

# *Interactive Computer Graphics*

## Lecture 16: Fire

# *Amorphous Objects*

Amorphous objects are still a major challenge in computer graphics.

Fire,

Smoke

Water

Clouds

&c.

They are the subject of current research - marketable in computer games and films.

# *Polygons*

Texture mapping polygons is fast and acceptable for short lived effects.

Overlay an flame point with a series of textured polygons (25 for a one second effect)



## *Using polygons and fixing up the lighting*

To make things more realistic we need to introduce secondary light sources at the point of the fire.



# *Limitations of Polygons*

Polygon fires are:

Difficult to sustain for for long periods

Difficult to spread or change shape

Difficult to introduce translucency

Though they are fast and therefore good for interactive graphics.

# *Modeling Fire*

Modeling fire is computationally expensive, but can produce better effects.

Flames are incandescent gases having temperature pressure and density.

The visual appearance and shape changes (due to diffusion) are functions of these.

Modeling the physics accurately is difficult

# *Particle Systems*

One solution is to approximate flame - and other amorphous objects - by a discrete set of small particles.

Particles can have:

Mass

Position

Velocity

Temperature

Shape (not often used)

Lifetime

# *Particle Creation*

Particles can be created according to a probability distribution.

They can be given an initial velocity and a lifetime

Depending on the simulation their movement can be determined by dynamics.



## *Example - (water not fire!)*



New particles created each frame  
the number created per frame is  
normally distributed

Each particle has an initial  
downward velocity

again normally distributed

Each successive frame each particle  
is acted on by a “wind” force to the  
right

# *Particle Dynamics*

Newtonian particles:  $\mathbf{f} = m \mathbf{a}$  (NB  $\mathbf{f}$  and  $\mathbf{a}$  are vectors)

$$\mathbf{a} = d\mathbf{v}/dt = d^2\mathbf{x}/dt^2$$

Given  $\mathbf{f}$  we need to find the change in position  $\mathbf{x}$  so we need to integrate.

But since we are working in frame intervals we can use a simple approximation.

$$\mathbf{v}_{t+1} = \mathbf{a}_t \Delta t + \mathbf{v}_t$$

$$\mathbf{x}_{t+1} = \mathbf{v}_t \Delta t + \mathbf{x}_t$$

Accuracy is not as important as effect!

# *Practical particle system algorithm*

for each video frame

{ generate new particles;

remove old particles;

for each particle

{ resolve forces by vector addition;

calculate  $\mathbf{a}$ ,  $\mathbf{v}$ ,  $\mathbf{x}$

apply rendering algorithm;

}

}

## *Enhancements - 1*

Introduce damping:

$$\mathbf{f} = m\mathbf{a} + s\mathbf{v}$$

$s$  is a scalar constant called damping or friction

for a gas it relates to the viscosity

An easy enhancement

## *Enhancements - 2*

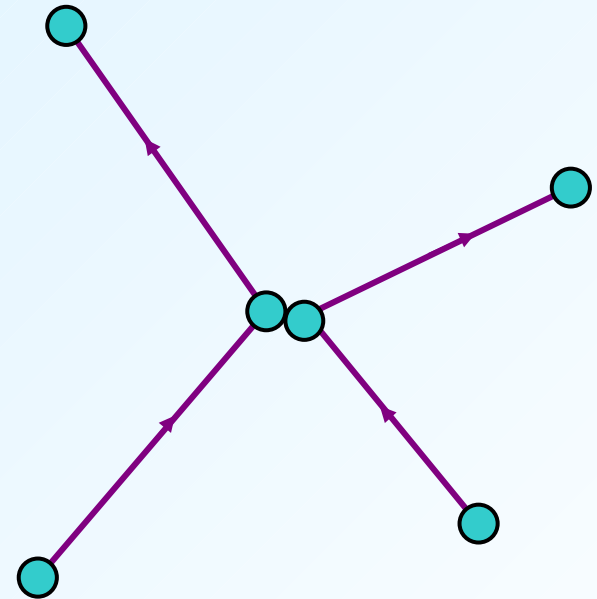
Allow particles to collide

Problems:

Sub frame calculation required

Particle shape and size needed

Large number of particle/particle  
checks needed



# *Simple Rendering*

Project particles to the view frame, then

1. Single point for each particle
2. Blob for each particle
3. Streak for each particle  
(line of motion during the frame)
4. Blend particle projection with the raster  
(translucency)

# *Rendering*

## Particle Colour

1. Make colour particle age dependent  
(simple to implement)
2. Make colour temperature dependent  
(requires modeling the temperature)

## *The wrath of Khan*

Strong contender for the worst film of 1982.

