Film Rendering on Game Hardware

Eric Enderton

Eurographics 2006

Outline

- What is film rendering?
- High-quality hidden surface removal on GPU
  - Modified REYES algorithm
  - Parameter-space shading
- Gelato demo
- Remarks on GPU programming
- Since game rendering is so fast, why is film rendering so slow?
  - The “first frame” problem
- Gelato relighting demo

GPU-Accelerated High-Quality Hidden Surface Removal

Daniel Wexler, Larry Gritz,

Eric Enderton, Jonathan Rice

Game Render vs Film Render

(missing image) (not Gelato)

enterthematrixgame.com/html/screenshots11.html

(missing image) (not Gelato)

from Matrix Reloaded movie

www.hollywoodjesus.com/movie/matrix_reloaded/reloaded5.jpg

Game Render vs Film Render

(missing image) (not Gelato)

Narnia game screenshot of girl in snowy woods

(missing image) (not Gelato)

Davy Jones by ILM, from Pirates of the Caribbean: Dead Man's Chest (missing image) (not Gelato)

Frame of CG airplane from “Returner” (Japan, 2002)

High-end Rendering = ?

- No distracting artifacts
- Richness
- High level description

High-end Rendering = ?

- No distracting artifacts
  - Space
    - Good filter (2 pixels wide, smooth)
    - Accurate silhouette → Adaptive tessellation
  - Time
    - Motion blur
    - No pops, no chattering
  - Shading
    - AA texture lookup
    - good derivatives
  - Order-Independent Transparency

High-end Rendering = ?

- Richness
  - GB's of geometry

Geometry : Hair

(~470,000 hairs)

Geometry : Displacement Mapping

model courtesy of Todd Durant

Geometry : Just complex

Ethan Summers & Shiew Yeu Loh

High-end Rendering = ?

- Richness
  - GB's of geometry
    - plus displacement
  - TB's of texture
    - disk, network
  - 10's - 100's of lights
  - 10K line shaders
  - Non-local effects
    - Ray tracing, global illumination, ambient occlusion, caustics, subsurface scattering

(missing image) (not Gelato)

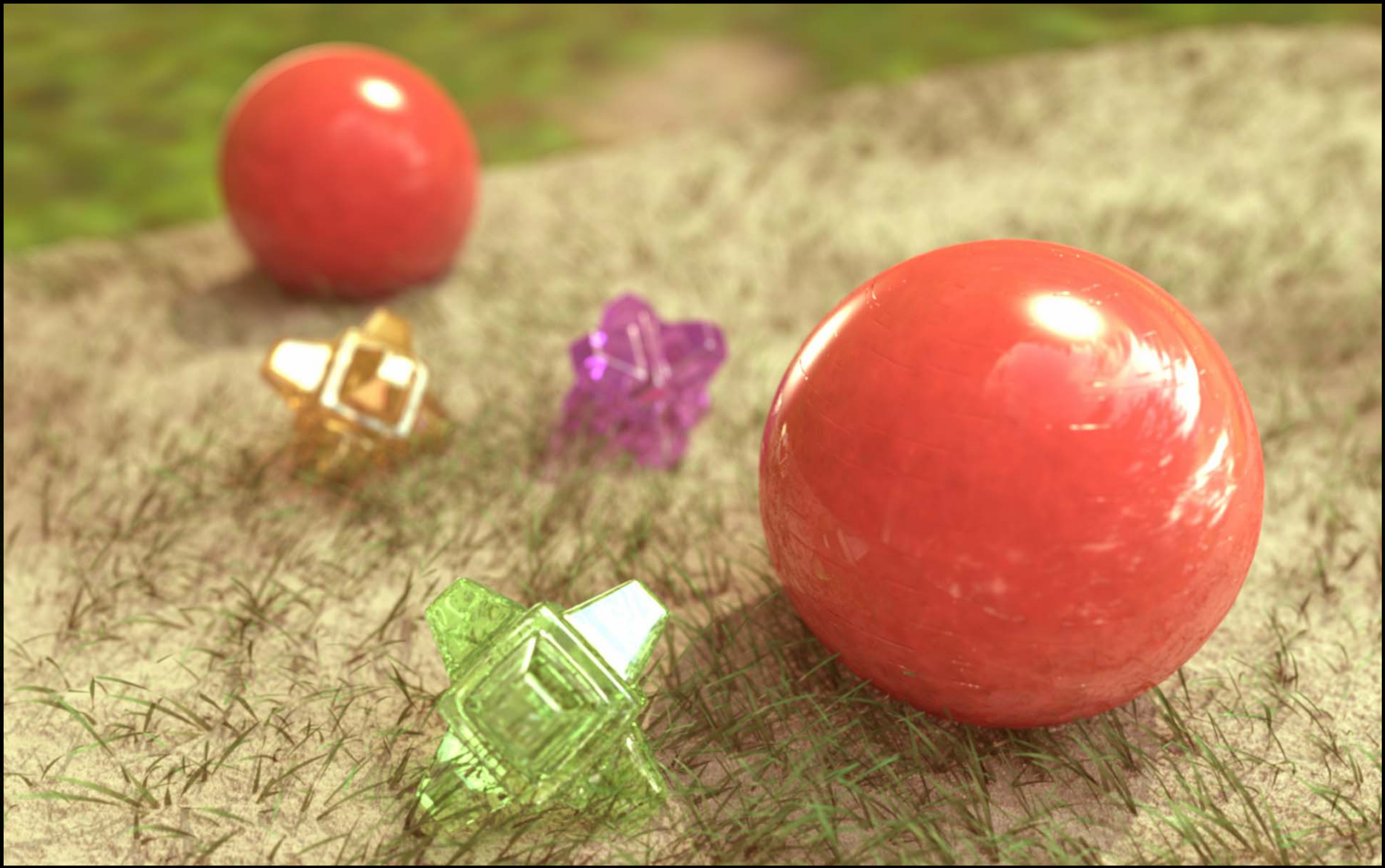
Close-up of diamond ring, from “Stuart Little 2”

