CUDA and the future of dynamic lighting

Graham Hazel and Chris Doran
GDC 2012
• Introduction to Enlighten
• Enlighten CUDA runtime
• Real-time area lights
• Dynamic object radiosity
Enlighten

- Enlighten is practical real-time global illumination
- Use at run-time to dynamically light your game
- Fast iteration of lighting design offline
Key titles

- Battlefield 3
- Need for Speed: The Run
- EVE Online
- Quantum Conundrum
Radiosity

- Without radiosity
  - Direct illumination plus a global ambient term
  - Image is very flat

- With radiosity
  - Real depth to image
  - Shadows filled in and coloured
  - Subtle gradients of shadows
  - Scene lit with only two lights (sun and skybox)
Enlighten goals

• Pre-computed quality with the flexibility of dynamic lighting
• Creative freedom for artists and designers
• Visual and atmospheric quality bar raised
Enlighten philosophy

• Target the current generation of hardware
  – Use lightmap techniques
  – All the complex calculations should take place in a single pre-processing step

• Keep the technology modular
  – Separate the direct and indirect lighting
  – Separate static and dynamic objects

• Keep the runtime as lightweight and simple as possible
  – See this with CUDA implementation
Runtime Philosophy

• Memory and compute power are contested commodities on PS3 and Xbox360

• Look at a range of optimisation strategies
  – Light a simplified mesh and project
  – Use temporal coherence
  – Stream the calculations

• Originally thought Enlighten would run on GPU, but changed direction for consoles
Design Philosophy

• Enlighten had to scale to all possible scenarios:
  – Interiors
  – Enclosed areas
  – Entire cities
Design Philosophy

• Put the artist in control
  – Instant feedback
  – As many knobs & dials as possible

• You can break the laws of physics
  – Why stick to realistic lighting?
The hardware zoo
Scales of dynamism

- Fully dynamic lighting
- Dynamic direct lighting with static Enlighten data
- Dynamic time of day
- Mixture - static lightmaps with dynamic lighting overlaid
- On-level load lighting
- Static lighting plus small number of direct lights
- Totally static lighting
Reaching all games

• Current consoles fully capable of real-time dynamic lighting
  – This is our base point
• Next-gen preparation on high end PC
  – DX11 / CUDA Features
• Scale back to mobile with hybrid static / dynamic solutions
  – Enlighten runtime ported to ARM architecture
  – Augment with ability to bake
• Introduction to Enlighten
• Enlighten CUDA runtime
• Real-time area lights
• Dynamic object radiosity
What is CUDA?

- Most advanced platform/language for GPGPU
- Widely used in scientific applications
- C-like language for massively parallel execution
- Units of execution (functions) are called "kernels"
- A kernel is launched on a grid of blocks of threads
- Need 10,000s of threads to fill up a graphics card
CUDA and graphics

• CUDA can inter-operate with D3D and OpenGL
• Read from or write to graphics resources directly (textures, vertex buffers...)
• Removes need for CPU<->GPU transfer
• GPU can write to "pinned" CPU memory
• It's fast!
• Further efficiencies from inter-operability
  – Read from shadow maps directly
Enlighten run-time

• Input Lighting
  – Light a large number of point samples
  – Naturally very GPU-friendly!
  – Able to use all D3D resources as inputs
  – We will look into just doing this phase on GPU

• Output computation
  – Optimised for memory usage/throughput
  – Nearby pixels share data
  – This allows us to leverage features of CUDA
Blocks

• Kernels launched on grid of blocks
• Blocks typically hundreds of threads
• Important because:
  – Blocks execute within one streaming processor
  – Data can be synchronised and shared within a block
• Using shared memory is critical for performance
Demo

- Dockyards
- 60+ dynamic lights
- Projected textures
• Introduction to Enlighten
• Enlighten CUDA runtime
• Real-time area lights
• Dynamic object radiosity
Soft shadows
Techniques

• Fixed-size approaches
  – PCF, VSM, CSM, ESM (and others)

• Depth-varying approaches
  – PCSS and variants of fixed-sized versions
  – Largely based on average blocker depth

• Back projection approaches
  – Basis for this technique

• Ray tracing
  – One day…
Techniques

- **Fixed-size approaches**
  - PCF, VSM, CSM, ESM (and others)

- **Depth-varying approaches**
  - PCSS and variants of fixed-sized versions
  - Largely based on average blocker depth

- **Back projection approaches**
  - Basis for this technique

- **Ray tracing**
  - One day...
PCSS limitations

- Large penumbra are expensive
- Noisy if you under sample
- Correct only when average depth exists
- Large area lights push limits
- Artefacts when assumptions fail
- Shadows have a “geometric” look
Backprojection

- Use rectangular light source
- Consider texels in shadow map as occluders
- Find occluders between receiver and light source
- “Back project” to light source to compute occlusion
- Remaining area gives visibility factor
Bitmask Back Projection

• Bitmap Soft Shadows
  – Main variant of Guennebaud 2006.

• Occluders as continuous mesh of quads
• No gaps between occluders (+ and -)
• Fixes under occlusion, but projection more complex
• Use bit mask to represent visible area
• Slower, but fixes over occlusion
This technique

• Basis:
  – Occluders as continuous surface (a la Schwartz)
  – Support both bitmask and area-based variants
  – Adaptive subdivision (a la Dmitriev)

• Plus:
  – 5x5 tap screen-space bilateral reconstruction
  – Downsamplingd 3x3 symmetric SM hierarchy
  – Various bells and whistles
Implementation

• CUDA
  – Hierarchy generation
  – Generate visibility in screen space
  – Upsample and shade

• Non-trivial
  – Too much code to talk through in detail
  – Dump data to memory was main debugging aid
Results
Demo

- Shipyard
- Dynamic area lights
• Introduction to Enlighten
• Enlighten CUDA runtime
• Real-time area lights
• Dynamic object radiosity
Participating dynamic objects

- Significant new feature for Enlighten
- Dynamic objects contributing to radiosity
- Suitable for large dynamic occluders in a scene
- Not suitable or necessary for small dynamic objects
- Runs extremely well on CUDA!
Participating dynamic objects

Standard Enlighten with blocker as a dynamic object

Enlighten with blocker as a participating dynamic object
Demo

- Shipyard
- Participating dynamic objects
Future concepts

• Beyond lightmaps
  – Greater use of probes

• Iterative refinement
  – Local updates for greater dynamism

• Volumetric effects
  – Smoke, fog, particles interacting with lighting
  – Film style compositing techniques
  – A next gen differentiator
Thank you

• Email
  – chris.doran@geomerics.com
  – graham.hazel@geomerics.com