First use of a particle system to model fire:

Reeves, (1983) Particle systems - a technique for modelling a class of fuzzy objects. *Computer Graphics (SIGGRAPH 1983)* 17(3):359-376

(the next few images are from the paper)



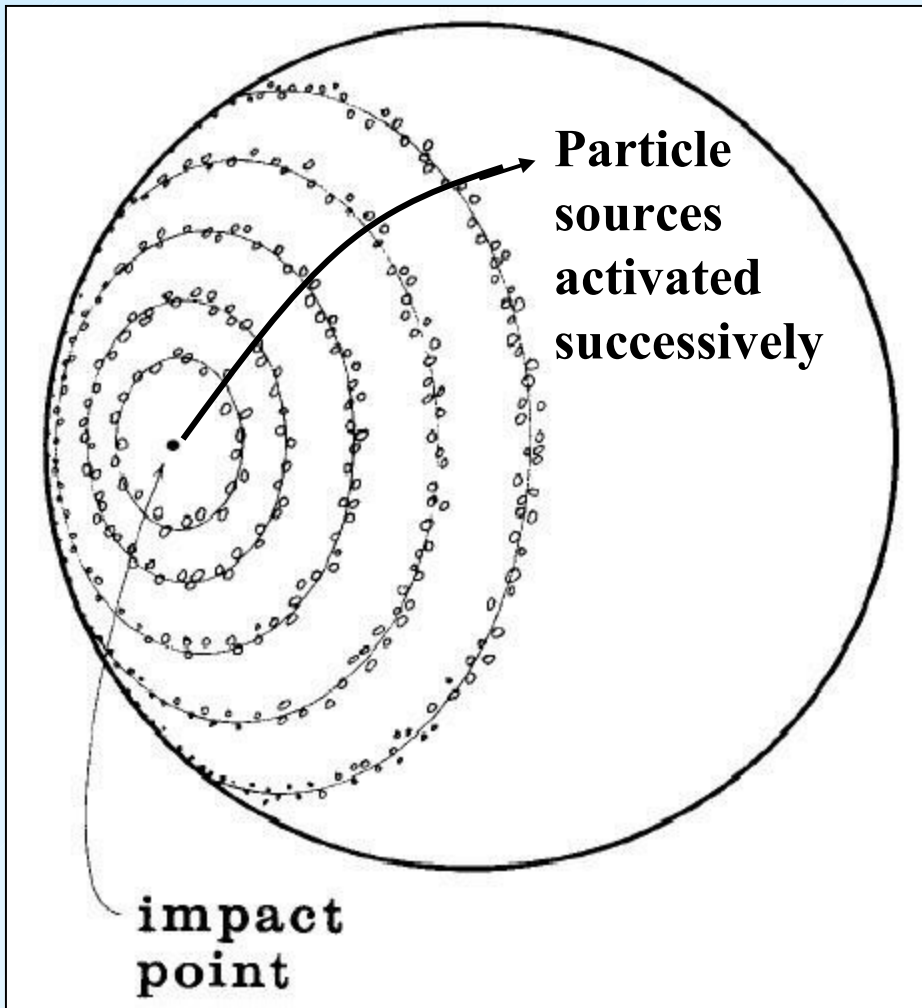
# *The Wrath of Khan - Storyboard*

The genesis project

A barren planet is to be brought to life by dropping a bomb on it.

Doesn't sound very likely to me.