Stuart Little 2 © Columbia Pictures

High-end Rendering = ?

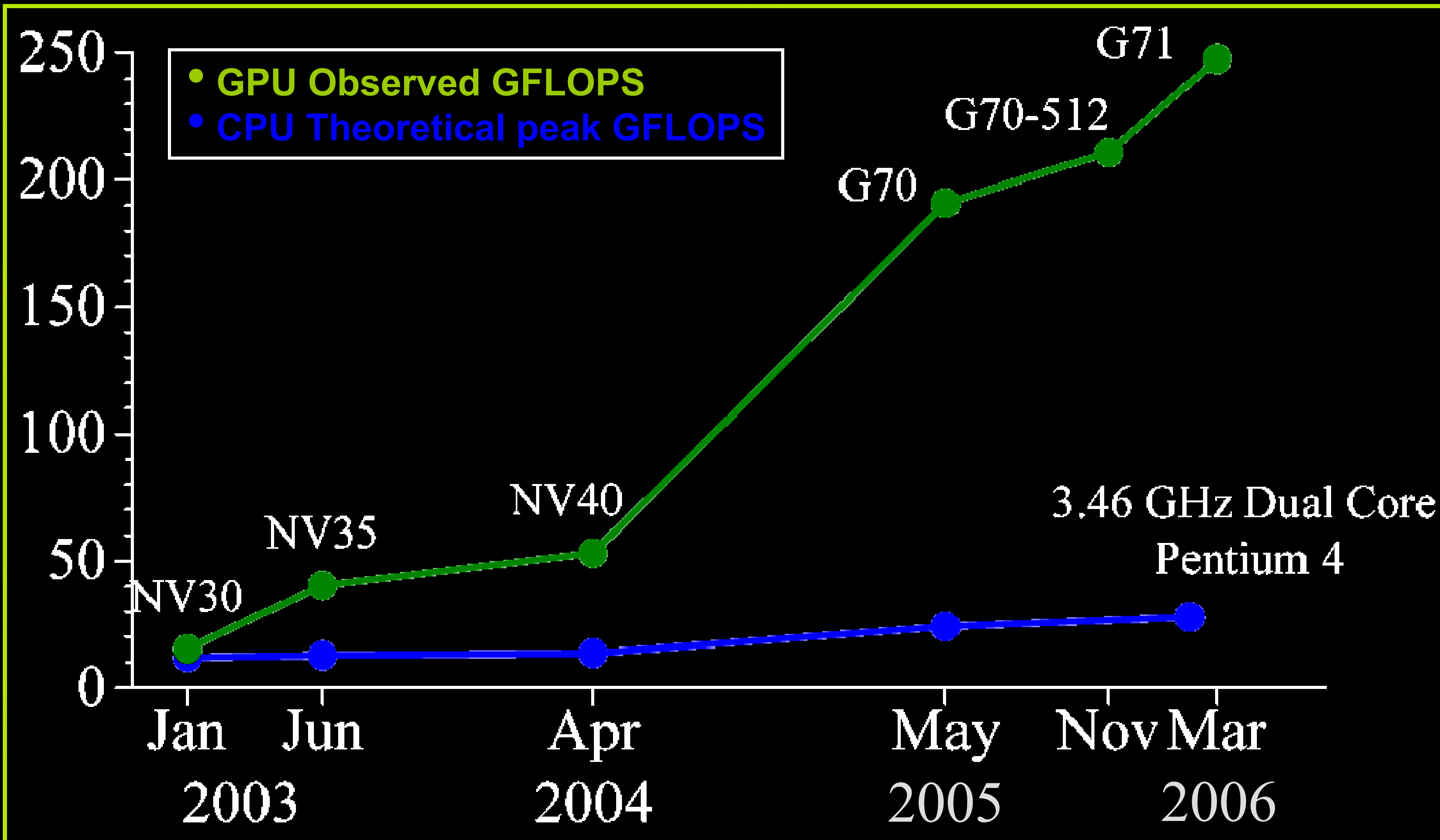
- High level description
  - NURBS, subd's
  - Gelato Shading Language
    - separate lights
    - texture by file name (constructed?)
    - ray queries
  - Delayed geometry (expanded at render time)

