



**NVIDIA**®

# Best Practices for Multi-threading

Eric Young  
Developer Technology

# Overview



- **Benefits of Dual-Core**
- **NVIDIA Multi-threaded Driver**
  - **OpenGL**
  - **Direct3D**
- **Dual-Core Performance Gains**
- **Summary**

# Benefits of Dual-Core



- **NVIDIA multi-threaded driver**
  - **Producer & Consumer threads dispatch commands to GPU**
  - **Improves parallelism between GPU/CPU**
  - **Performance gains from 10 to 40%**
- **Application has more computation power available!**
  - **Handles even more Complex Math, Physics, and AI**
  - **Add Dual-Core optimizations to your application, and you can achieve up to 2X the performance**
  - **Best combination:**
    - **Application has Dual-Core optimizations**
    - **Running with the multi-threaded driver!**

# OpenGL on Dual-Core



- Both OpenGL runtime and driver can be completely offloaded
- Application gains 20 to 40% performance
- Driver can dynamically switch between multi-thread and single-thread mode

# OpenGL Performance Tips



- **Immediate mode for small amounts of vertex data**  
`glVertex3f(x, y, z)`
- **VBO(s) & Display Lists for large amounts of Vertex Data**
- **Avoid passing Vertex Pointers**
  - Driver will need to copy extra data
  - Use Buffer Objects instead for better performance
- **When using VBOs, use element array buffers**  
`glBindBufferARB(GL_ELEMENT_ARRAY_BUFFER_ARB, index_buffer);`

# OpenGL Performance Tips (contd.)



- **Avoid glGetXXX calls or Driver Queries**
  - CPU Thread synchronization needed
  - Release builds should not have many `glGetError()` calls
- **Do not lock application threads to single CPU**
  - OS can do a better job scheduling CPU resources
- **Avoid using Vertex Arrays without Buffer Objects**
  - Requires a data copy from pointer for every call
  - Driver will have to wait until it is finished using data

# Direct3D on Dual-Core



- **Also uses Producer & Consumer threads**
- **Only driver can be offloaded onto the second CPU**  
**Note: Direct3D runtime runs in the application thread**
- **10 to 30% performance gains with a multi-threaded NVIDIA driver**

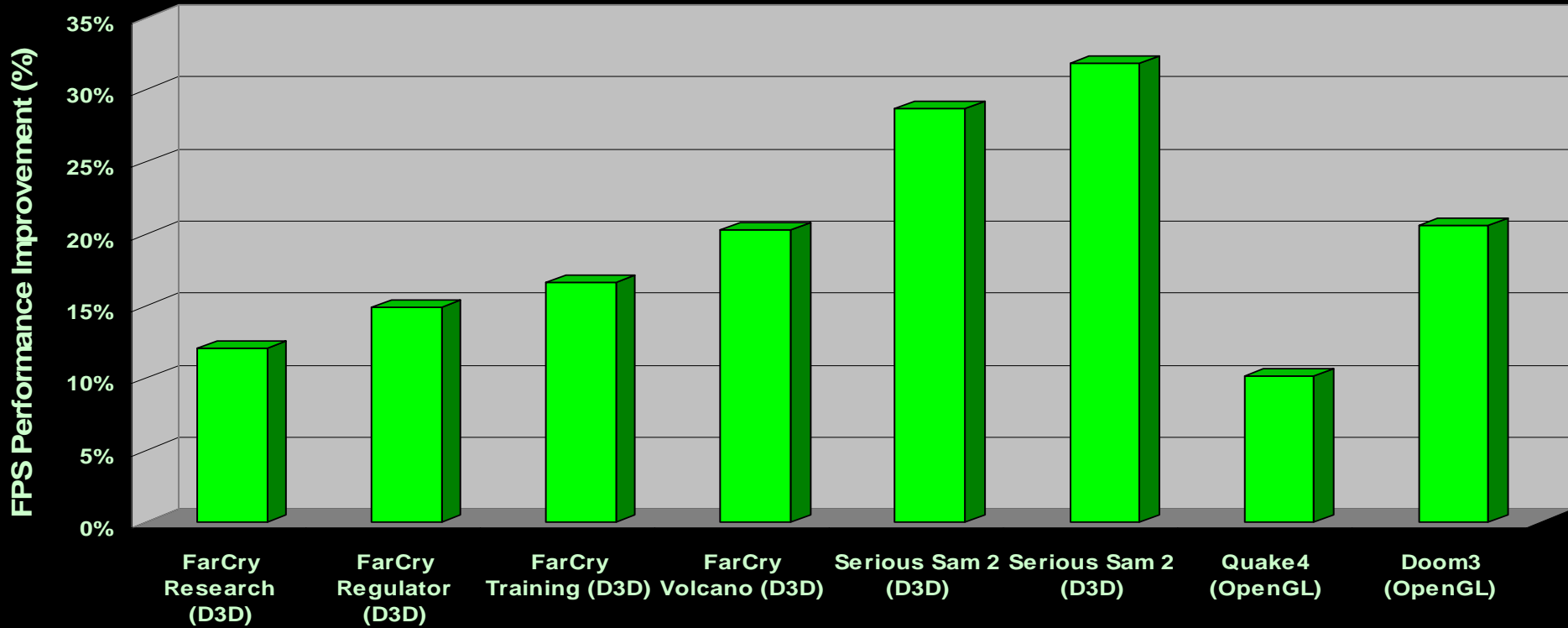
# Direct3D Performance Tips



- **Pack your DrawPrimitive2 calls together**
- **Frequently creating & destroying shaders, VB, IB, and surfaces will impact performance**
- **Avoid allocating too many system memory resources**
- **Minimize use of any locks/unlocks**
- **Avoid any calls that return GPU state information**
  - **Requires a CPU thread synchronization**
  - **Driver Queries are OK (calls are asynchronous)**
- **Do not lock threads to a specific CPU!**



# Total Gains with Dual-Core



# Summary



- **NVIDIA multi-threaded drivers available now**  
**Download drivers from <http://www.nvidia.com>**
- **On a dual-core system 10 to 40% improvement**
- **Avoid any API calls that return driver state**
- **Do not lock application threads to a specific CPU**
- **Applications optimized for dual-core will get even better performance**
- **We will continue to improve dual-core driver performance**

# The Source for GPU Programming

[developer.nvidia.com](http://developer.nvidia.com)

- Latest News
- Developer Events Calendar
- Technical Documentation
- Conference Presentations
- GPU Programming Guide
- Powerful Tools, SDKs and more ...



Join our FREE registered developer program for early access to NVIDIA drivers, cutting edge tools, online support forums, and more.

**nVIDIA**

[developer.nvidia.com](http://developer.nvidia.com)

©2004 NVIDIA Corporation. NVIDIA, and the NVIDIA logo are trademarks and/or registered trademarks of NVIDIA Corporation. Nalu is ©2004 NVIDIA Corporation. All rights reserved.