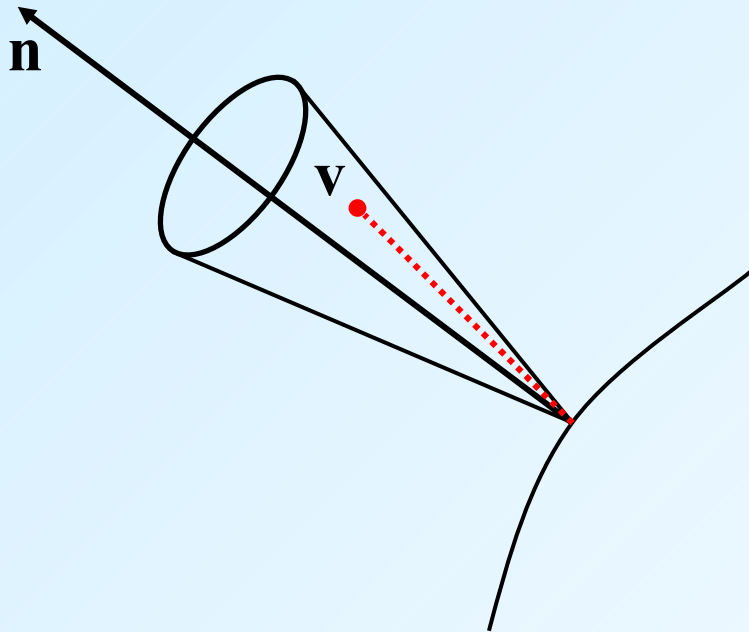
# *The Wrath of Khan - overview*



Many particle systems starting at different times to simulate the flames spreading

In total around 750,000 particles

# *The Wrath of Khan - Particle Initialisation*

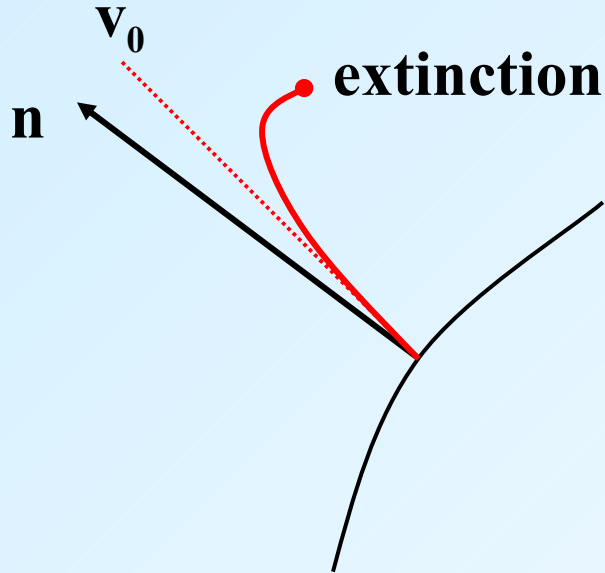


Velocity magnitude  
distributed normally

Magnitude diminishes  
with time

Velocity direction  
distributed about the  
normal

# *The Wrath of Khan - Particle Dynamics*



Gravitational force pulling the particles towards the planet.