High level description



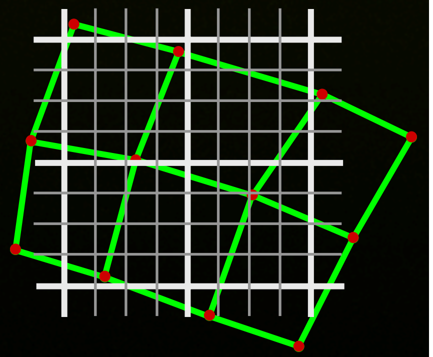
Jared Martin 2006

NVIDIA GPU Pixel Shader GFLOPS

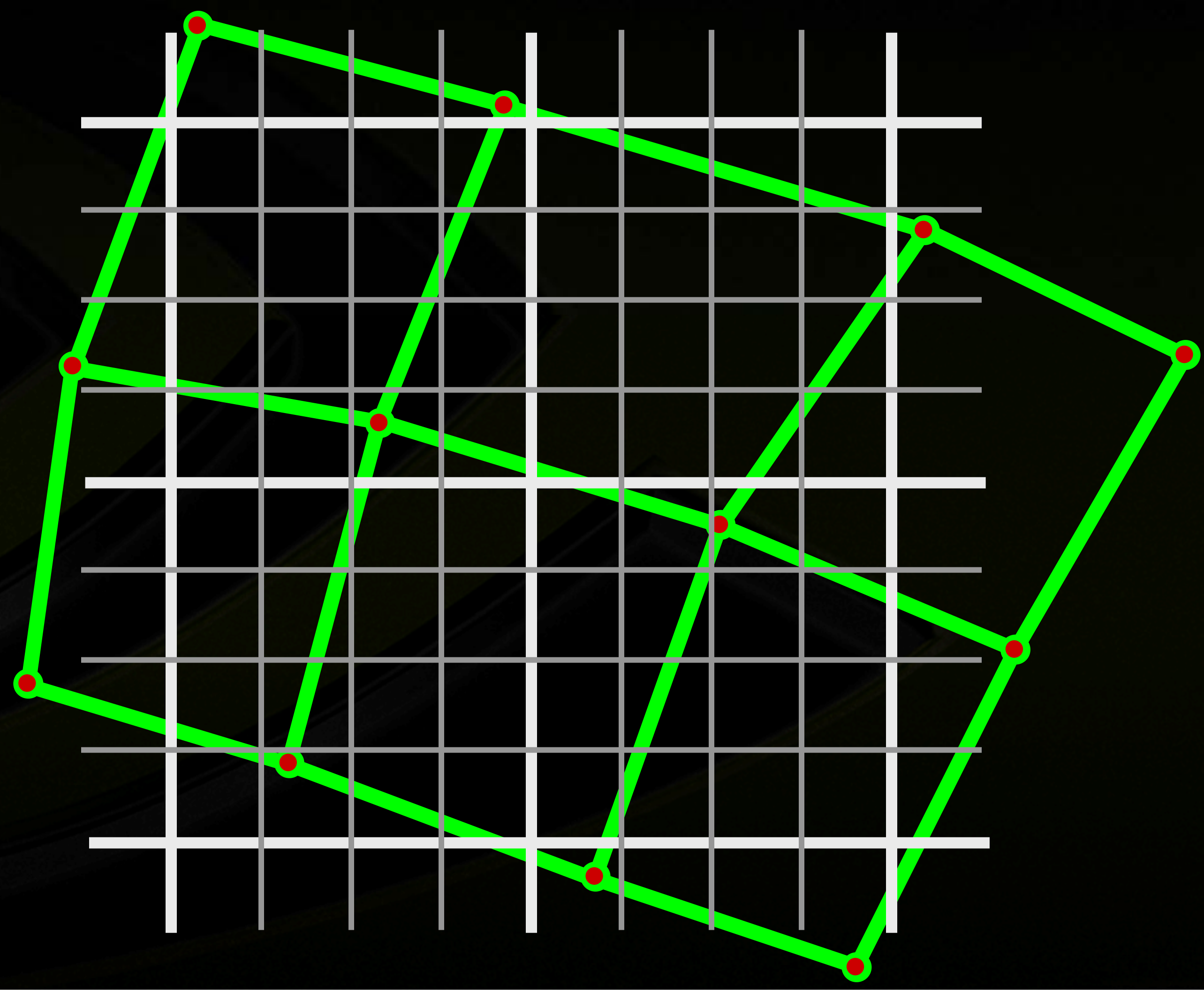


Hider Overview

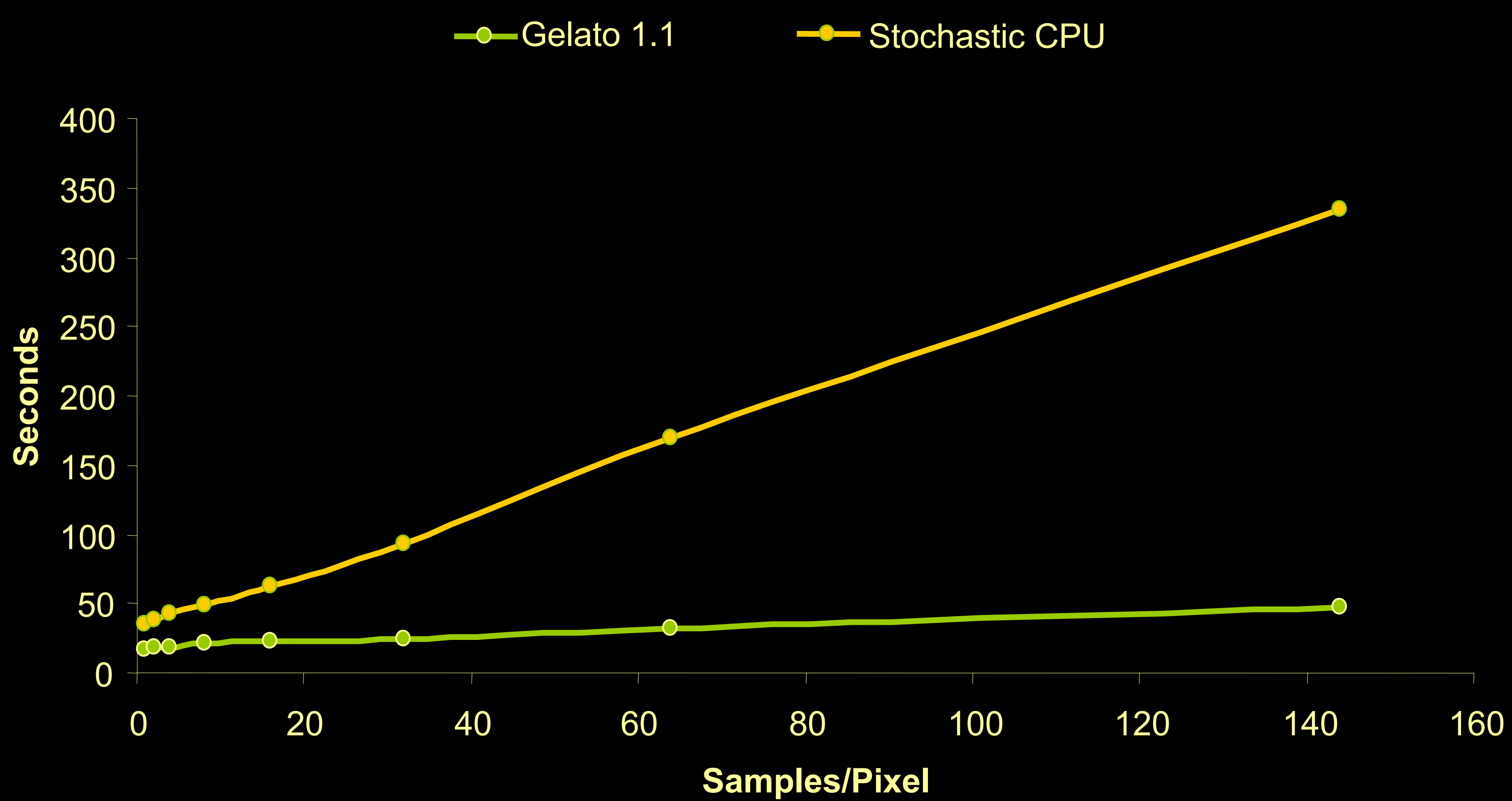
- Spatial AA: Over-sample
  - Two-pass downsampling for filtering
- Motion & DOF: Accumulation buffer
- Transparency: Enhanced depth peeling
- REYES-style geometry processing
  - Parameter-space shading
  - Occlusion query for culling (two types)



