



Atlas Creation Tool Using the Command-line Tool

The Atlas Creation Tool is a command-line tool that accepts a collection of arbitrary textures and packs them into one or more texture atlases. At the end of packing it generates a human-readable texture-atlas index (**.tai**) file that describes how each original texture maps to an atlas (for example, which atlas file contains the original texture and which sub-rectangle of that atlas it occupies). The **.tai** file thus allows transforming the original texture-coordinates to texture-coordinates that access the identical data from an atlas.

To compare the output of the Atlas Creation Tool—the generated atlases—to the original texture data, you can run the Atlas Comparison Viewer (refer to the *Atlas Comparison Viewer User Guide*.)

Using texture atlases instead of individual textures helps in reducing state changes (**SetTexture ()** in particular), which reduces the number of **Draw ()** calls per frame. Reducing state change and **Draw ()** calls per frame ultimately reduces CPU-utilization caused by driver overhead. The ShaderX3 chapter *Improved Batching Via Texture Atlases* (also available as the NVIDIA white paper *Improved Batching Via Texture Atlases*) discusses the theory, benefits, and limitations of texture atlases in more detail.

Using the Atlas Creation Tool

The Texture Atlas Tool has the following command-line invocation syntax:

```
TextureAtlasTool.exe [-h -help -? -nomipmap -volume
-halftexel -width <w> -height <h> -depth <d> -o <filename>]
<img1> <img2> ...

-h           prints this usage help
-help       prints this usage help
-?          prints this usage help
-nomipmap   only writes out the top-level mipmap
-volume     only valid w/ -nomipmap: make atlases
            volume textures
-halftexel  adds a half-texel offset to the
            generated texture coordinates
-width <w>  limits texture atlases to a maximum width
            of w texels
-height <h> limits texture atlases to a maximum
            height of h texels
-depth <d>  limits texture atlases to a maximum depth
            of d texels
-o <filename> mandatory option that specifies output
            filename
```

What this means is that all parameters are optional except for the output filename and a list of two or more images to pack into an atlas. The tool supports the following file formats: **.bmp**, **.dds**, **.dib**, **.jpg**, **.png**, and **.tga**. Textures can be any size, including non-square.

Typing the following on the command-line:

```
TextureAtlasTool.exe -o grass.png graydirt.png
```

Given two images **grass.png** and **graydirt.png** of the same format generates the output shown in Figure 1. It also generates two new files: **TexAtlas0.dds** and **TexAtlas.tai**.

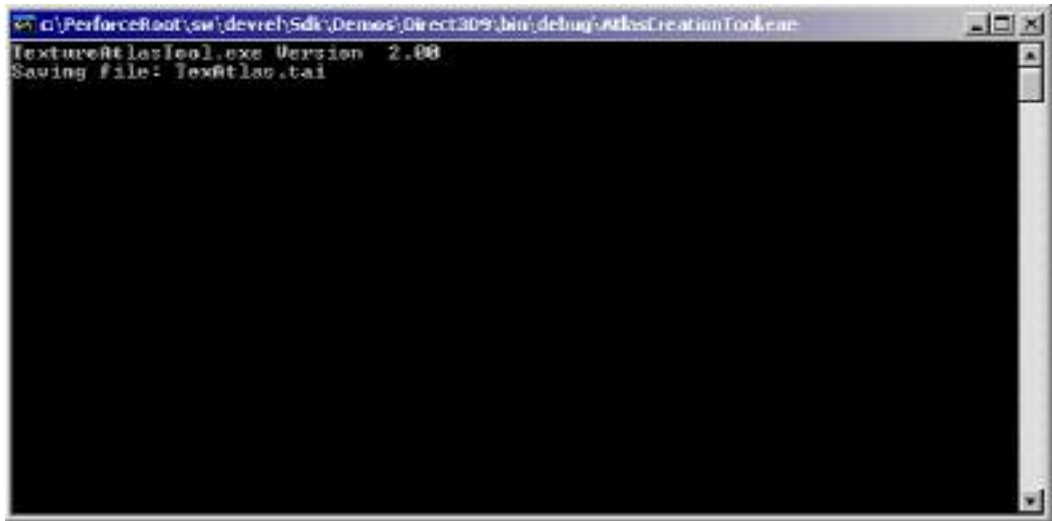


Figure 1. Command-line Output for Invoking Texture Atlas Tool with Minimal Parameters

TexAtlas0.dds is an image file containing **grass.png** and **graydirt.png** in the Direct3D format X8R8G8B8. The **TexAtlas.tai** file is human-readable (see Figure 2) and describes how the texture coordinates of models using these textures need to be modified to access the same data out of the generated atlas: For example, all models using the **grass.png** texture before require that their **u**-coordinate be divided by **2** and add **0.5**; to access the entire **grass.png** texture out of the **TexAtlas0.dds** the atlas texture-coordinates need to range from **(0.5, 0)** to **(1, 1)**. Similarly, all models using the **graydirt.png** texture require that their **u**-coordinate be divided by **2**; to access the entire **graydirt.png** texture out of the **TexAtlas0.dds** the atlas texture-coordinates need to range from **(0, 0)** to **(0.5, 1)**.

