Virtual Texture Demo

http://www.linedef.com

This is an implementation of Virtual Texturing (also known as MegaTexture in id Tech 5). It is written in C# and uses Direct3D 10 with the help of SlimDX. I'm releasing it under the MIT license. If you find bugs or fix any problems please let me know!

Virtual texturing is analogous to virtual memory. A large texture (potentially larger than can fit in memory) is split into blocks of pixels called pages in a preprocessing step. During runtime the frame is rendered to determine which pages are required. The information is downloaded to the CPU, analyzed, then the required pages are scheduled to load in the background. Loaded pages are uploaded into a texture atlas. Pages are tracked in a quad tree which is used to build a page table. When rendering the scene, the page table is used to map from normal texture coordinates to the page stored in the texture atlas.

Features

- Huge texture sizes (see notes).
- Smaller memory footprint compared to regular texturing.
- Frame feedback based page requests.
- Out of core paging of tiles and seamless streaming.
- Indistinguishable from regular texturing (for trilinear).
- Out of core tile generator.

Notes

This is a simple, non optimized, implementation designed to illustrate the technique. I have support for bilinear and trilinear filtering. Trilinear filtering is calculated by doing two lookups of the virtual texture and lerping between them.

id Software and Sean Barrett talk about compressing the pages on disk as well as with DXT. I simply load the pages uncompressed. As a result the tile caches are huge on disk and loading may be a little slower than if compressed. Defragmenting your hard drive can greatly help with this. As it is now this implementation benefits from disk caching. It wouldn't be hard to add compression, I left it out for simplicity.

I've only tested it with textures up to 64k. I'm assuming it will work up until the point you reach float precision limits, in which case you could probably use doubles, or pack your texture coordinates into multiple values.

This project was built and developed on an x64 machine with 6gb of RAM. All my memory is used during the tile generation phase. There may be address space issues on x86.

Future Work

Ok, so I can load huge textures, how do I create them? With the right tools it shouldn't be a problem. Photoshop, 3dsmax, Mudbox, etc. don't support large textures in a way which is easily workable in a production environment. With an editor that supports virtual texturing you should be able to create the textures without having to worry about memory. I've been working on a texture painting/stamping tool that I plan on extending with virtual texturing. I will post more information on my website soon.

Usage

Download the binaries and one of the data files. Make sure you have downloaded and installed the required libraries (see Requirements). Extract the data into the "Data" folder from the archive. I've included two sets of data just in case you don't want to download the full size one. The data is compressed with 7zip because of its great compression ratios.

When you start the demo a window pops up that will give you various options. You can mouse over the various options to see what they do. The sample data I've included needs to be processed before running. It may take a few minutes.

The standard first person controls are used to navigate (WSAD/Mouse). The mouse wheel controls the camera speed. Mouse1 zooms. Tab toggles application focus. There are a few demo specific controls as follows:

- F1 - Toggle colored mip levels. When you select this option the page loader will fill pages with colors based on the page's mip level instead of texture data from disk.
- F2 - Toggle page borders. This option will add a green border to the edges of a page to help visualize page loading.
- 1/2 - Decreases/Increases the mip bias. The mip bias controls how far away from the camera the pages are requested.
- Space Bar - The space bar toggles updating of the virtual texture.

Requirements

- .NET Framework 3.5
- SlimDX (August 2009)

Links

I've based my implementation on what I have read from various sources. Some of them include:

- Sean Barrett's: Sparse Virtual Texturing
- id Software's: id Tech 5 Challenges
- Crytek's: Advanced Virtual Texture Topics

Credits

- Planet textures are from NASA.
- Puget Sound height map from The University of Washington.
- Some of the textures are from Mayang's Free Texture Library.

Change Log

1.02 (11/29/09):

- Removed dependency on Microsoft's Parallel Extensions for .NET 3.5 June CTP

1.01 (11/11/09):

- Added plane geometry
- Added 8k demo scene preset

1.0 (10/29/09):

- Initial release (1.0)