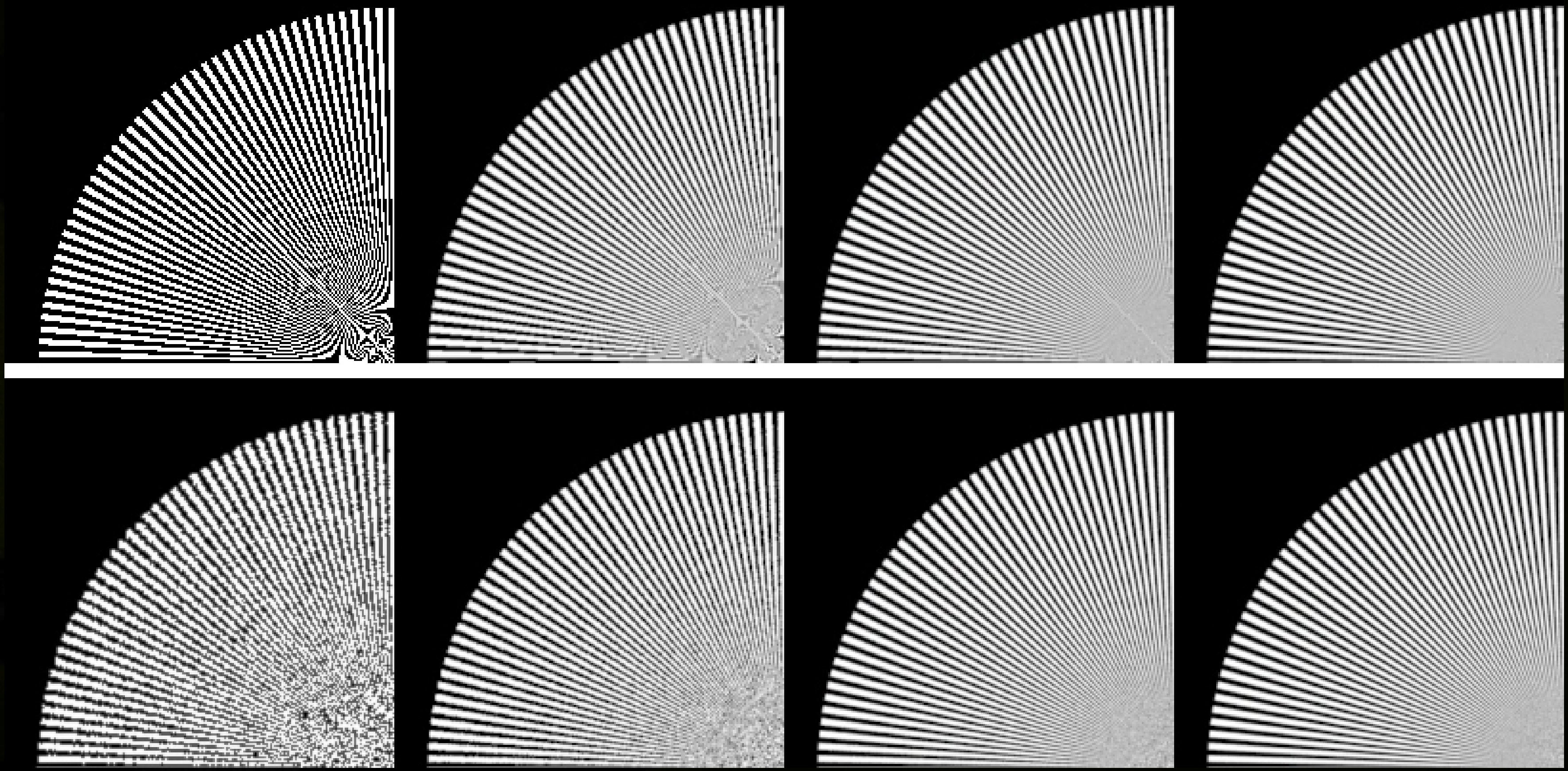
Micropolygons, Pixels & Samples



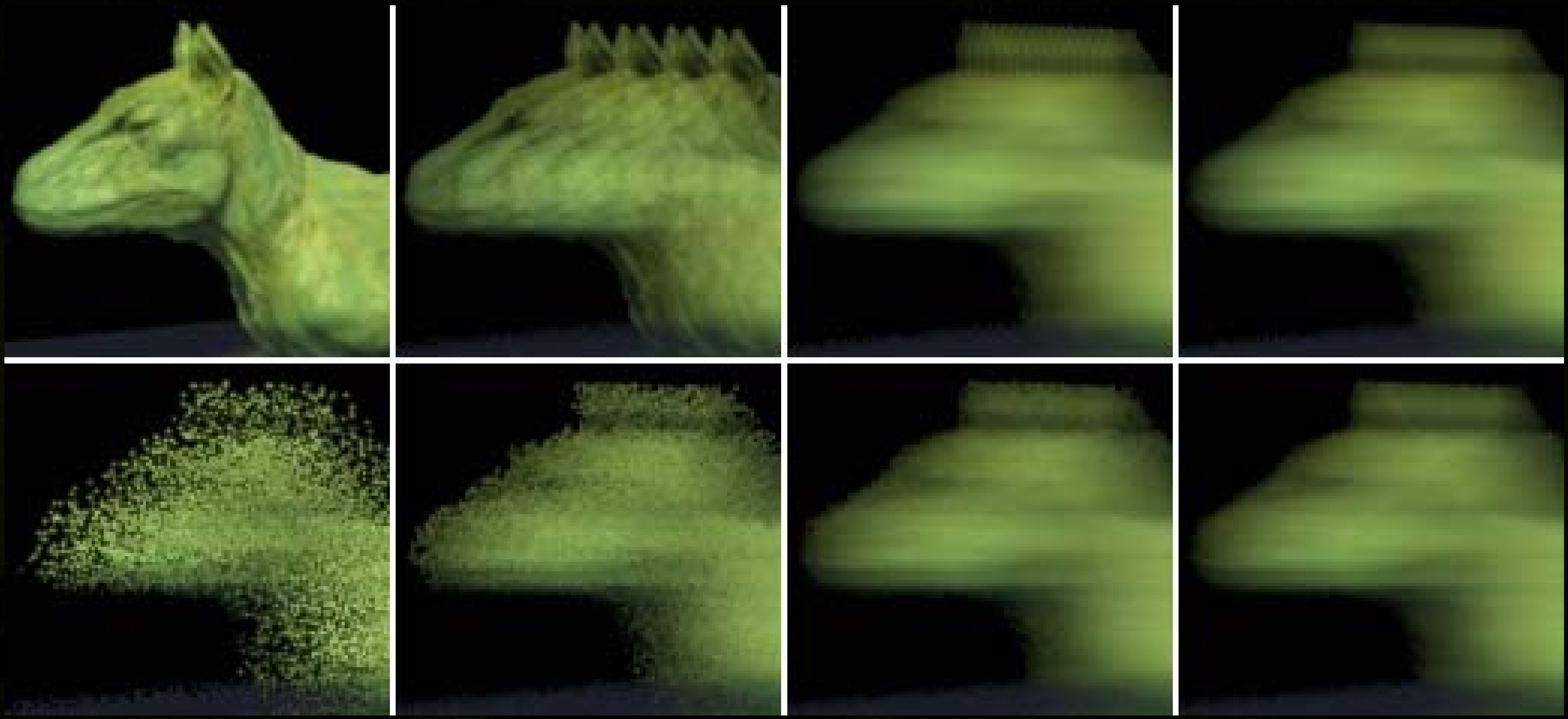
Spatial Samples



Sampling and Filtering

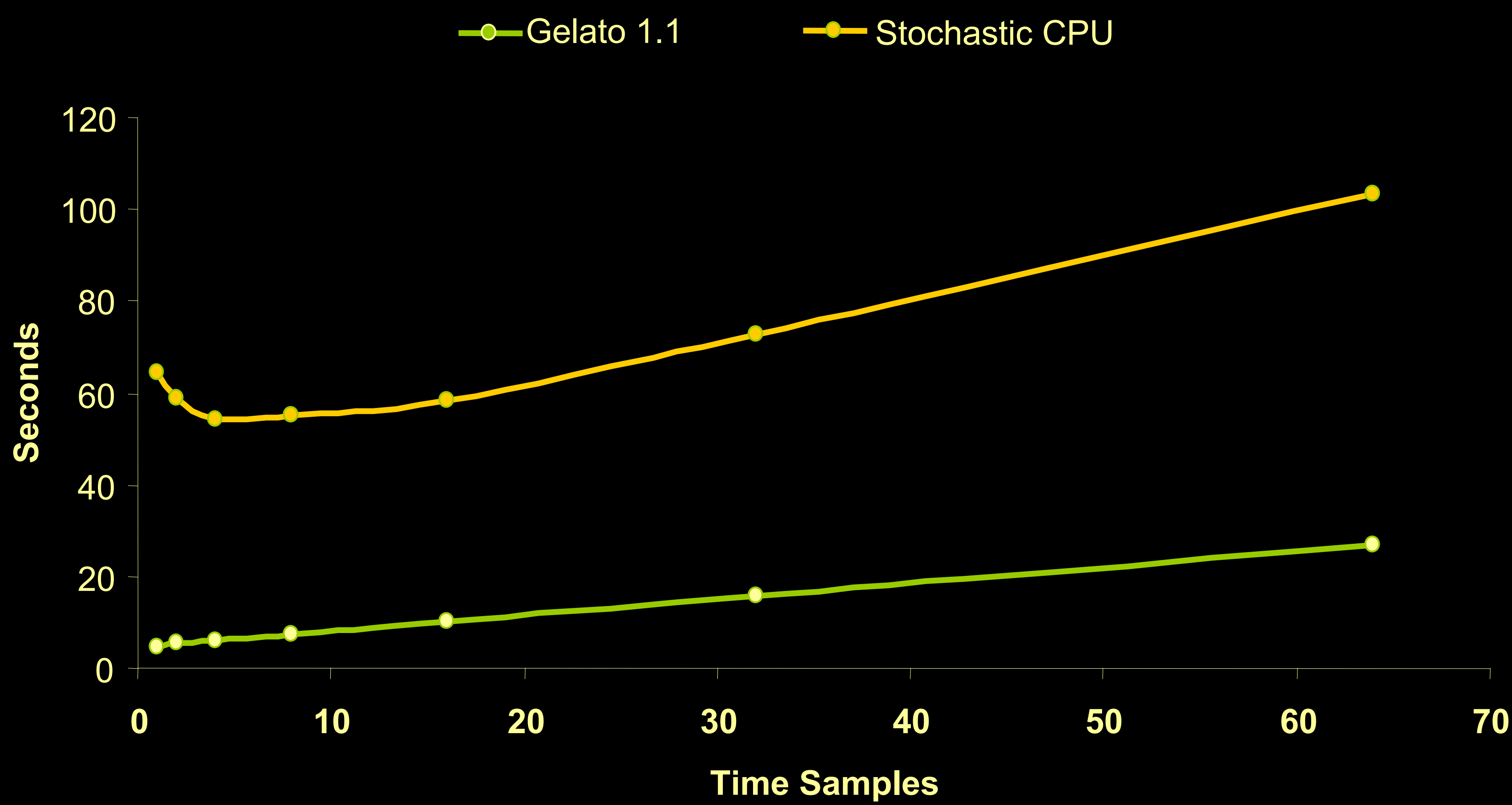


Temporal Samples



model courtesy of Headus

Temporal Samples

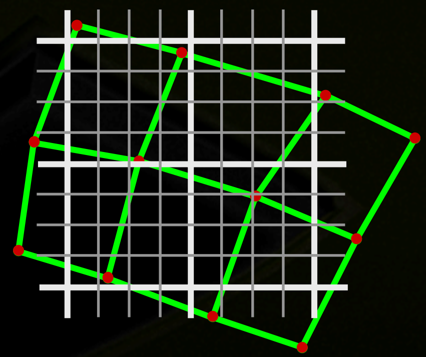


Parameter-Space Shading

- **Overshading:**
