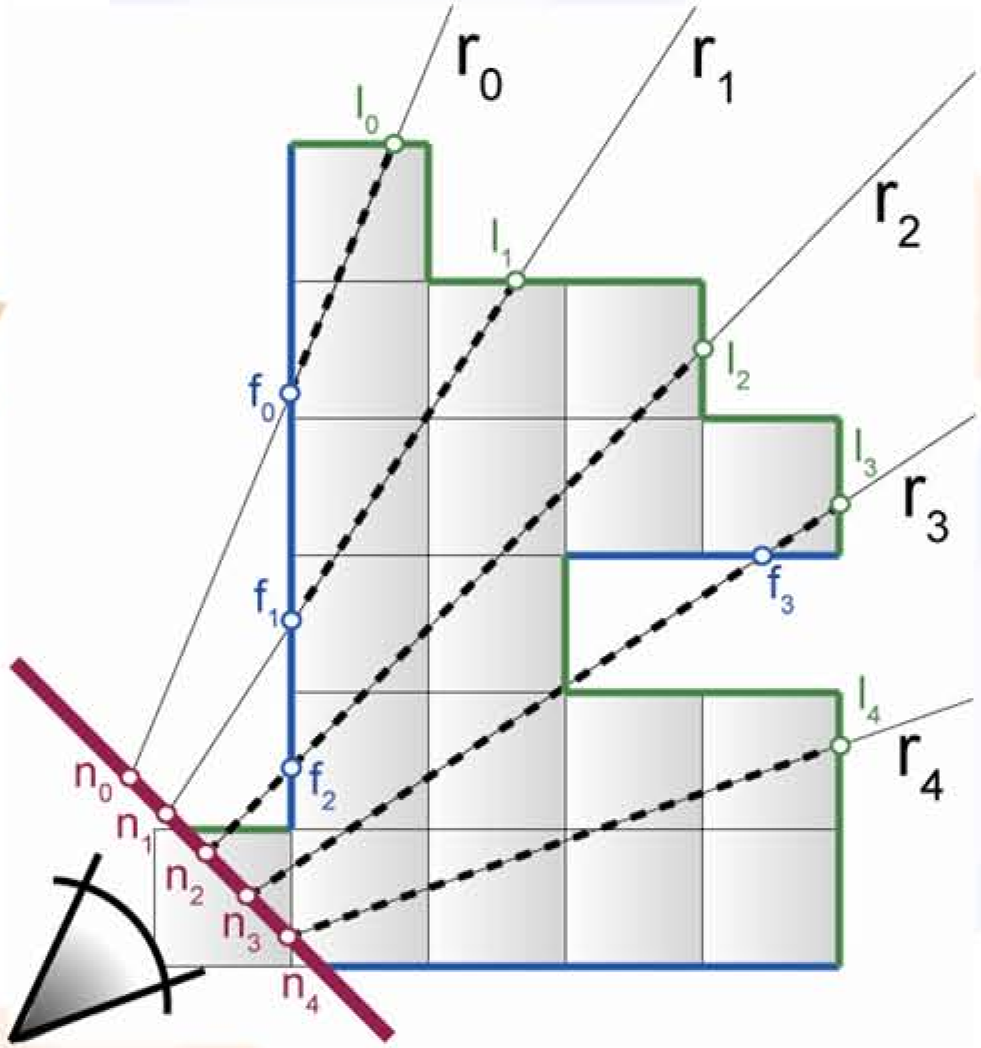


Cass Everitt

Viewpoint Inside the Volume

- Two main possibilities
  - Cap with geometry (clip near plane against frustum)
  - Render near plane and use stencil/depth buffer





Integration with Scene Lighting...

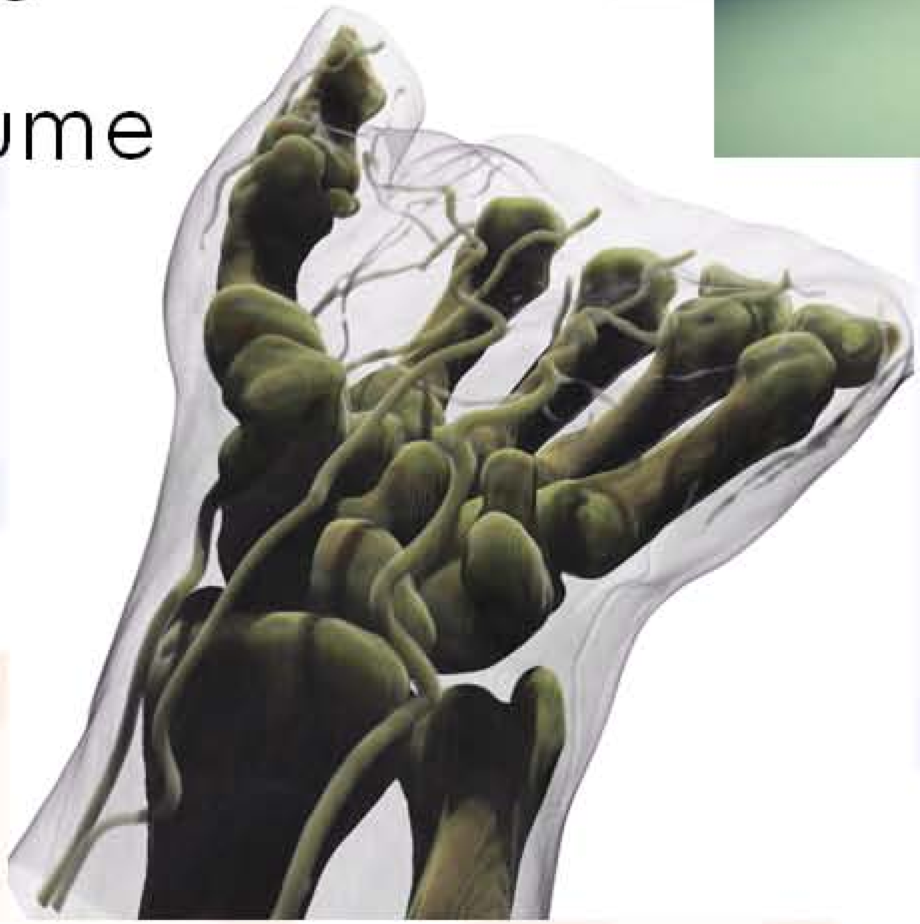
...and shadowing

- Lighting
  - Dynamic direct lighting
  - Pre-computed lighting?
- Shadowing
  - Shadow maps
  - Shadow volumes… not really

Shadow Casters and Receivers

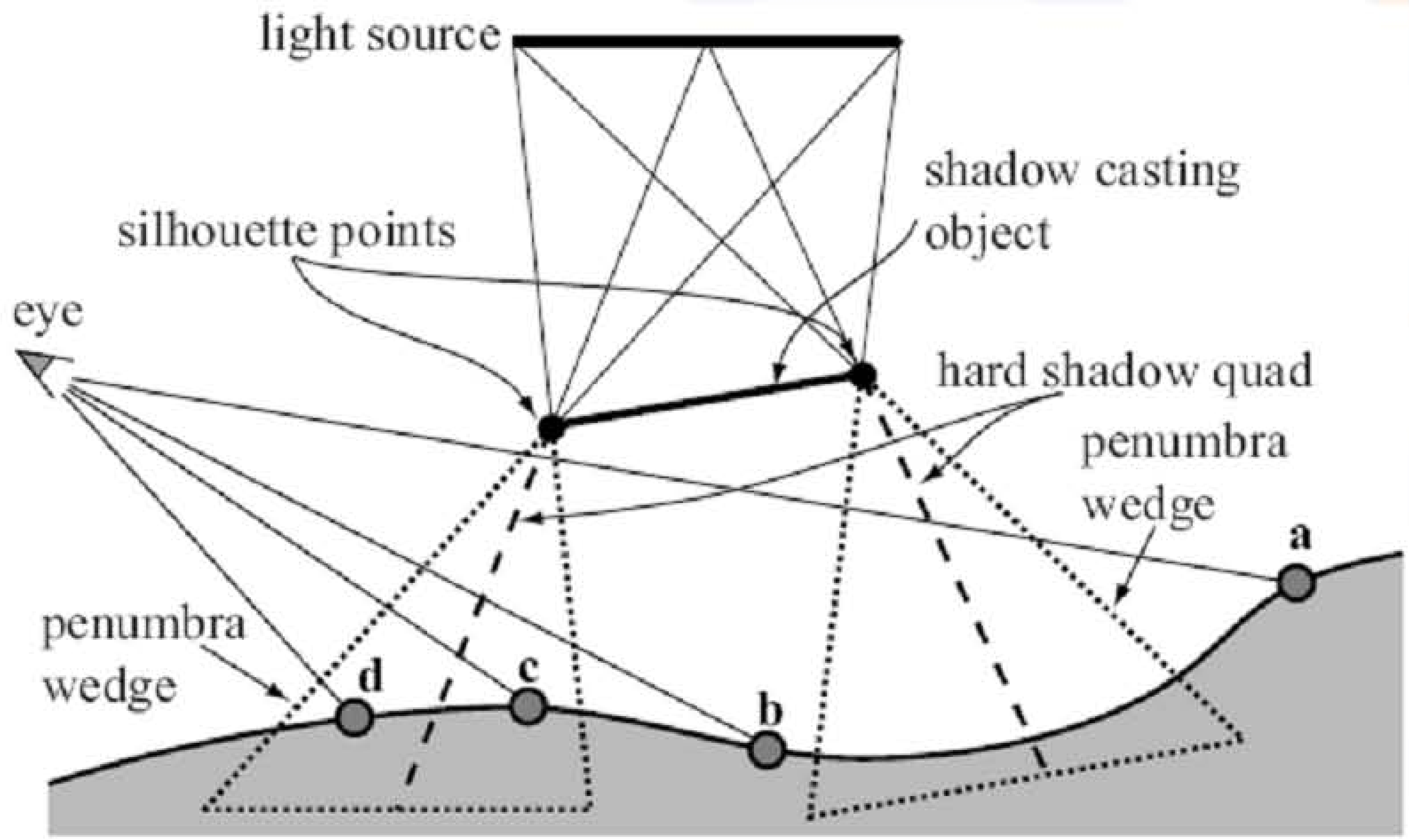
- Geometry onto geometry
- Geometry onto volume
- Volume onto geometry
- Volume onto volume
- Shadows within volume