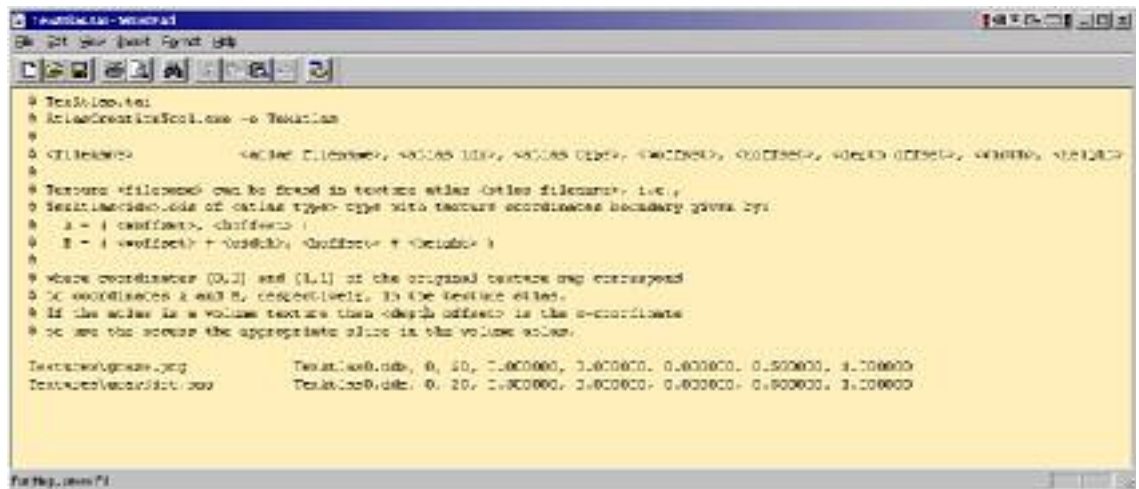
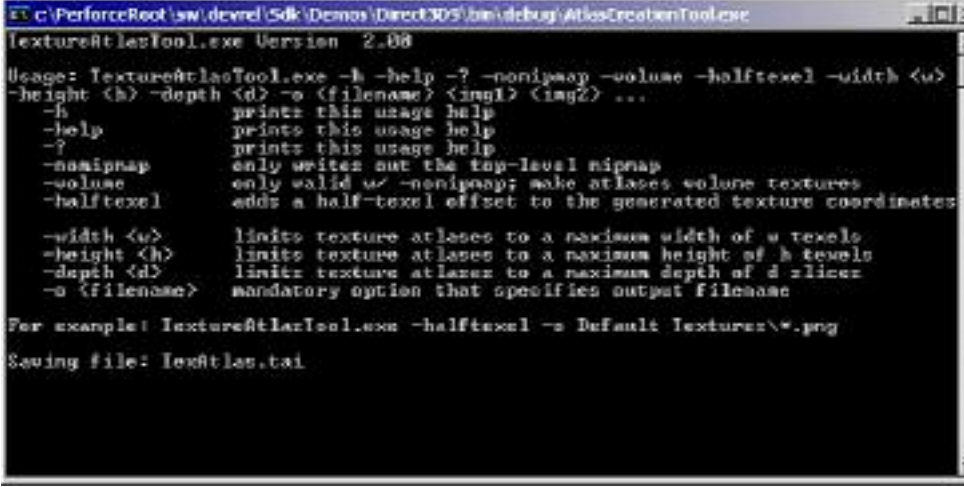


Figure 2. Generated TexAtlas.tai File

Command Line Options

Table 1 lists the commands used from the Atlas Creation tool.