  How many pixels drawn per micropolygon?

  64 spatial
  64 temporal
  x ~2 transparency
  ≈ *8000x*

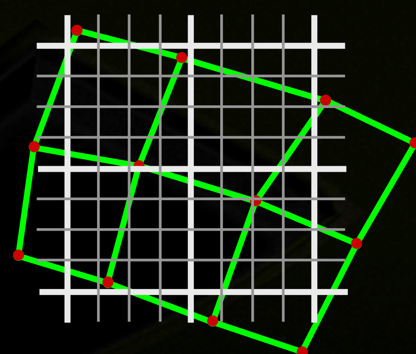
→ ***Decouple shading rate***



(Decouple all rates)

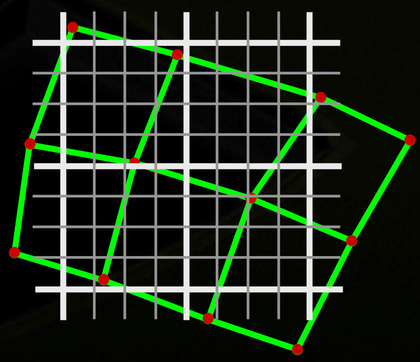
Parameter-Space Shading

- Screen space
  - fixed positions in image
  - regular x,y
  - interpolated s,t,z
- Parameter space
  - fixed(?) positions on surface