Particle lifetime short to prevent a whole parabola being visible

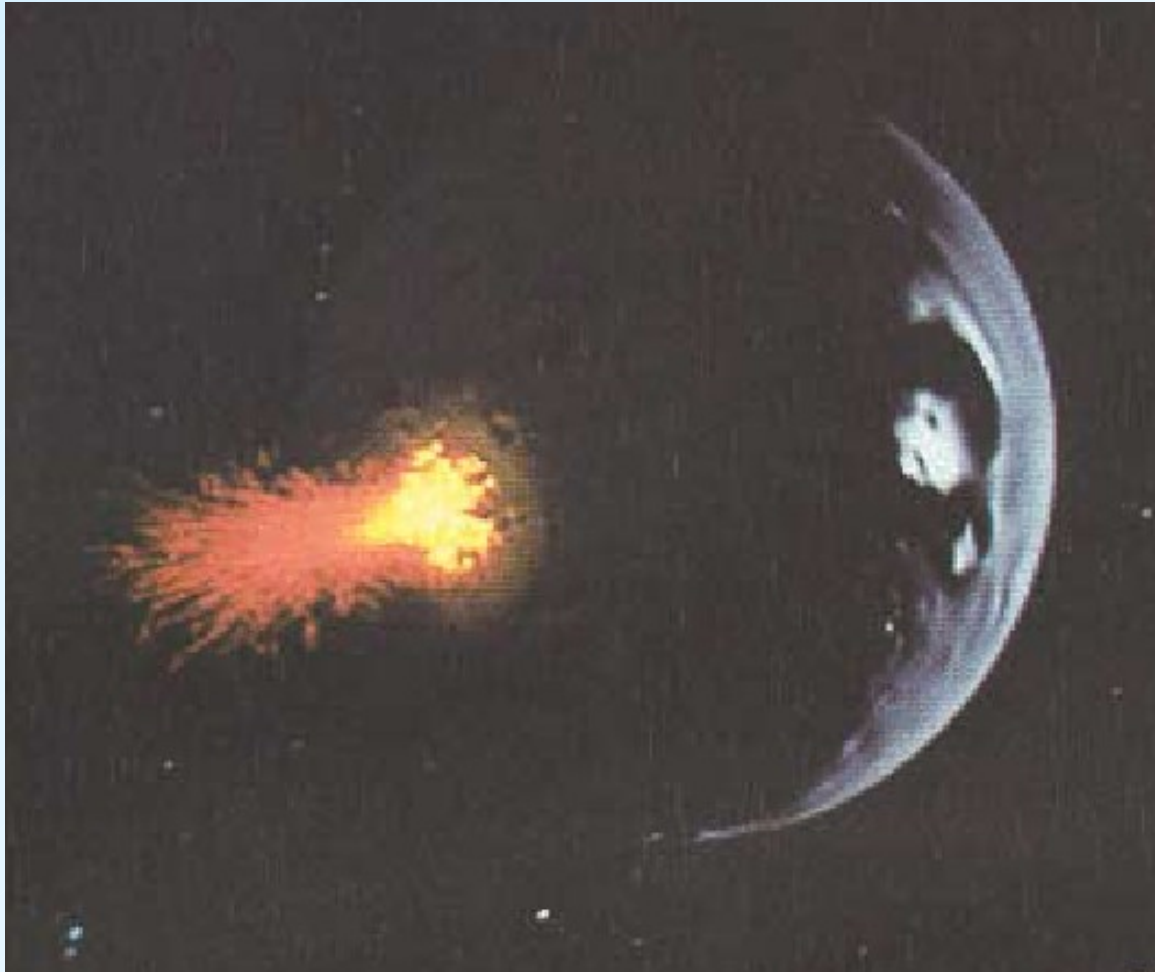
## *The Wrath of Khan- Rendering*

Planet rendered first - fake light source above the particle system to create the light emitted by the flames.

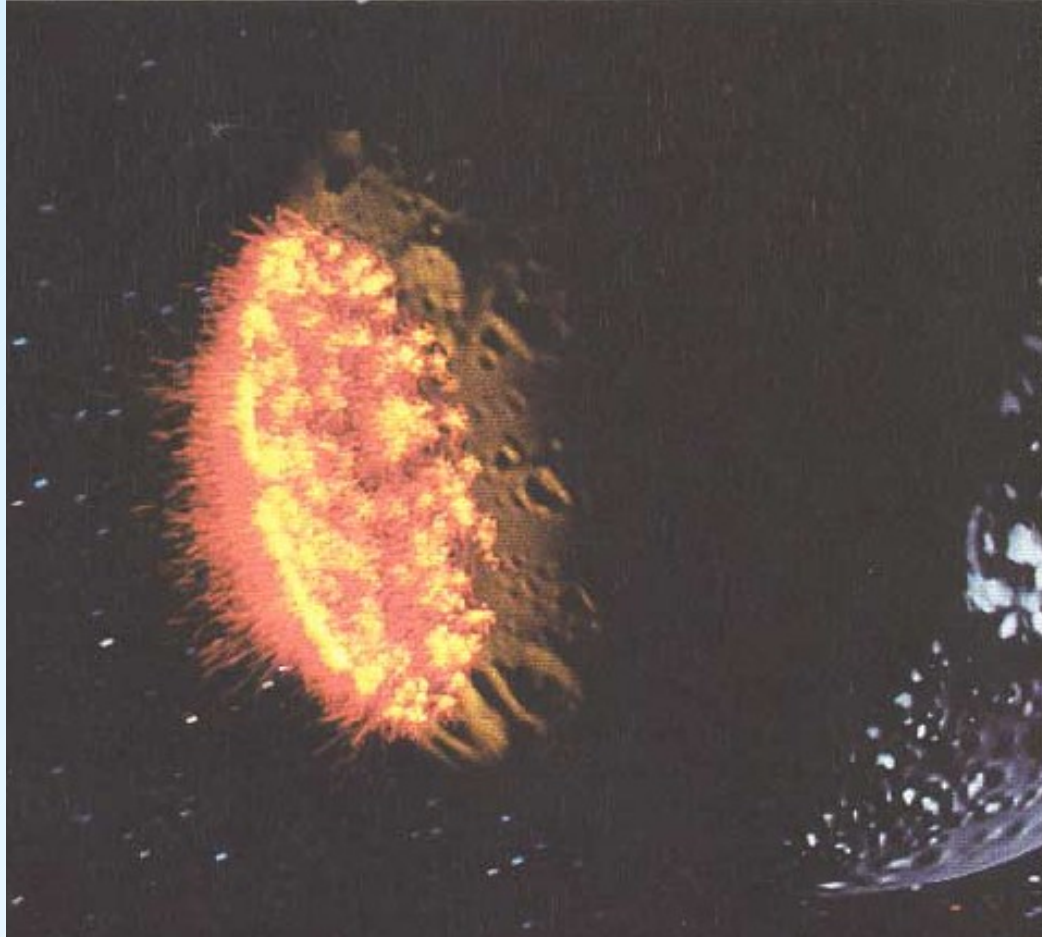
Line segment for each particle projected and drawn over the planet image, motion blur applied