Table 1. Command Line Options

Command	Description
-h -help -?	<p>The <code>-h</code> command-line option prints the current usage pattern as shown:</p>  <pre> Usage: TextureAtlasTool.exe -h -help -? -nomipmap -volume -halftexel -width <w> -height <h> -depth <d> -o <filename> <img1> <img2> ... -h prints this usage help -help prints this usage help -? prints this usage help -nomipmap only writes out the top-level mipmap -volume only valid w/ -nomipmap; make atlases volume textures -halftexel adds a half-texel offset to the generated texture coordinates -width <w> limits texture atlases to a maximum width of w texels -height <h> limits texture atlases to a maximum height of h texels -depth <d> limits texture atlases to a maximum depth of d slices -o <filename> mandatory option that specifies output filename For example: TextureAtlasTool.exe -halftexel -o Default\textures*.png Saving file: TexAtlas.tai </pre>
-nomipmap	<p>The <code>-nomipmap</code> option instructs the application to ignore all mip-maps for all textures: as a result, all generated texture-atlases contain only the top-level texture, i.e., no mip-maps. This behavior is useful for user-interface or similar textures where it is guaranteed that none of the mip-map levels are ever accessed. In general, disabling mip-mapping is undesirable, as it causes prohibitive performance reductions.</p> <p>In contrast, not specifying <code>-nomipmap</code> loads and copies all of a texture's mip-maps into the respective atlas; if a texture does not contain any mip-maps or only a partial mip-map chain, then the Texture Atlas Tool relies on <code>D3DXCreateTextureFromFileEx()</code> with <code>D3DX_DEFAULT</code> parameters to generate the missing mip-maps. The resulting quality is thus questionable: using NVIDIA's texture tools (http://developer.nvidia.com/object/nv_texture_tools.html) to generate all mip-maps for all textures before passing them to the Texture Tool Atlas is preferred.</p>
-volume	<p>The <code>-volume</code> option specifies that all generated atlases are to be volume textures. This option requires that <code>-nomipmap</code> is also specified, since volume atlases cannot store the mipmaps of textures.</p>
-halftexel	<p>The <code>-halftexel</code> option offsets all generated texture coordinates by half a texel so as to ensure that only the texels of the original texture are accessed. Refer to the ShaderX³ chapter for details.</p>
-width <w>	<p>The <code>-width</code> option limits the maximum width of all generated atlases to <code>w</code>.</p>
-height <h>	<p>The <code>-height</code> option limits the maximum height of all generated atlases to <code>h</code>.</p>
-depth <d>	<p>The <code>-depth</code> option limits the maximum depth of all generated volume atlases to <code>d</code>.</p>
-o <filename>	<p>The <code>-o</code> option specifies the name of the generated atlas and <code>.tai</code> files. For example, specifying <code>-o Test</code> will generate atlas files of the form <code>Test0.dds</code>, <code>Test1.dds</code>, etc, and a <code>Test.tai</code> file.</p>

Typical Use Case

A graphics-engine programmer identifies a collection of textures that are responsible for a large number of `DrawPrimitive()` calls due to repeated calls to `SetTexture()`. These textures ideally are of or are converted to the same format to maximize the efficiency of packing them into atlases. If some of these textures do not contain complete mip-map chains, integrate NVIDIA's texture tools into the tool chain to generate and store these mip-maps together with the original textures. (Various file-formats, e.g., `.dds`, allow storing mip-map levels together with the original image.) Similarly, integrate the Atlas Creation Tool into the tool chain to convert these texture collections into collections of atlases. Create and integrate a separate tool that parses the generated `.tai` file to remap texture-coordinates of models to enable texturing out of the generated texture atlases.

Optionally, use the Atlas Comparison Viewer or a similar home-made tool able to display converted models to inspect the resulting image quality of texturing out of an atlas versus texturing from the original texture.

Known Bugs

Running with the DirectX 9 Summer 2003 DEBUG runtime and packing DXTn-format textures generates a Direct3D error and thus forces the application to exit. We are currently investigating the root cause for this error. Switching to the retail run-time works around that error.

Next Version Features

Following is a list of currently unimplemented features under consideration for the next version of this application.

- ❑ Add `-cubemap` option: only valid if at most six textures of the same format/dimension are supplied.
- ❑ Use the reference rasterizer as the hardware device: that way we get maximum capabilities, texture formats, and texture resolutions.
- ❑ Add a command-line option to force all generated atlases to a particular format.



Notice

ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE.

Information furnished is believed to be accurate and reliable. However, NVIDIA Corporation assumes no responsibility for the consequences of use of such information or for any infringement of patents or other rights of third parties that may result from its use. No license is granted by implication or otherwise under any patent or patent rights of NVIDIA Corporation. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. NVIDIA Corporation products are not authorized for use as critical components in life support devices or systems without express written approval of NVIDIA Corporation.

Trademarks

NVIDIA and the NVIDIA logo are trademarks or registered trademarks of NVIDIA Corporation. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright

© 2004 NVIDIA Corporation. All rights reserved



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