Lokovic and Veach



Integration with Shadow Volumes

- Stencil-based shadow volumes care only for one scene depth per pixel
- Soft shadow approaches depend even more on rasterization
- So: integration extremely hard

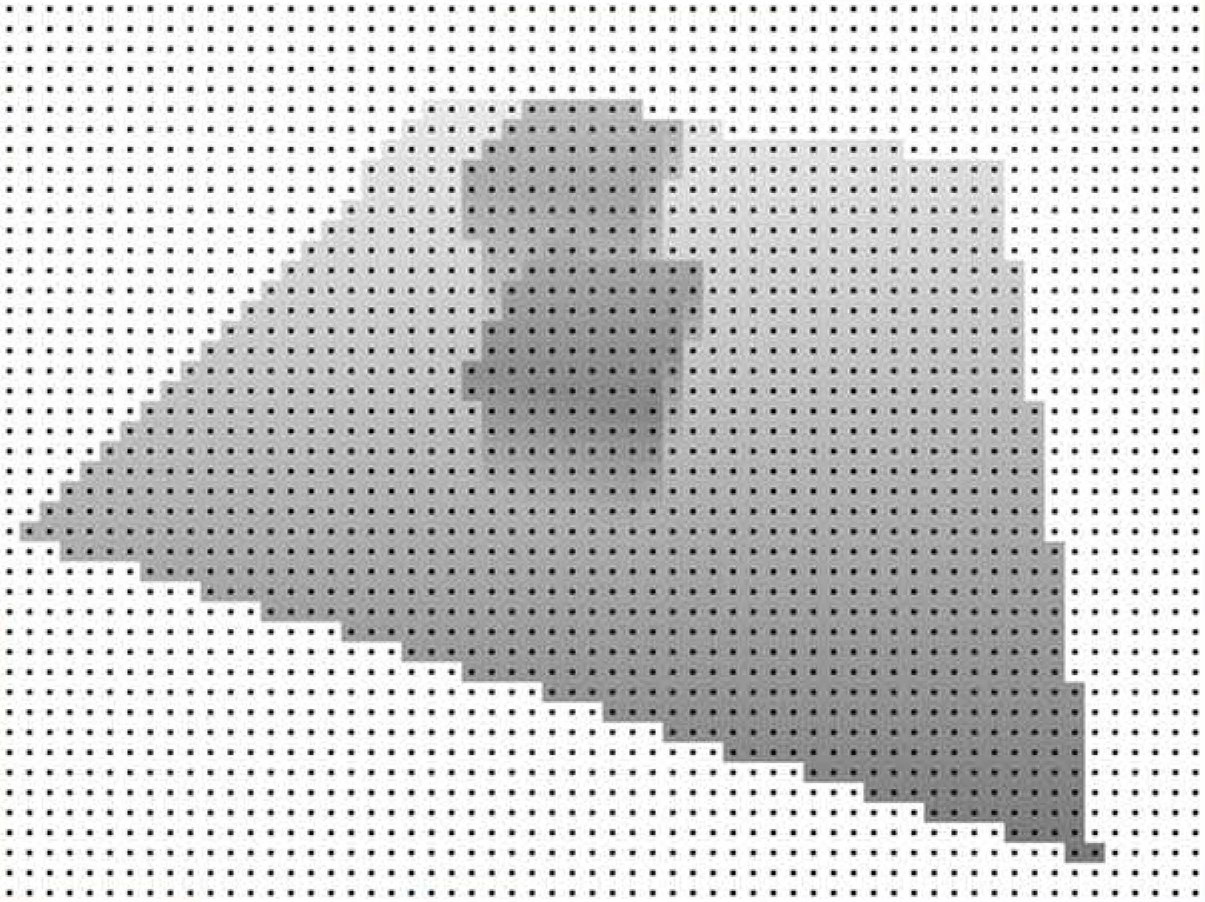


Assarsson and Akenine-Möller, Siggraph 2003

Integration with Shadow Maps

- Any depth can be tested for “in shadow or not”
- Shadows onto volumes very similar to geometry (volume is shadow receiver)

- Check each sample point inside the volume against shadow map

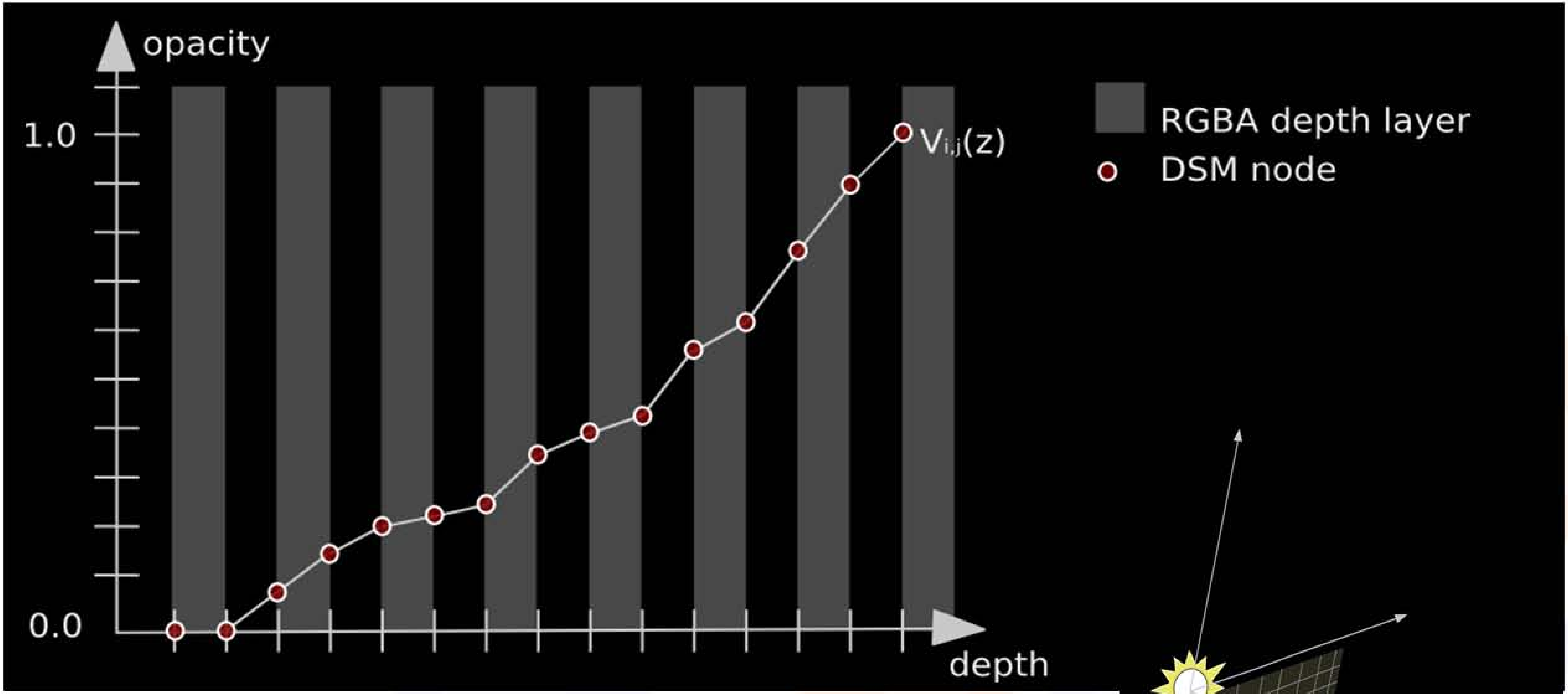


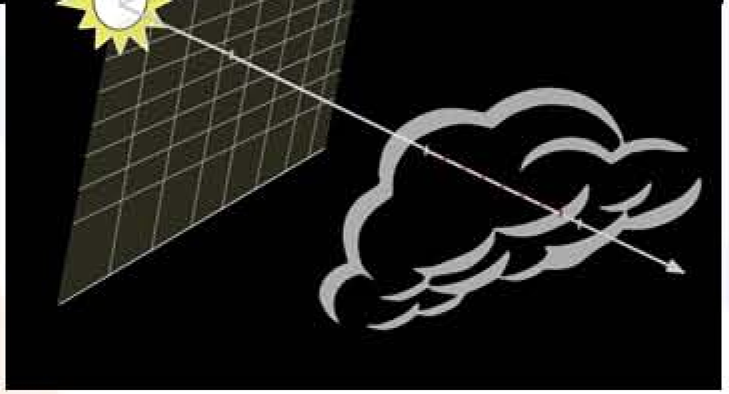
Aila and Laine

GPU Deep Shadow Maps (1)

- Presented at Graphics Hardware yesterday

GPU Deep Shadow Maps (2)