Colour time dependent, yellow to red

*How did it look? Good Explosion!*



# *Effective flame spread*





*Close up it looks like the edge of a carpet!*





# *Particles in Real Time - Quake*

1. A pre-rendered  
bitmap

2. Real time  
rendered glow

3. Animated  
glowing particles



## *Where Next - Modelling*

Particle systems are very flexible modelling tools capable of simulating a lot of effects.

The particle modelling in the Wrath of Khan was simple and effective, but there might be some benefit in incorporating more features.

# *Temperature*

Gases cool as they expand - particles move slower at lower temperature.

Gas expansion is related to pressure differential.

The possibilities are endless and there have been lots of ideas tried.

## *Where next - rendering*

The real failure in the Wrath of Khan is the rendering.

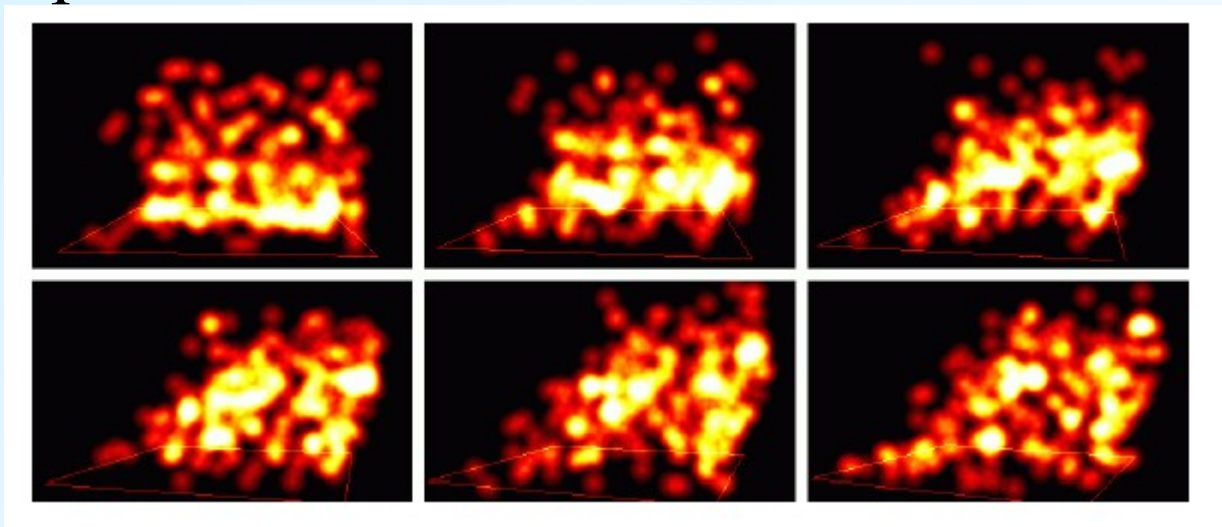
Straight line segments are not a visual characteristic of incandescent gases

The lines are very much apparent where the particle density is low

# *Bitmap Splatting*

Render the particles as splats rather than lines.