  - regular s,t
  - computed x,y,z



Parameter-Space Shading

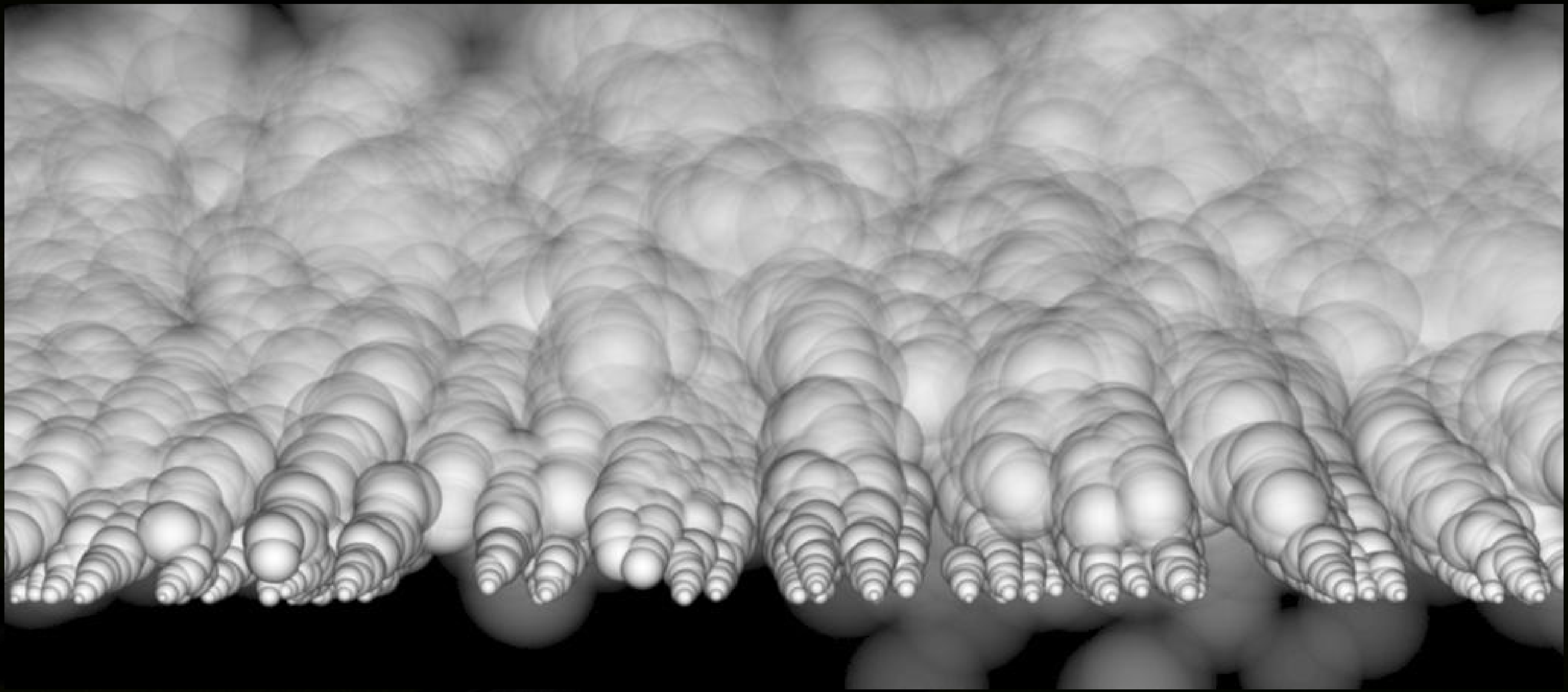
- Derivatives
  - Need for texture mip level
  - Need for procedural AA
  - Estimated by differences
  - Win: neighborhood more regular
  - Win: more stable for motion, deformation
  - Caution: avoid pops between grids
    - smooth derivatives [Gritz]