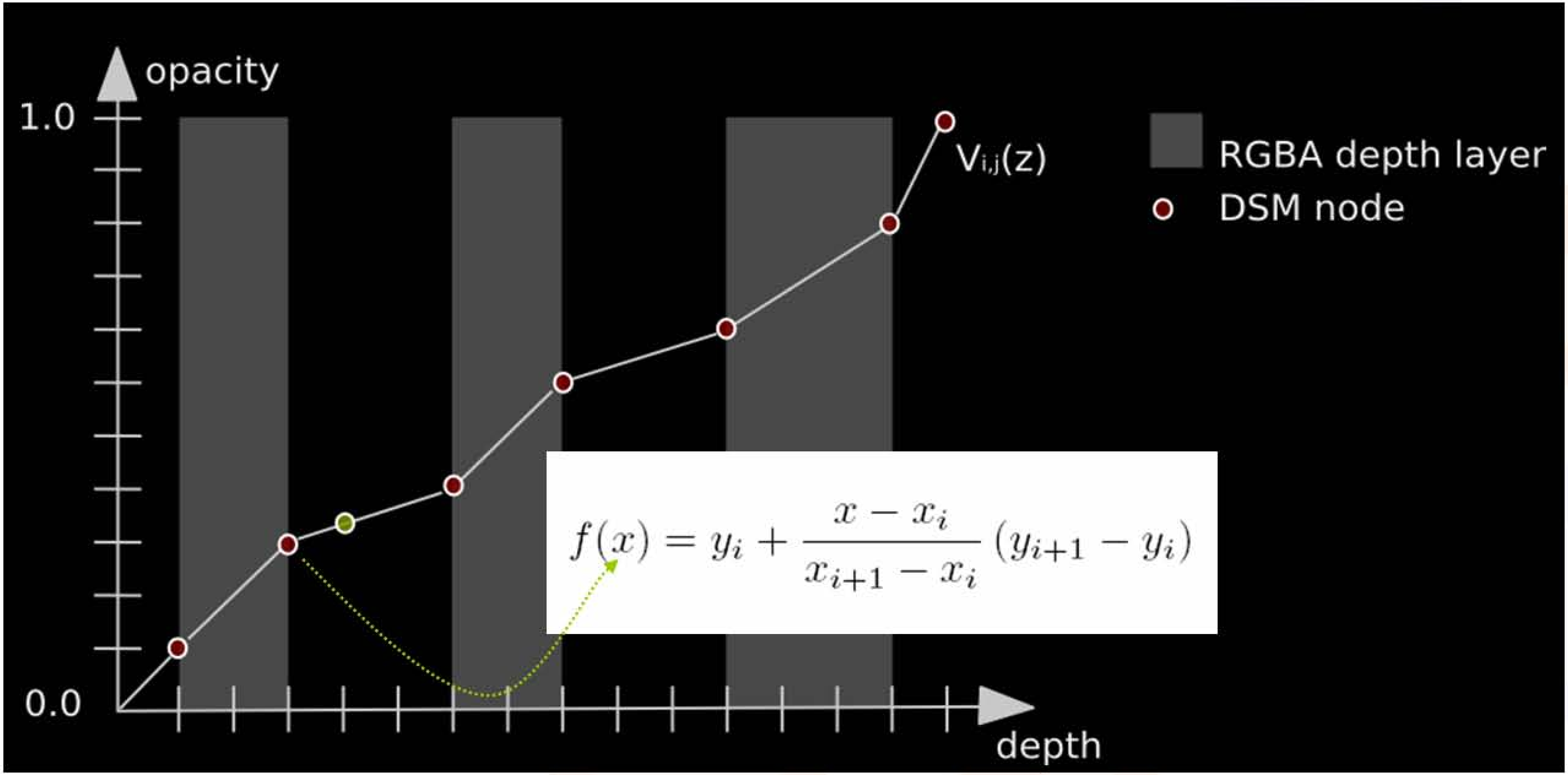




- Visibility (opacity) function
- Node := (opacity, depth) pair

GPU Deep Shadow Maps (3)

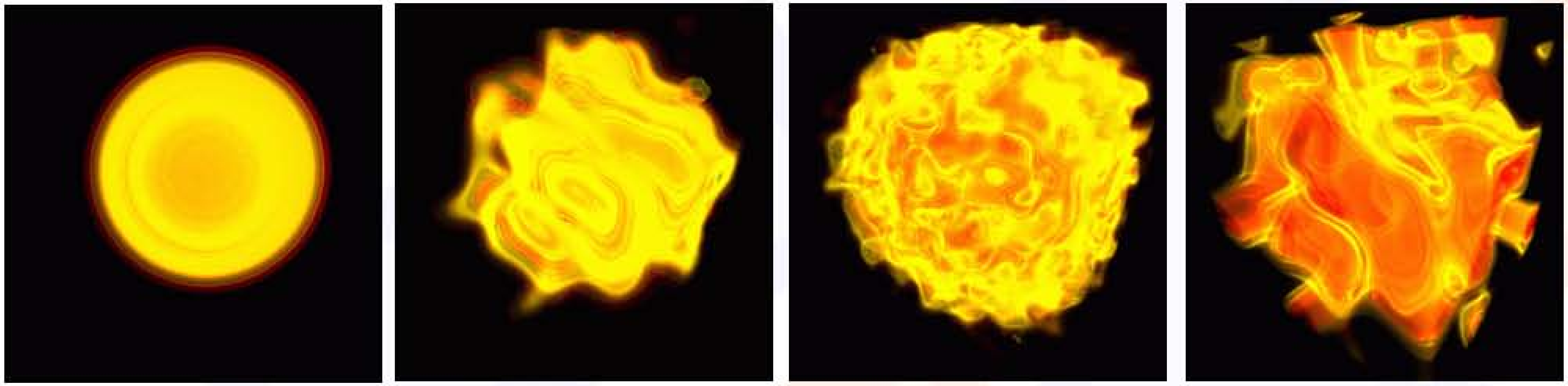
- Visibility (opacity) nodes are stored in 3D texture



$$f(x) = y_i + \frac{x - x_i}{x_{i+1} - x_i}(y_{i+1} - y_i)$$

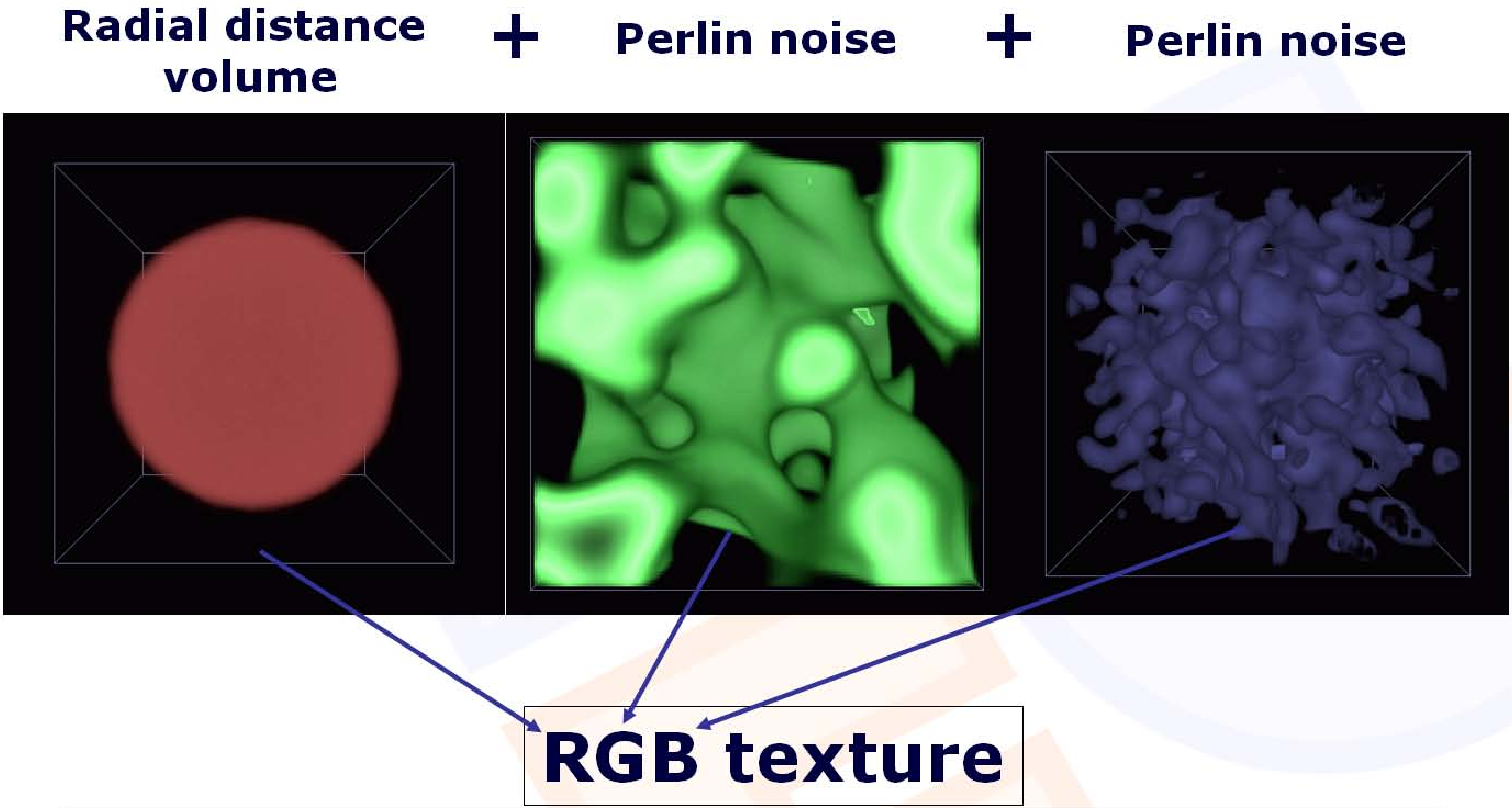
Simple Volumetric Effects (1)

- Pre-Integration + noising
  - animation:
    - change weighting through texture coords.
      => distortion of dependent lookup



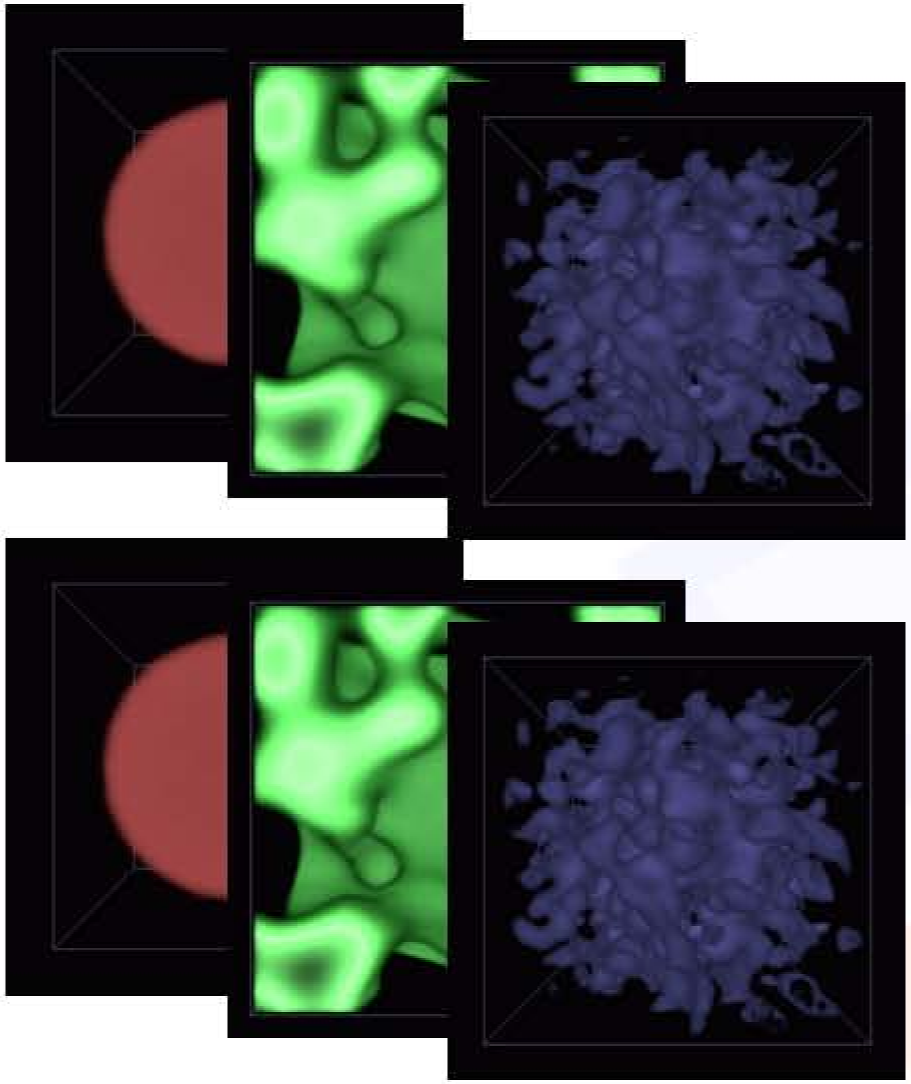
    - color cycling with transfer functions
      => outwards movement