Each splat has a pattern with degrees of opacity at each pixel



Laur and Hanrahan 1991

## *Ray tracing - Blobs*

Blobs can be thought of as 3D splats

Particles are treated as the centre of blobs and the flame dynamics can be modelled by the particles

Blobs can be warped in 3D to produce a less regular appearance for the fire.

## *Ray Tracing Blobs is volume rendering*

As the ray travels through a blob it adopts a proportion of the blob colour.

It may travel through the whole particle field and take up a proportion of the scene behind

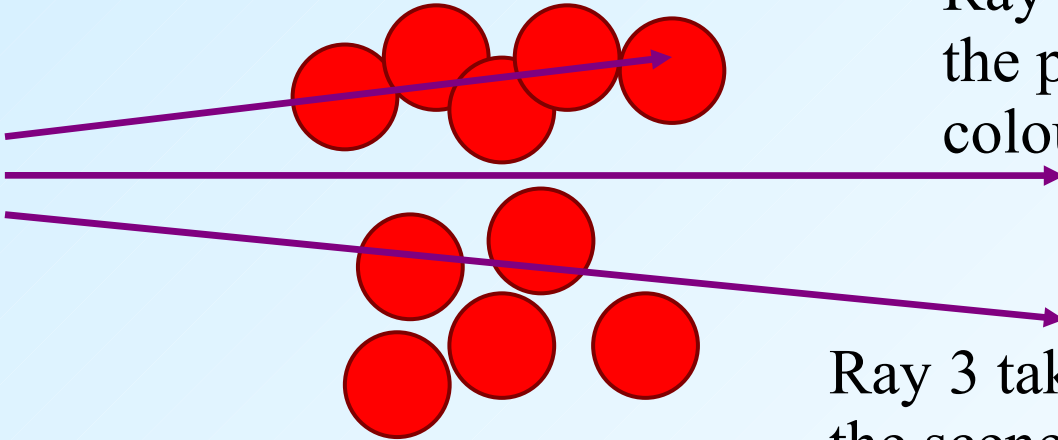
Alternatively it can terminate within the particle field in places where it is denser.

# *Ray tracing particle fields*

Ray 1 takes all its colour from the particle field. The background is occluded

Ray 2 passes unchanged through the particle field and takes its colour from the scene behind

