

Deferred Particles

*This sample is compatible with the Microsoft Game Development Kit (June 2020)*

# Description

This D3D12 sample demonstrates a method for rendering lit smoke particles in either a forward or deferred fashion. When utilizing the deferred path, each particle’s normal, opacity, and color is accumulated into deferred buffers. The result is lit and composited into the original scene in a final pass.



# Building the sample

If using an Xbox One devkit, set the active solution platform to Gaming.Xbox.XboxOne.x64.

If using Project Scarlett, set the active solution platform to Gaming.Xbox.Scarlett.x64.

*For more information, see* Running samples*, in the GDK documentation.*

# Using the sample

This sample uses the following controls.

|  |  |
| --- | --- |
| Action | Gamepad |
| Exit the Sample | Select |
| Move Camera | Left/Right Stick |
| Toggle Forward/Deferred | “A” Button |
| Pause Simulation | “X” Button |

# Implementation notes

The particle simulation itself is performed on the CPU. The resulting particle positions are copied to GPU memory each frame for rendering. The memory is sub-allocated from within a transient D3D12 resource committed to an upload heap, which is CPU-writable, GPU-readable. The GPU virtual address to this memory is then referenced directly as a vertex buffer using a D3D12\_VERTEX\_BUFFER\_VIEW and ID3D12GraphicsCommandList::IASetVertexBuffers(…).

Two rendering modes are available – forward and deferred. In the forward path particles are rendered directly into the scene. When following the deferred path particle data is accumulated into two deferred buffers – a normal map and an opacity/color map. When particle rendering is complete these deferred buffers are used to light the particles and composite them back into the main scene.

The particles are not only lit by the scene light (a single directional light), but also are lit by point lights inside the explosions themselves. These point lights are non-directional, so their lighting contribution to the particles is based on quadratic falloff over distance, rather than orientation.

# Update history

April 2019 – Port to new template from legacy Xbox Sample Framework.

# Privacy Statement

When compiling and running a sample, the file name of the sample executable will be sent to Microsoft to help track sample usage. To opt-out of this data collection, you can remove the block of code in Main.cpp labeled “Sample Usage Telemetry”.

For more information about Microsoft’s privacy policies in general, see the [Microsoft Privacy Statement](https://privacy.microsoft.com/en-us/privacystatement/).