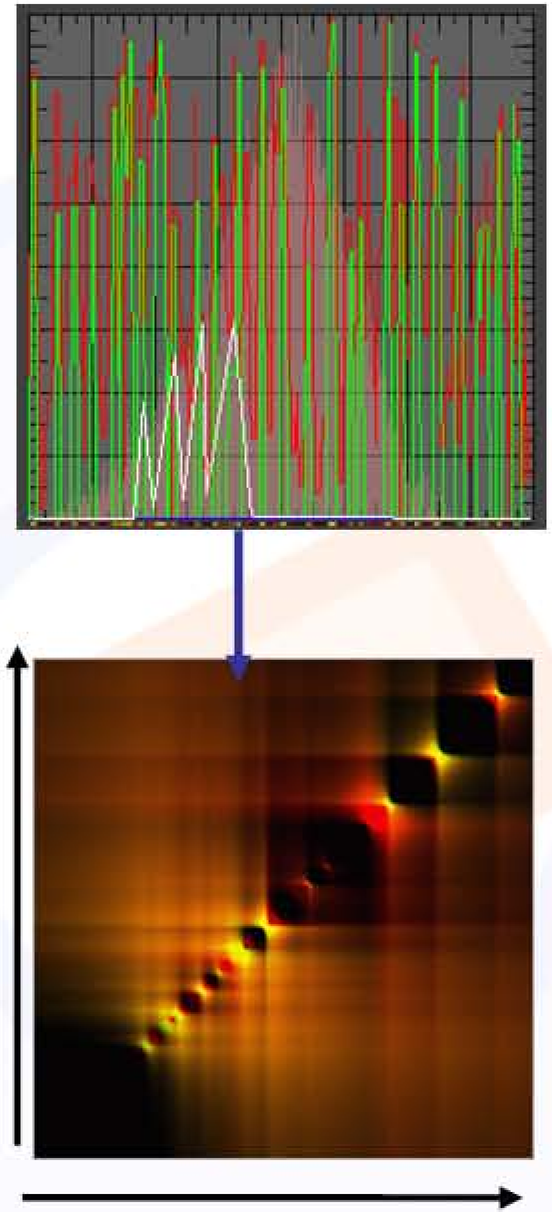
Simple Volumetric Effects (2)



Simple Volumetric Effects (3)

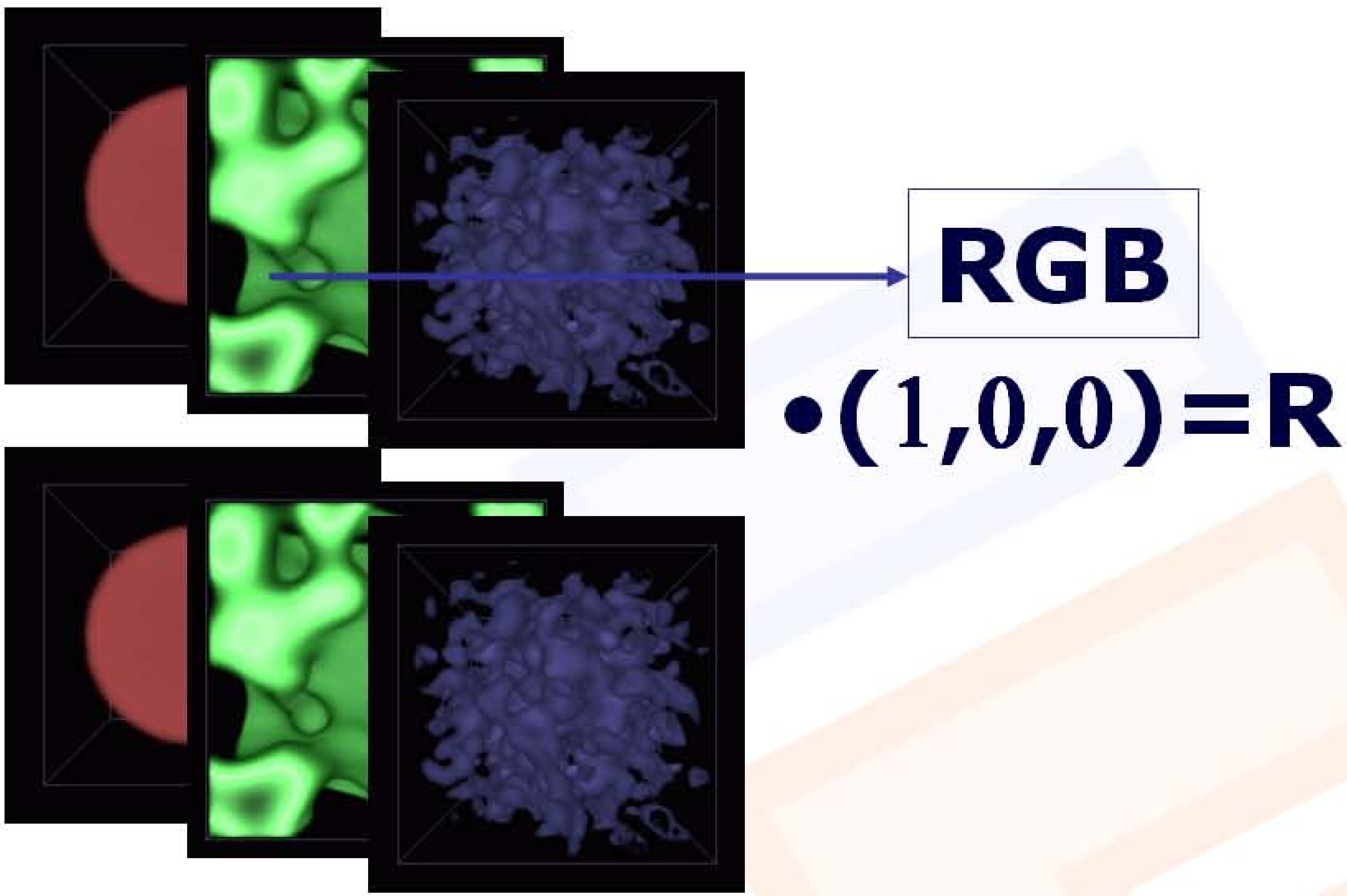
- Pre-Integration + noising
- dot-product weighting

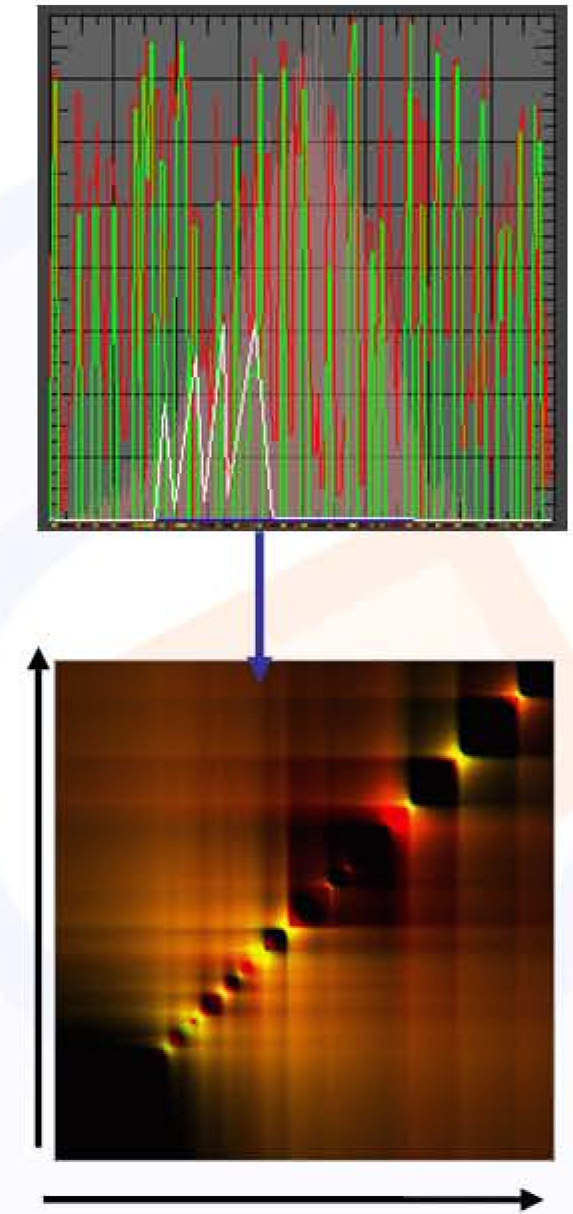




Simple Volumetric Effects (3)