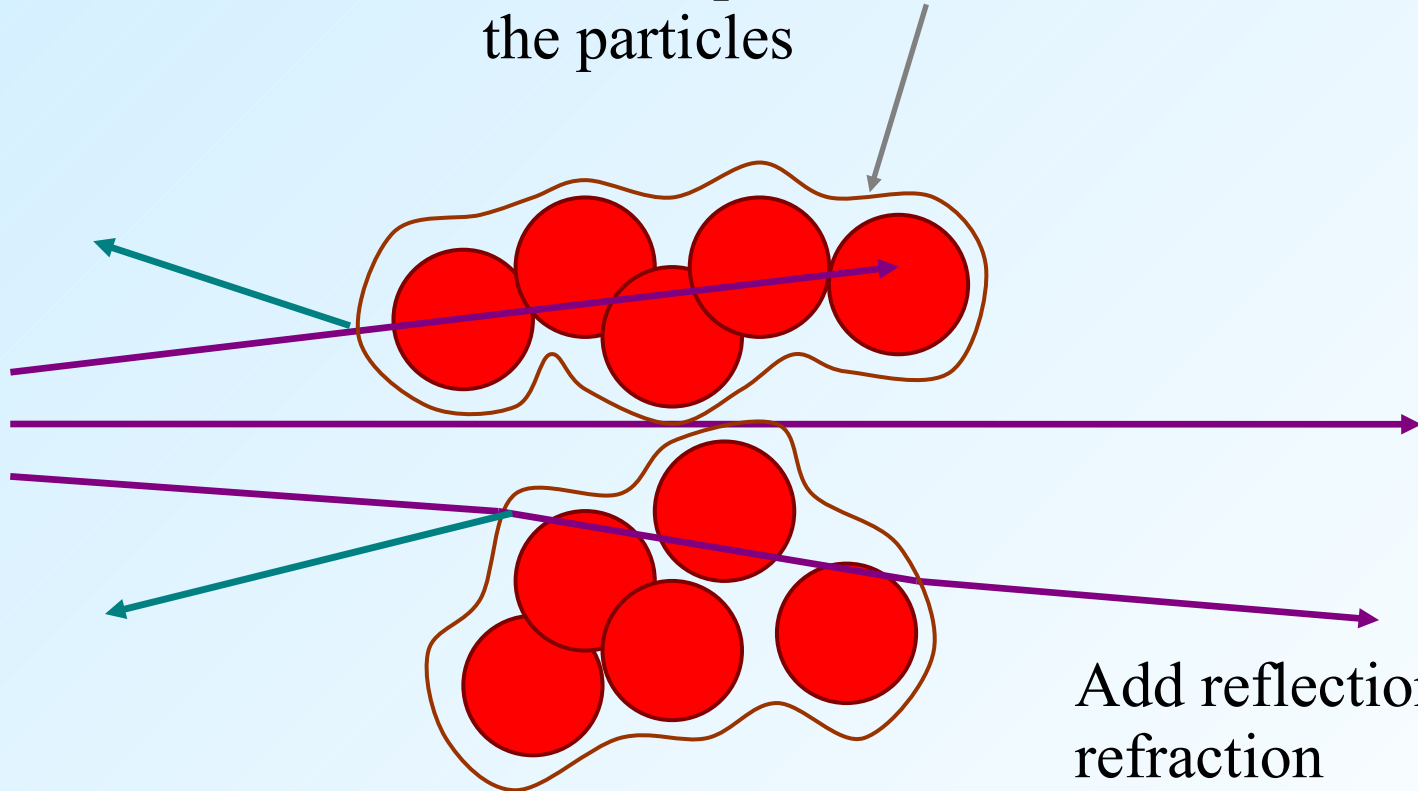
Ray 3 takes part of its colour from the scene behind and part from the particle field





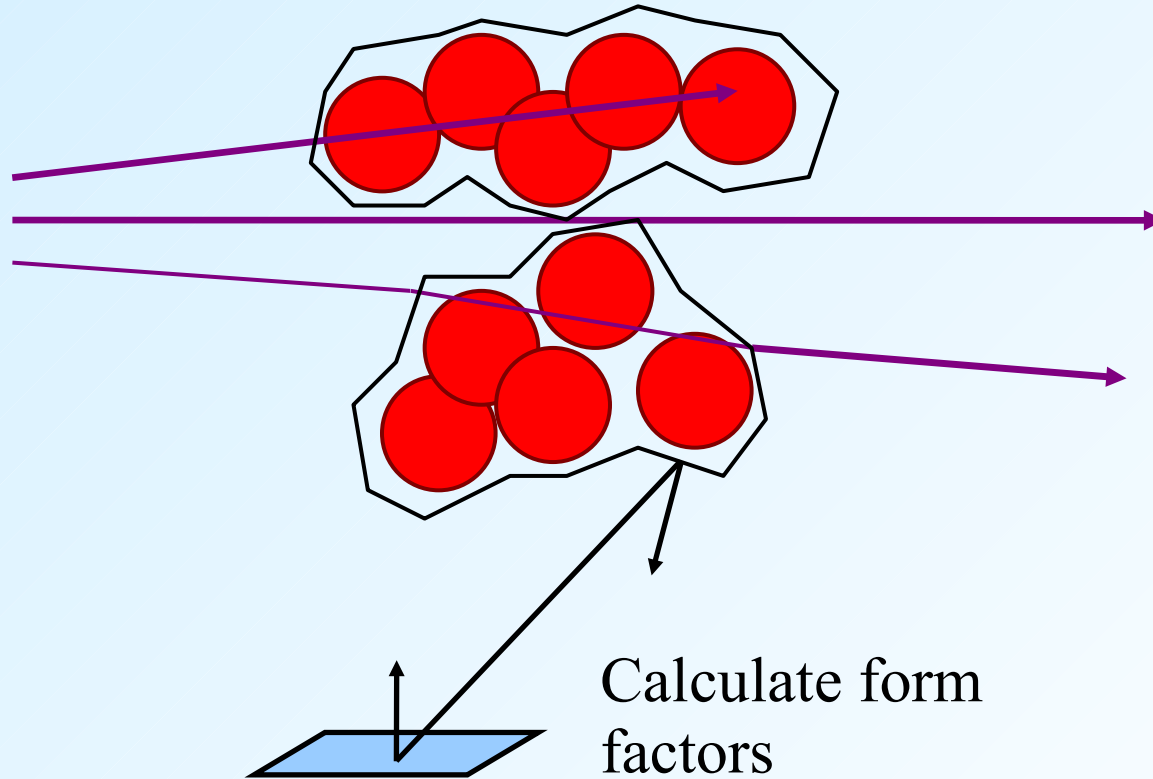
# *Further possibilities - Implicit Surfaces*

