RenderMonkey™: An Effective Environment for Shader Prototyping and Development

Natalya Tatarchuk
ATI Research, Inc.
Overview

- What is RenderMonkey?
- Shader development workflow
- Integration into existing pipelines
- Conclusion
It’s Not A Monkey, It’s Not a Man: It’s RenderMonkey

• Shader development tool
  – Abstracts GPU programming
• Manages the shaders and all associated visual resources in a single environment
• Rapid prototyping and debugging of new graphics algorithms
• Artist-friendly environment
  – Fosters collaboration between effect programmers and artists
Uses Standard Graphics APIs

Shader production on any hardware with shader capabilities for standard graphics APIs

- OpenGL 1.5 with GLSL support
- DirectX 9.0c with HLSL and assembly support
  - Includes support for all shader models up to SM 3.0
Abstracting GPU Programming

Why hide the truth?
- Shaders are more than just stand-alone programs
- Setting all related state can be complex

RenderMonkey simplifies that task
- Abstracts GPU programming setup via user interface and database manipulation
- Now you can focus on shader and algorithm development

Siggraph 2004
9 Months Since Version 1.0

More than 100,000 downloads

- Adopted by developer community as a powerful tool for shader prototyping and development
  - Prototyping new algorithms
  - Developing production shaders
  - Used Beta SDK for custom exporter / loader plug-ins

Version 1.5 released at Siggraph 2004

- Many new features: making shader production more streamlined

Siggraph 2004

• Setting up draw calls, texture surfaces, primitives, multipass rendering, offscreen buffers – as easy as a few clicks
  – You don’t even need to be a programmer to do it
• Full control over the rendering setup
  – A general shader development environment
  – Not just one preset way to create your effects
• All effect’s data is stored in an extensible XML file
  – Includes a published DTD
Managing Effect Setup

• Manipulate effect database via workspace tree view

• Specify texture and model repositories
  – Allows stream-lined creation of texture and geometry objects

• No hunting for data details: informative tooltips tell it all

• A monkey with a long memory
  – Application remembers user preferences for editors and window positioning
  – Customizable application settings
Share Your Knowledge

Effect package import / export functionality

- Mechanism for automatic packaging of effects with geometry models and textures in a single zip file
- It can then be directly imported back into the application
- Zip, ship and view!

Facilitates sharing of algorithms in the graphics community
Focus on Shader Development

Custom shader editor for each shading language

- User-customizable syntax highlighting
- Customizable text editing settings
- Integrated compile error reporting
State-of-the-art DirectX Rendering

- Support for multiple render target output
- Multipass rendering
- Customizable buffer formats, multisampling type and variety of other settings
- HLSL include file support
  - Ships with a set of HLSL math and rendering functions
- Export to Microsoft .FX
New: OpenGL with GLSL Support

RenderMonkey is the first shader development IDE supporting GLSL
  - Multipass rendering
  - Render to texture passes
  - Can use the same resources as DirectX (textures, models, etc.)

Cross-vendor implementation uses efficient OpenGL techniques
  - Vertex buffer objects when available, vertex arrays otherwise
  - Automatic mip map generation when available
Control Simulation with Engine-Driven Parameters

Can use engine parameters to access more state within shaders
- Transformation matrices
- Time parameterization, random values
- And more

Customizable parameter names linked to specific semantics
- Semantics match Microsoft .FX annotations

~ 80 total
Access User Input from Within Shaders

User input via mouse is directly accessible from shaders

- Use engine mouse parameters: mouse buttons state and coordinates

Examples:

- Fluid flow injection
- NPR painting program
- Object selection
Debugging Shaders and Performance Tuning

- Interactive preview with full rendering resource error reporting
  - Minimize user error in setup
  - Quick iteration on effect’s algorithm, look and feel
- Frame statistics provide performance hints
- DirectX optimizations
  - Built-in HLSL disassembler
  - Integrated ALU and texture op counter
Know What the Algorithm is Actually Doing

Output a shader quantity directly to the preview
- Requires understanding of how quantities map to visual results

View pass build-up and pass interaction for multipass effects
- Breaks down effect’s creation into distinct steps
- Allows easy comprehension of the algorithm
- Aids in quick debugging when things go awry
Generate Look-Up Tables and Geometric Objects

Support for procedural texture and geometry object creation

- Via existing generator plug-ins
  - Procedural texture via HLSL functions (2D, 3D & cube maps)
  - Basic geometry shapes and auto edge quad generation
- Create your own data of any type

Preserve your data: anything generated can be saved to file by RenderMonkey
Give Power to the Artists!

Artist’s involvement is crucial for shader effects

- Expose the power of programmable shaders to the artists without slowing them down

Let shader developers collaborate with artists

- Algorithm developer can select which parameters are artist-editable
- Artists can view only relevant shader parameters via Art tab or in a single Artist Interface
Artist Interface

- Provides familiar look and feel of DCC widgets to artists working on shader effect creation
- Tweak parameters and *instantly* see changes
  - Quicker iteration to achieve the best visual result
  - Allows exploration of parameters influence on the effect
What’s in My Texture?

- All texture surfaces can be previewed
  - Preview thumbnails and preview full surfaces including the contents of the alpha channel
  - Includes all renderable offscreen buffers
What’s in My Texture?

• All texture surfaces can be previewed
• Dynamic preview of offscreen buffer contents
  – Separate preview of all renderable surfaces
  – Dynamically updated as effect is being rendered
  – View contents of a shadow map or pressure buffer
Texture and Model Resource Management is Simple

- Specify user resource repositories for textures and models
Texture and Model Resource Management is Simple

- Specify user resource repositories for textures and models
- Connect rendering resources with external editors
  - Specify desired program for texture and model editing
  - Clicking on a texture thumbnail or geometry object launches an external editor
Texture and Model Resource Management is Simple

• Specify user resource repositories for textures and models
• Connect rendering resources with external editors
• Automatically update rendering resources as they are modified outside of RenderMonkey
  – The application auto-rebuilds geometry objects and texture surfaces whenever files are updated on disk
Integrating into Existing Pipelines

- Create custom components using our SDK (ships with version 1.5)
- Our entire application was developed using this plug-in architecture
  - Plug-in architecture allows solving problems not originally anticipated
  - Provides a convenient mechanism for us to create and distribute new tools using this framework
  - Truly extensible architecture
Variety of Possible Custom Components

- Exporters / Importers
- Data generators
  - Ex: Procedural texture / geometry generation
- Texture / Geometry object loaders
- Data editor plug-ins: create your own custom widgets
- RenderMonkey ships with a set of convenient libraries
  - Effect database management
  - Efficient STL-like library
  - Math, graphics and UI widgets libraries
SDK APIs and Compatibility

- Written in pure C++ on Windows platform
- Supports development on Visual Studio 6.0 and Visual Studio 7.1 .NET 2003
- Create plug-ins using either Win32 or MFC
- Supports both DirectX and OpenGL
Conclusion

RenderMonkey is a powerful development environment for shader-based effects

- Designed for rapid effect prototyping
- Minimal state setup overhead
- Artist-friendly
- Extensible framework allows custom component creation
- Adopted by the industry
Acknowledgements

• To the entire RenderMonkey team for their hard work and dedication
  – Ben Mistal
  – Toshiaki Tsuji
  – Bill Licea-Kane

• To 3D Application Research Group for inspiration in all that is graphics

• To 3Dlabs, Inc. for contributions to OpenGL rendering pipeline
Questions?