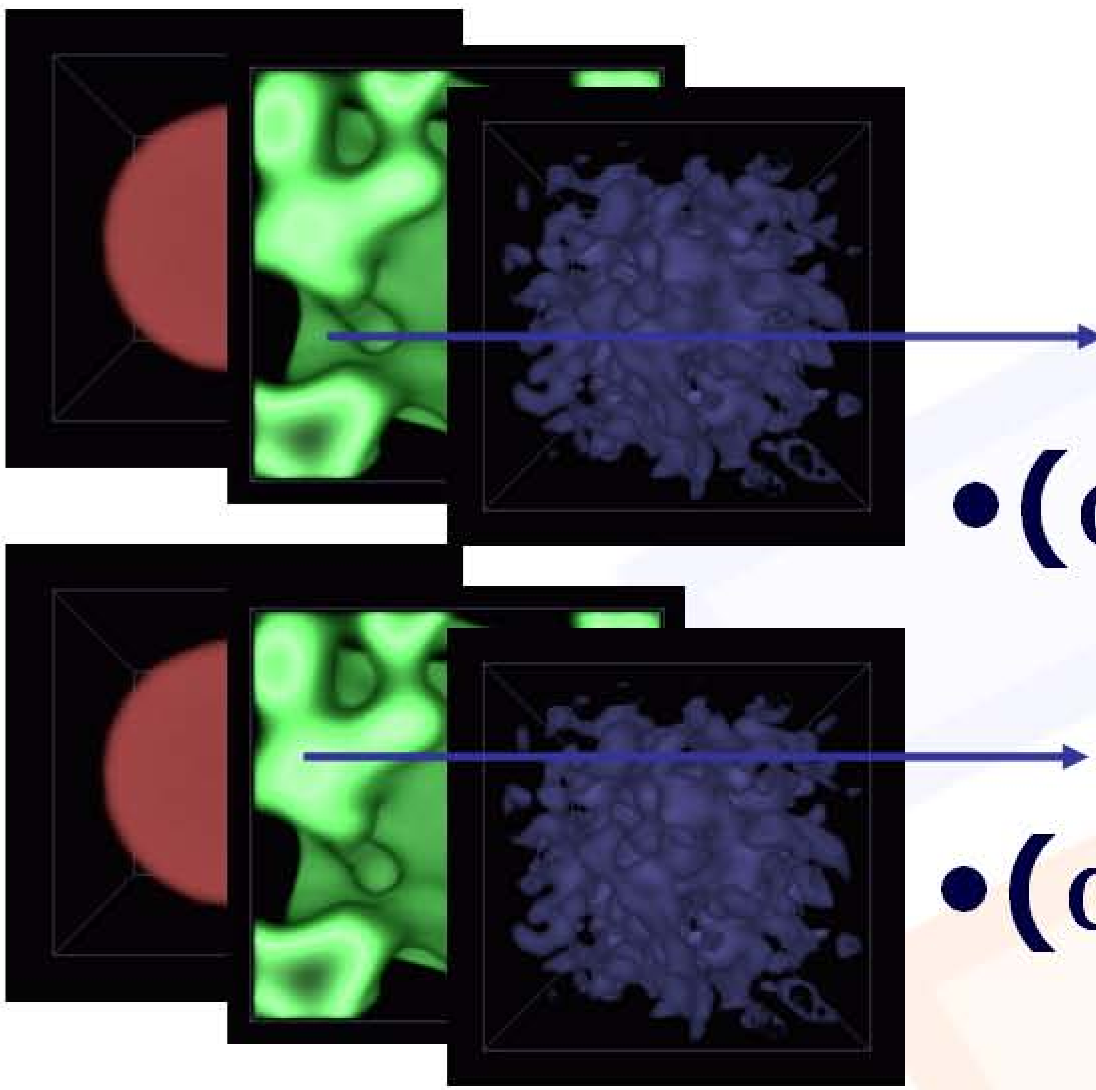
- Pre-Integration + noising
- dot-product weighting





Simple Volumetric Effects (3)

- Pre-Integration + noising
- dot-product weighting

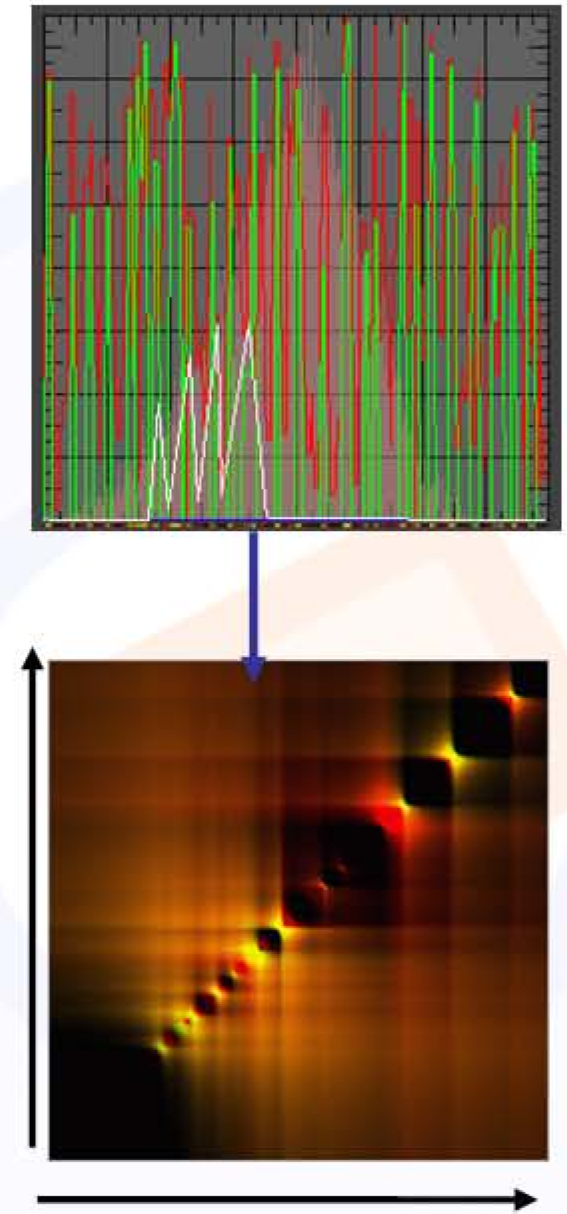


RGB

• $(\alpha,\beta,\gamma)=S_0$

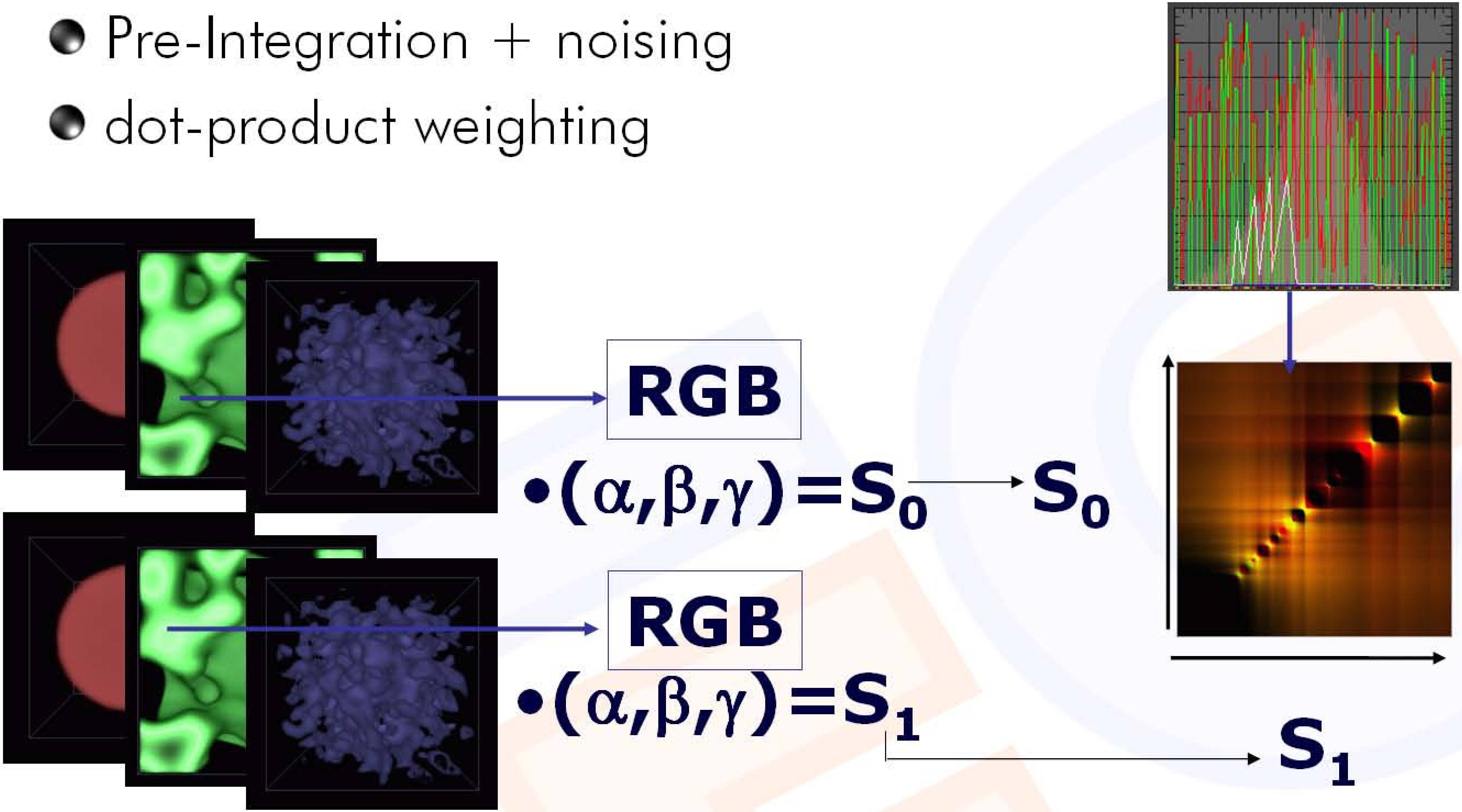
RGB

• $(\alpha,\beta,\gamma)=S_1$



Simple Volumetric Effects (3)