Fit an implicit surface over the particles

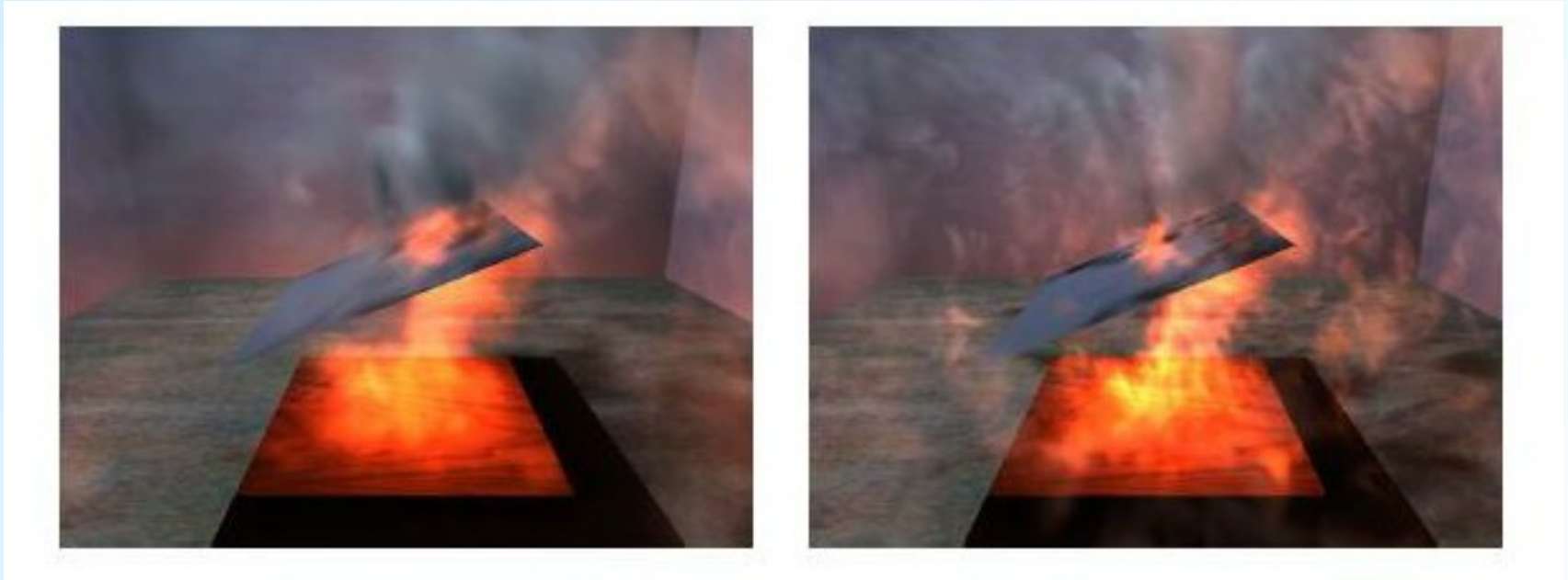


# *Further possibilities - Radiosity*

Polygonise the implicit surface



# *Flames with radiosity and blob shadows*



Engell-Nielsen and Madsen

University of Copenhagen

(<http://www.it-c.dk/~beyond/Animations/index.html>)