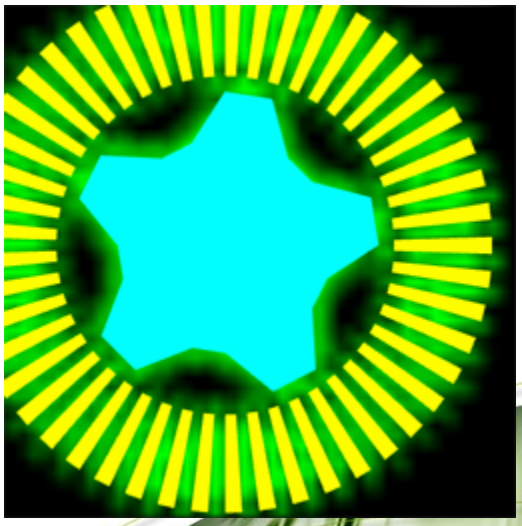
Technical Report

Antialiasing with Post-Processing



DEVELOPMENT

Antialiasing with Post-Processing

Post-processing effects are becoming increasingly popular in 3D games. From the video noise, lens flares, and edge detection in "Wreckless" to the glow in "Tron 2.0," "World of Warcraft," and other titles, post-processing effects offer a wide variety of techniques to improve and distinguish real-time 3D rendering. Since post-processing operates on a 2D image stored in a render-target texture (RTT), it is tempting to generate the render-target texture image by rendering the scene directly into it. Under Direct3D9, it is not possible to use full-scene antialiasing (FSAA) when rendering to a RTT, so such an approach would not offer end users the benefit of hardware-accelerated FSAA. Instead, it is still possible to post-process an antialiased scene by copying the FSAA backbuffer to a render-target texture.

The copy operation is available through the hardware-accelerated IDirect3DDevice9:: StretchRect(..) function. Aside from enabling FSAA, using StretchRect(..) can benefit your application in other ways. Namely, if you are downsampling the backbuffer for post-processing, as is common in creating glow and bloom effects, StretchRect(..) can perform the downsampling and save you the memory cost of a full resolution render target texture to match the flip chain backbuffer. Rendering first to the flip chain also saves you the work of rendering the full resolution render-target texture into the flip chain.

The StretchRect(..) approach is illustrated in the acompanying demo. The demo first renders a rotating teal object and surrounding yellow objects to an antialiased backbuffer. StretchRect() copies the FSAA backbuffer to a render-target texture which is half the resolution of the backbuffer. This RTT is processed to further reduce the resolution. The low resolution copy is then multiplied by a constant green color and added to the backbuffer. It is added at several jittered positions to produce the green halo effect. Note that applying the low resolution copy at several jittered positions is not the best method for creating a good halo or glow effect. A separable convolution is best for that. For reference, see the "GPU Gems" article referenced below.

Bibliography

1. Greg James, John O'Rorke, "Real-Time Glow" in "GPU Gems" ed. Randima Fernando, Addison Wesley : 2004, ISBN 0-321-22832-4, p 343 http://developer.nvidia.com/object/gpu_gems_home.html
2. Greg James, "Operations for Hardware-Accelerated Procedural Texture Animation" in "Game Programming Gems 2" ed. Mark DeLoura, Charles River Media : 2001, ISBN 1-58450-054-9, p 497
3. NVIDIA Developer Web site: 2003. http://developer.nvidia.com

Notice

Trademarks

Copyright

NVIDIA Corporation
2701 San Tomas Expressway
Santa Clara, CA 95050
www.nvidia.com