- Pre-Integration + noising
- dot-product weighting

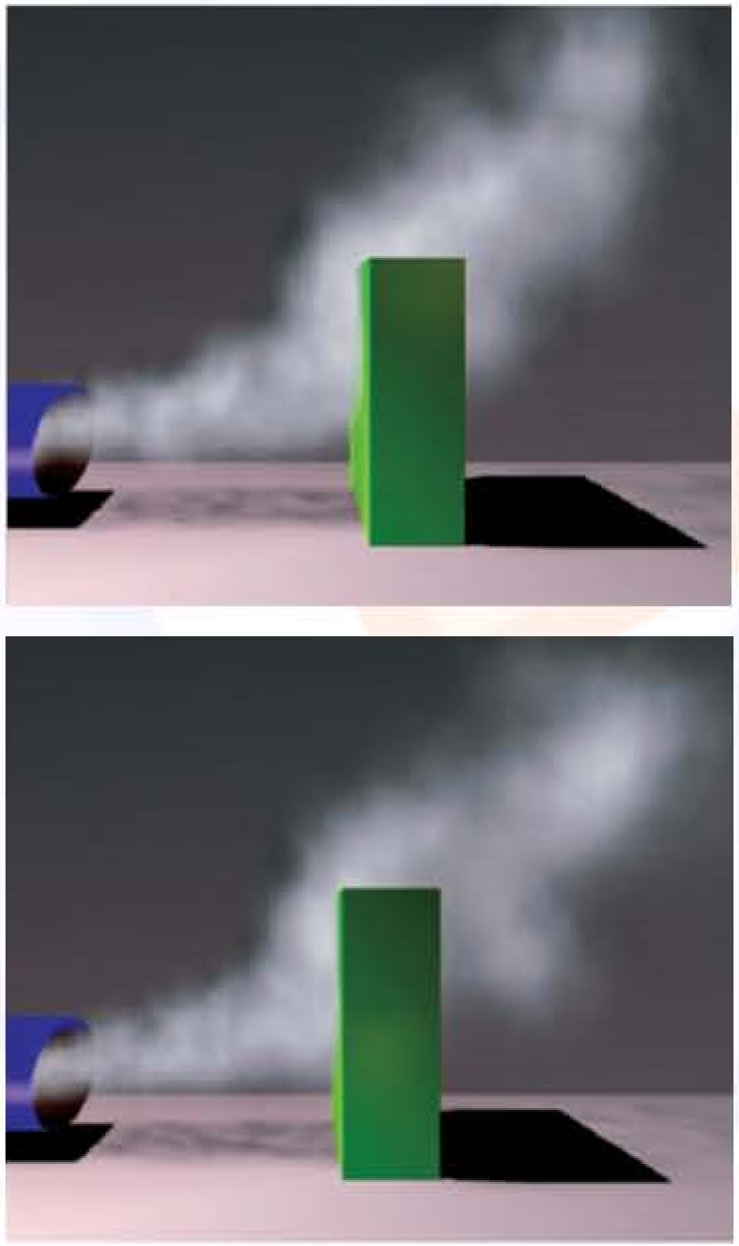


Volumetric Effects Simulation

- Procedural effects animation
- Particle systems (not really volumes)

- Incompressible Navier Stokes
- Lattice Boltzmann models (LBMs)
- Reaction diffusion

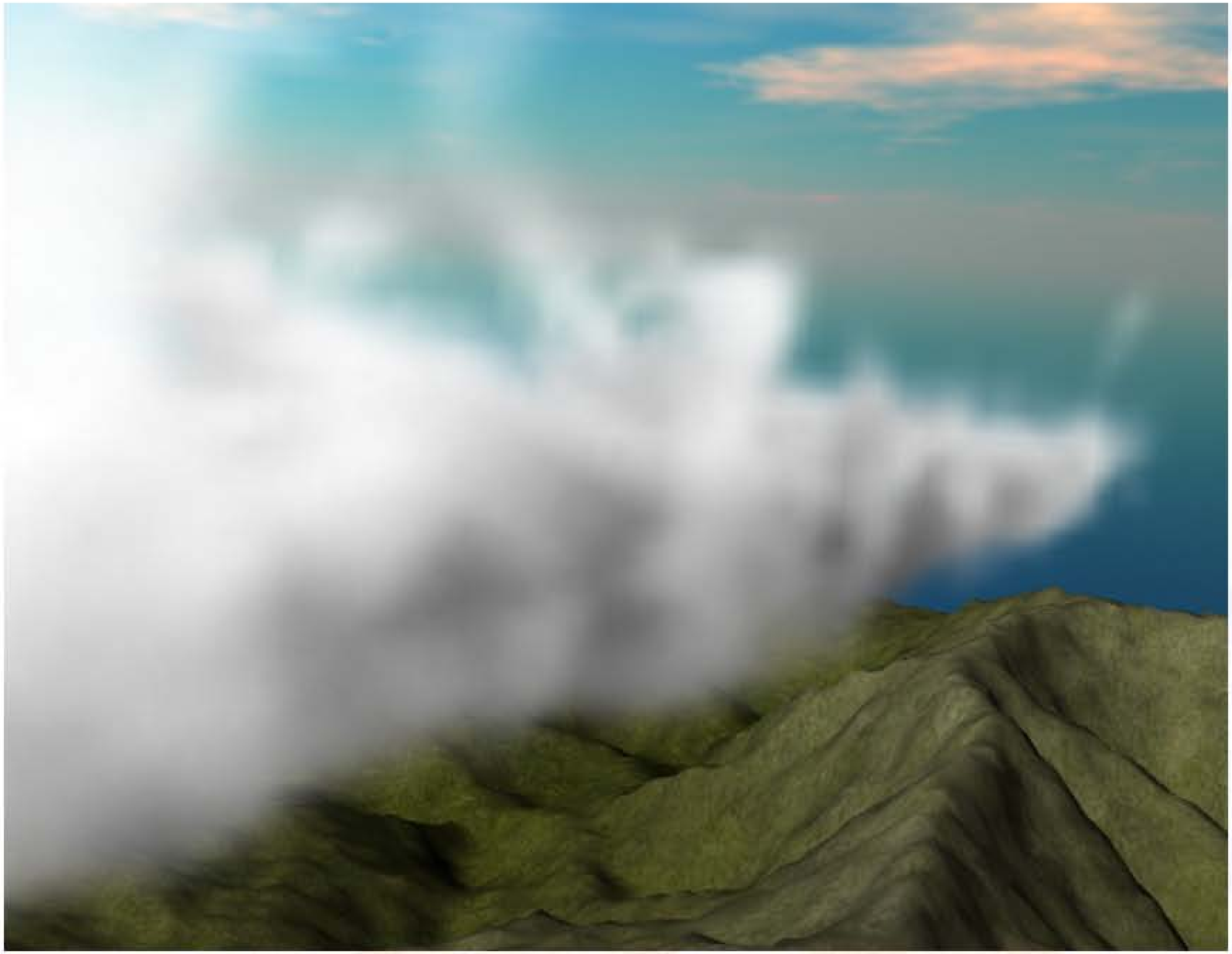
- (Pre-computed CFD solutions)



Wei et al.

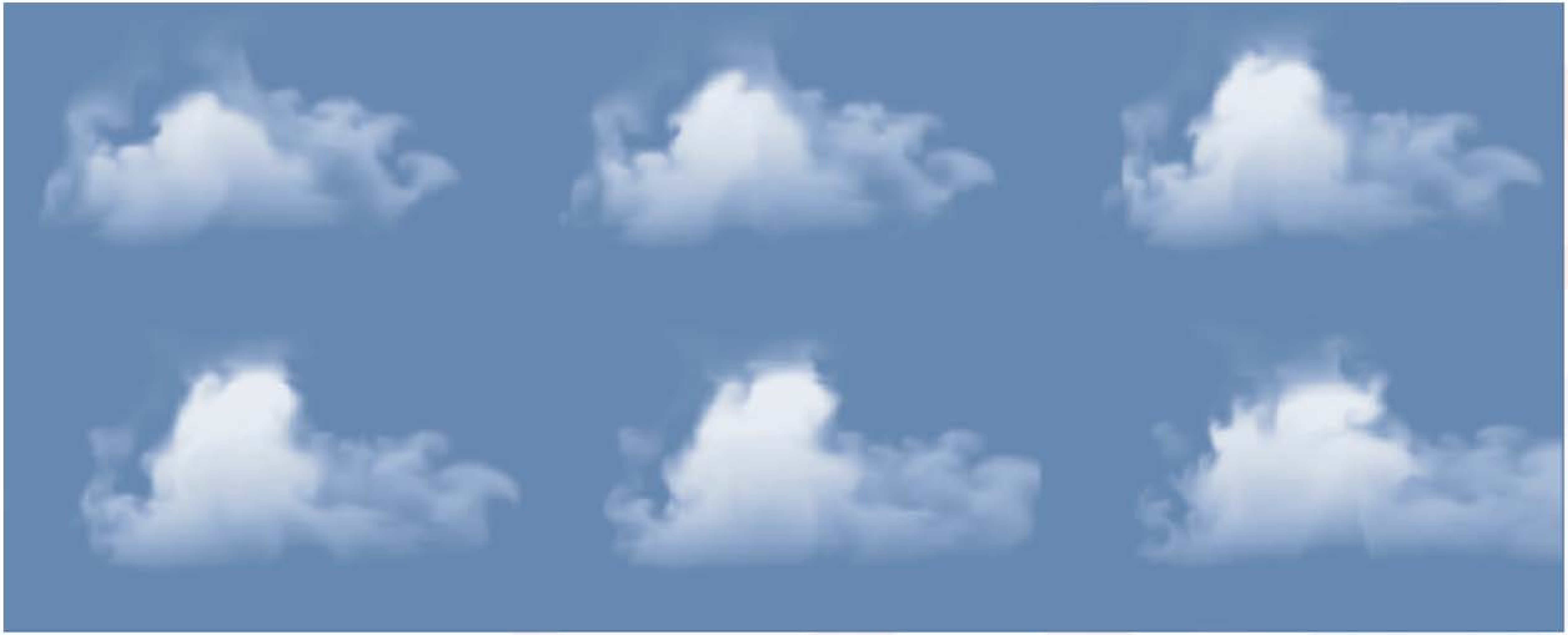
Cloud Dynamics (1)

- [Harris et al., GH 2003]



