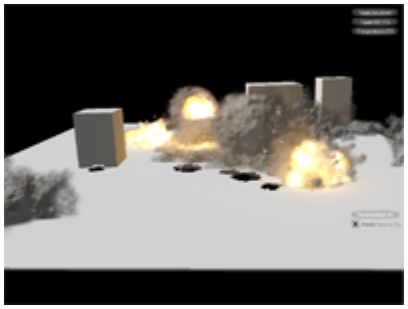
DeferredParticles Sample

Collapse All



Path

Source	SDK root\Samples\C++\Direct3D10\DeferredParticles
Executable	SDK root\Samples\C++\Direct3D10\Bin\x86 or x64\DeferredParticles.exe

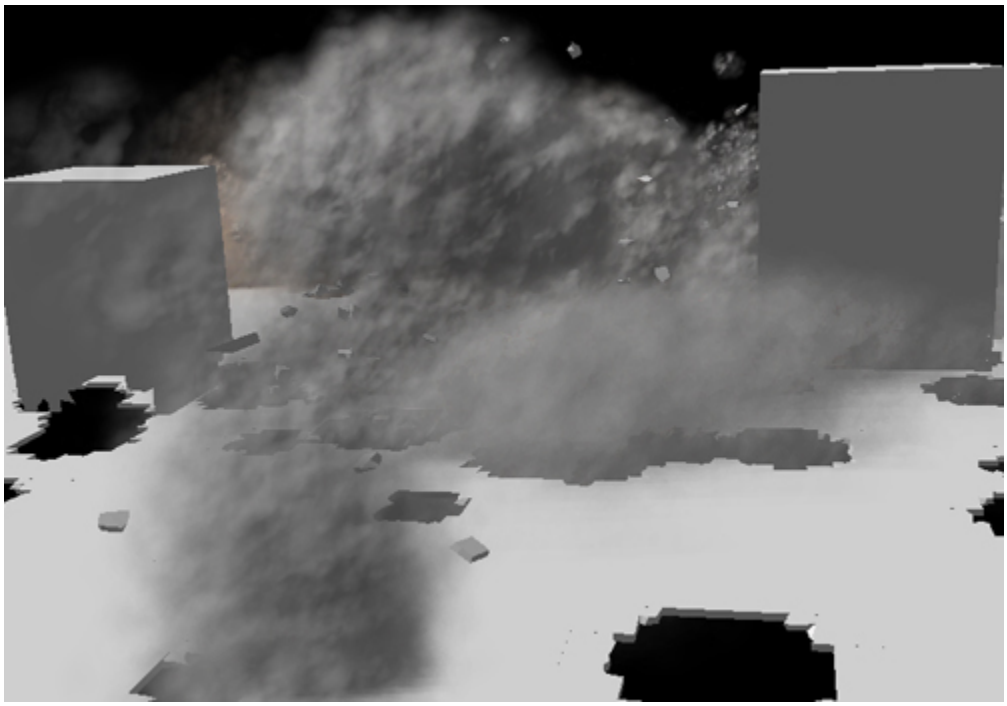
How the Sample Works

The sample works by rendering smoke particles into an offscreen buffer. The normals, opacities, and internal lighting are accumulated, and the final lighting and composite onto the original scene is done in another pass.

Accumulating Normals and Opacity

Particle systems representing smoke are often rendered as a series of screen-aligned sprites. Any lighting information is computed when the particle is rendered. The resulting lit particle is alpha-blended over the top of existing scenery or particles. This results in a flat-looking particle system. Whereas the system is supposed to represent a volume of smoke, the particles are still light as if they are individual entities.

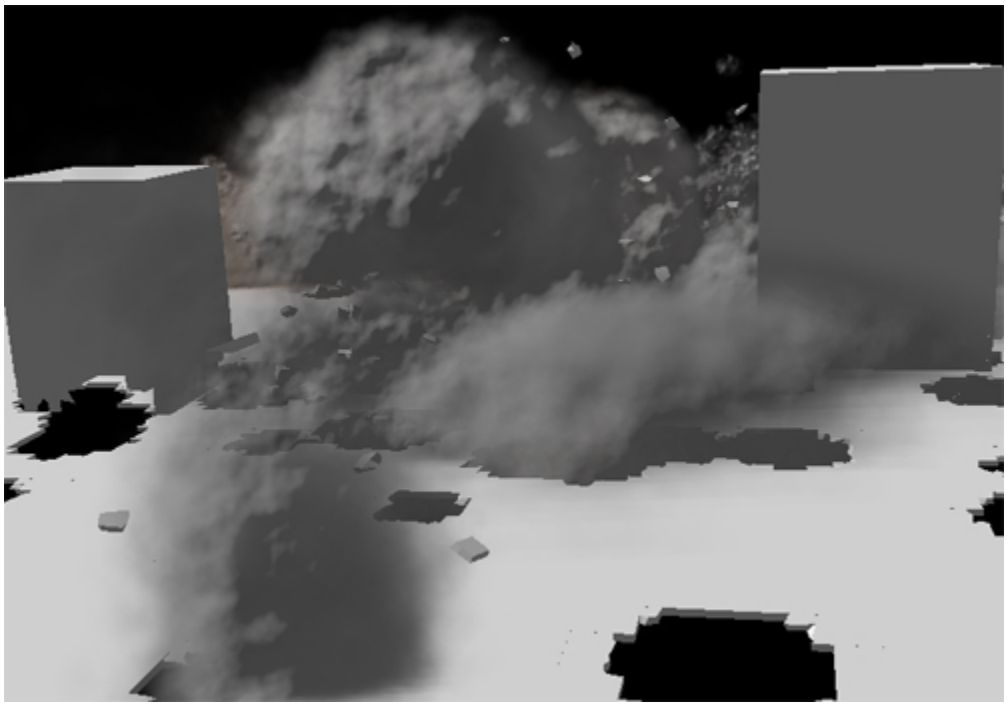
Figure 1. Forward Particles: Flat-looking



Rendering the particles in a deferred manner allows us to accumulate normals and opacity in an offscreen buffer and then light the accumulated data in another pass. This allows us to treat and light the particle

system as one continuous entity rather than individual sprites.

Figure 2. Deferred Particles: Overall shape is more defined.



Lighting the Particles

To add more realism, the particles are lit not only by the scene light (the sun), but also by internal lights generated by the explosions themselves. Each light has a short but intense lite at the center of each explosion. These lights can affect particles in neighboring explosions as well; therefore, there is a limit on the maximum number of lights that can be active at any one time. In the vertex shader, each vertex loops over the active lights in the scene, and calculates the intensity of the light at that vertex using a quadratic falloff. This light intensity is also stored in an offscreen buffer and used in the lighting pass to add extra light to the particles. Since this interior light is not directional, it is simply added to the particles without regard for the particles orientation of the normals.

References

[1] "Deferred Particle Shading." http://unity3d.com/blogs/nf/files/page0_blog_entry73_1.pdf