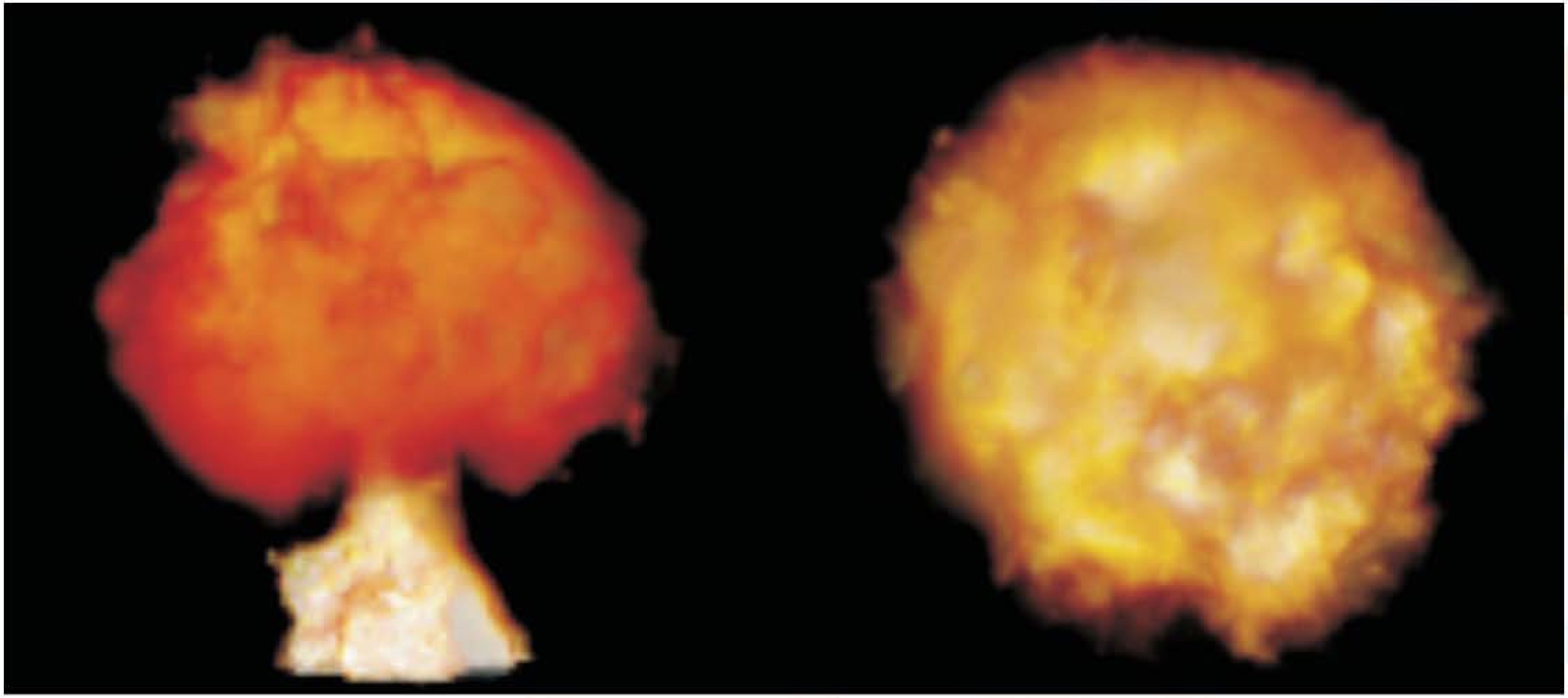
Cloud Dynamics (2)

- [Harris et al., GH 2003]



Volumetric Game Effects Framework

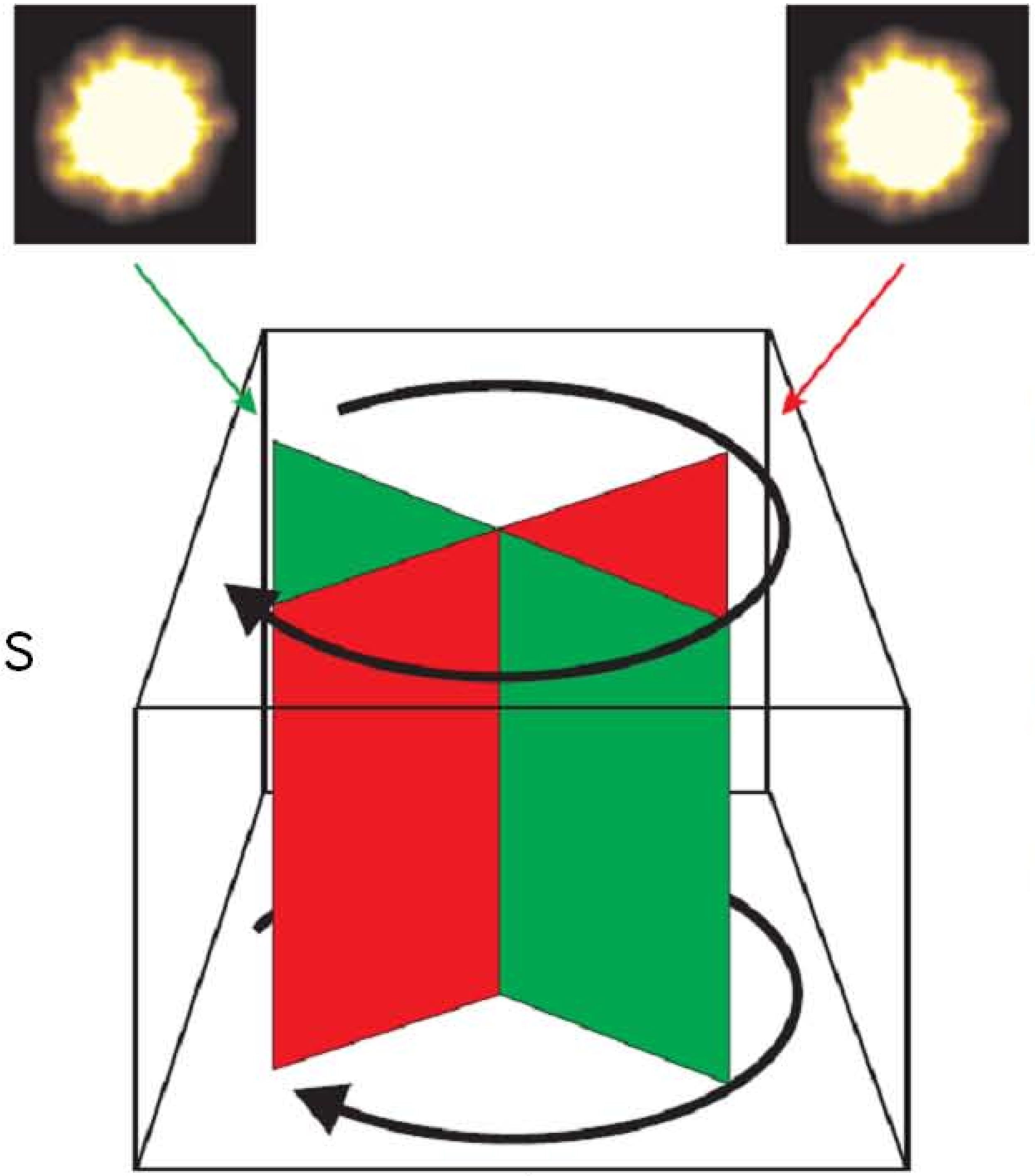
- [Krüger and Westermann, EG 2005]
- Simulate effect on 2D grid
- Turn into 3D volume on-the-fly during rendering



Krüger and Westermann

Extrusion from 2D Simulation

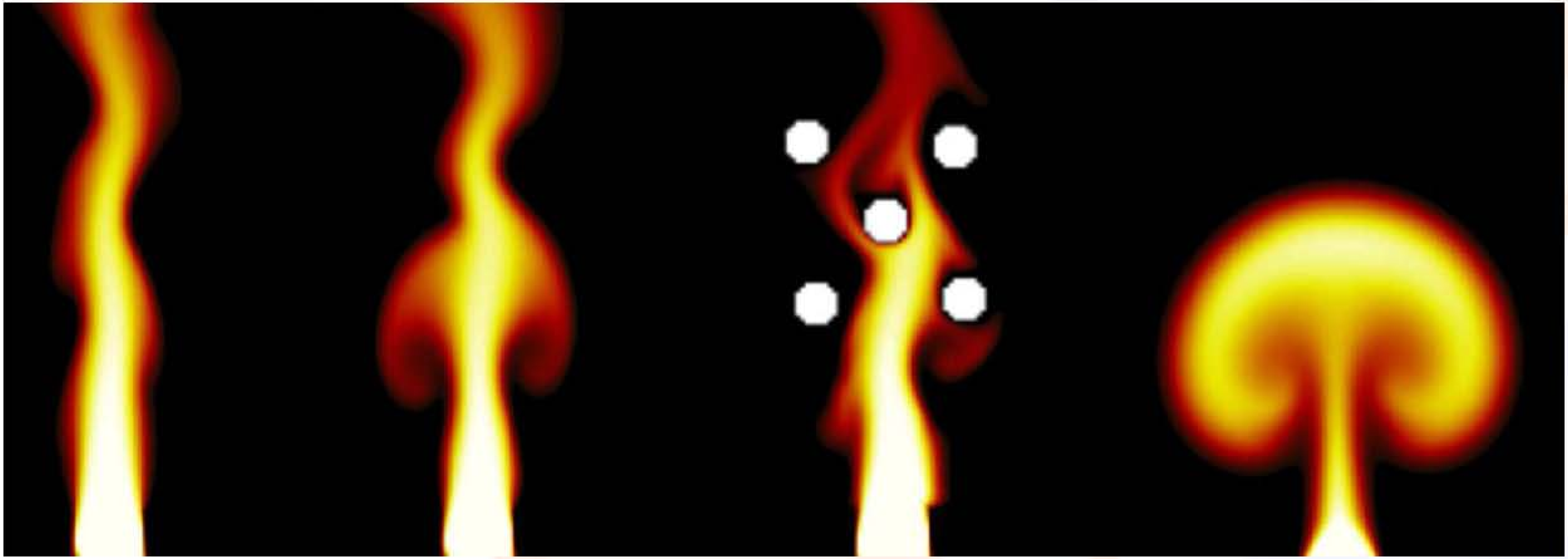
- Simulate on 2D grid
- Sample rotated version directly during ray-casting
- Simple texture coordinate computations



Krüger and Westermann

Flow Simulation (1)

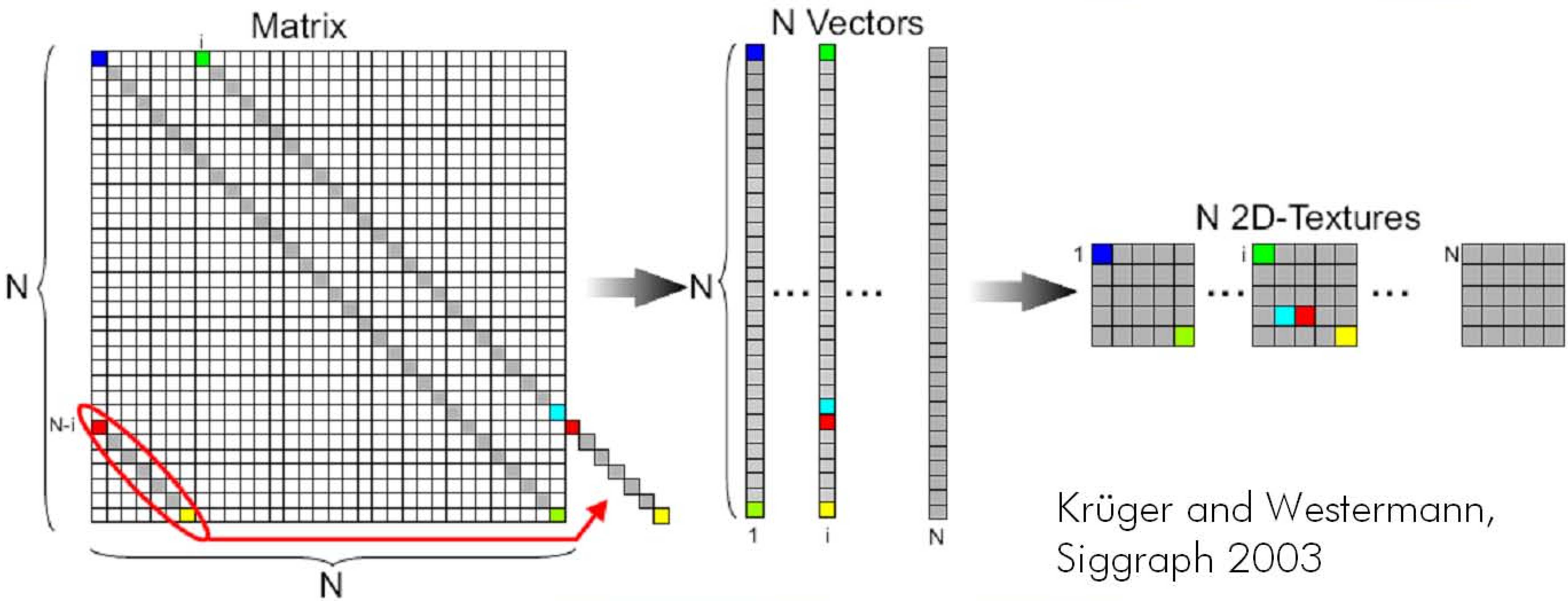
- Solve incompressible Navier Stokes
- Use GPU matrix / linear systems solver



Krüger and Westermann

Flow Simulation (2)

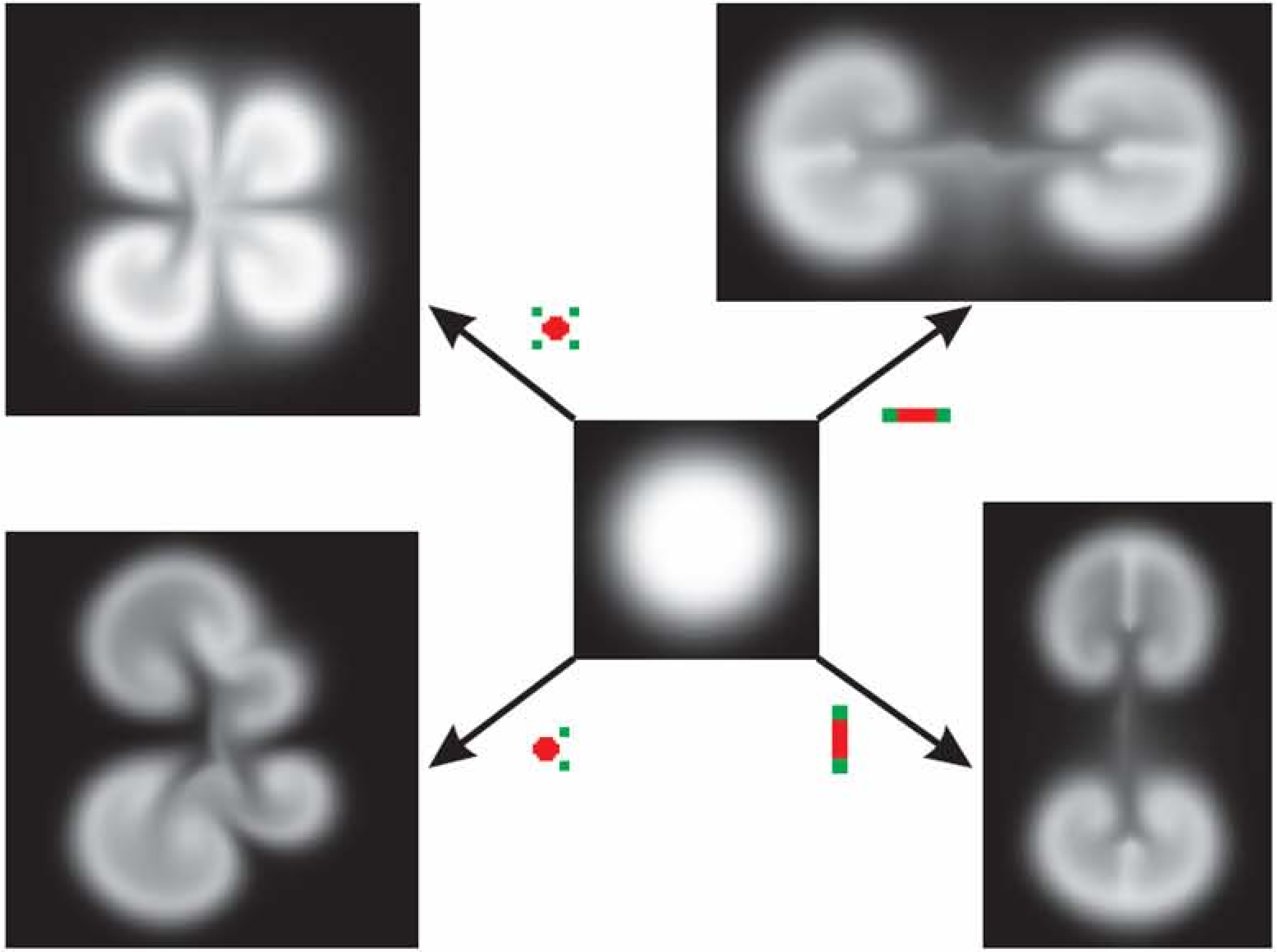
- Matrices: sets of vectors; vectors stored in textures
- Linear algebra via texture multiplication/addition



Krüger and Westermann, Siggraph 2003

Velocity Field Generation

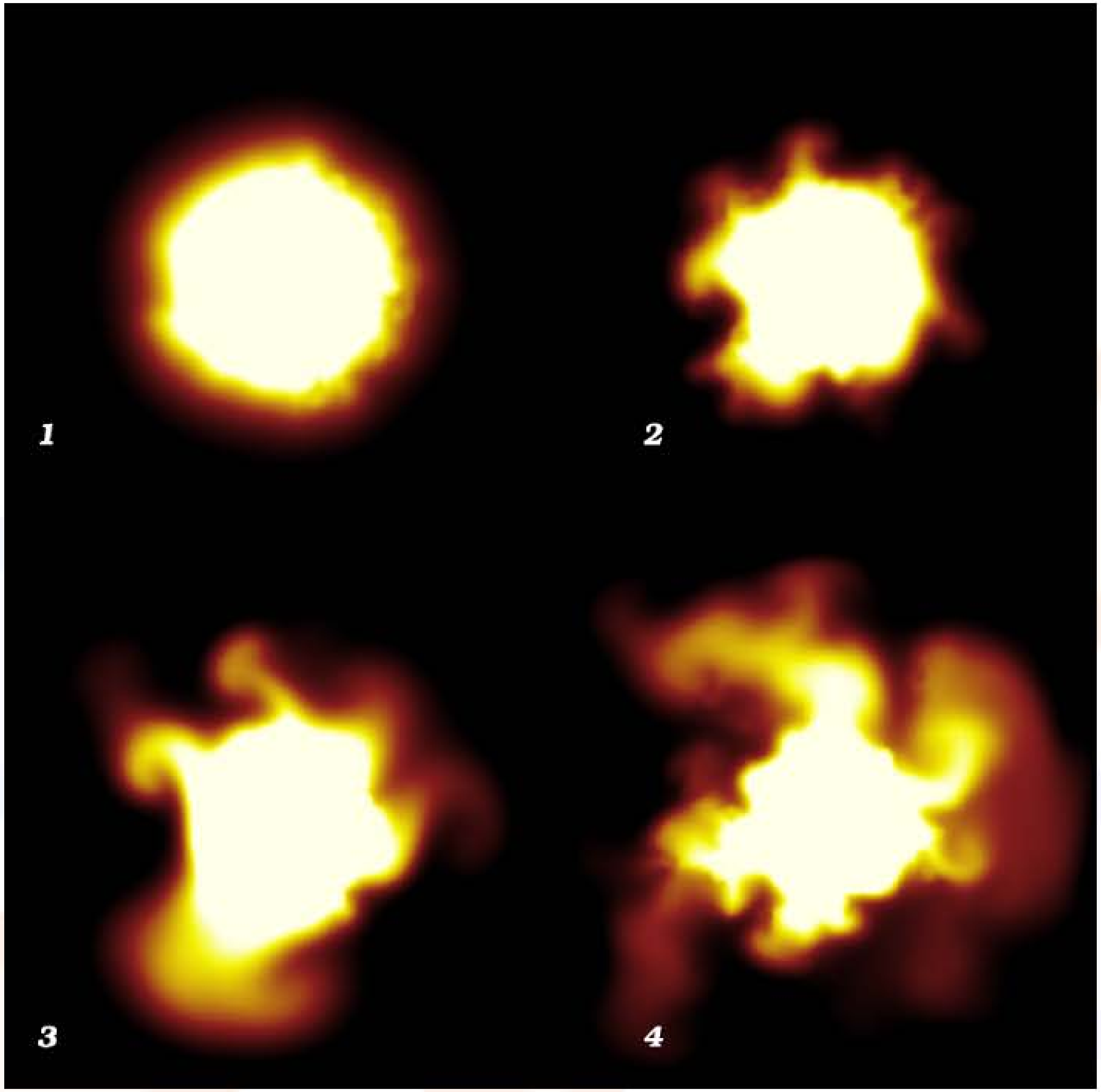
- Pressure templates



Krüger and Westermann

Pressure Templates

- f



Krüger and Westermann

Integrate Other Approaches

- Particle systems for very complicated structures
- Simulation computed on vertex buffer

Krüger and Westermann