Parameter-Space Shading

- Shade at lattice of (s,t) values
- Shade once, draw many pixels × many times
- Derivatives more stable during motion

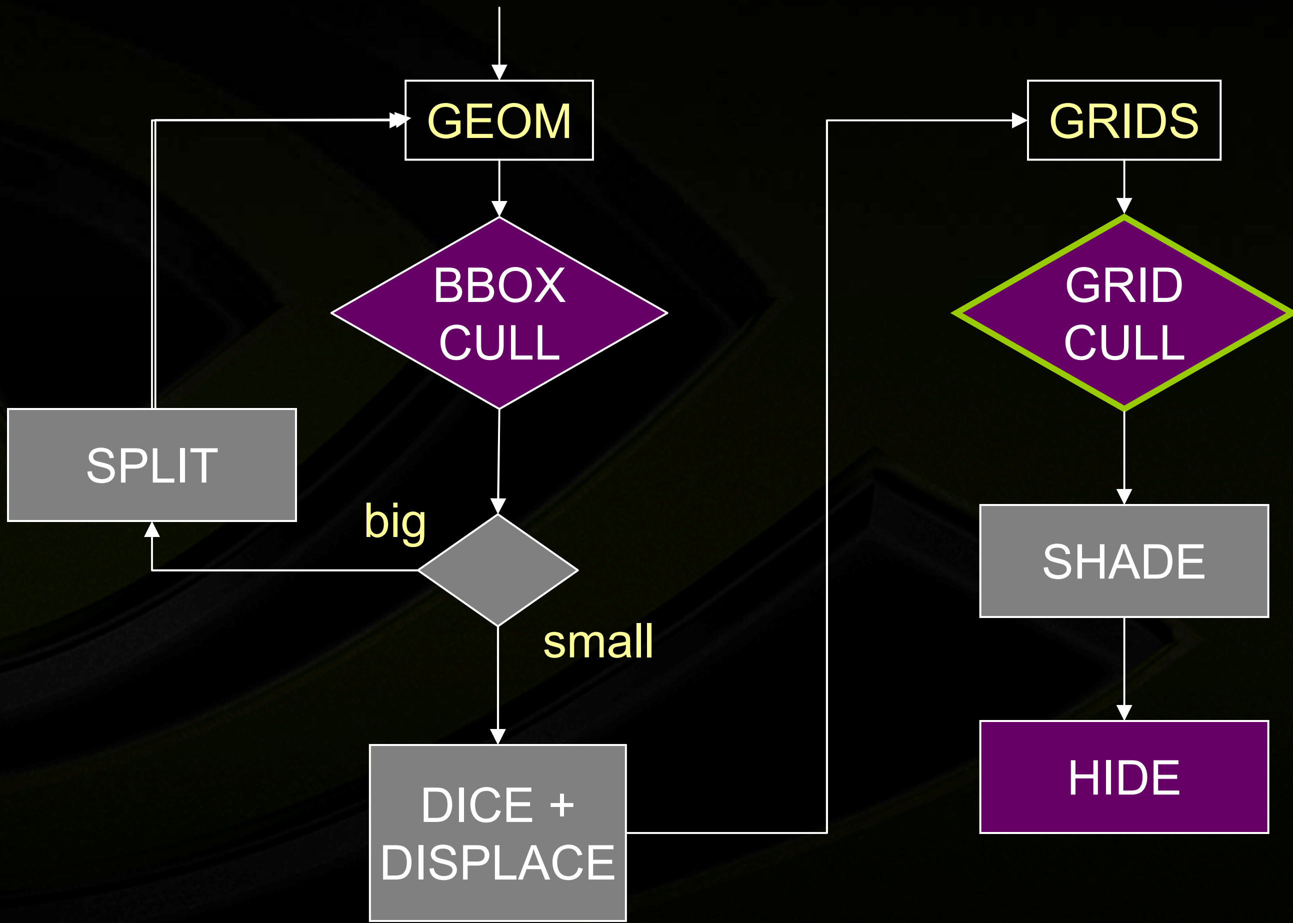
Transparency



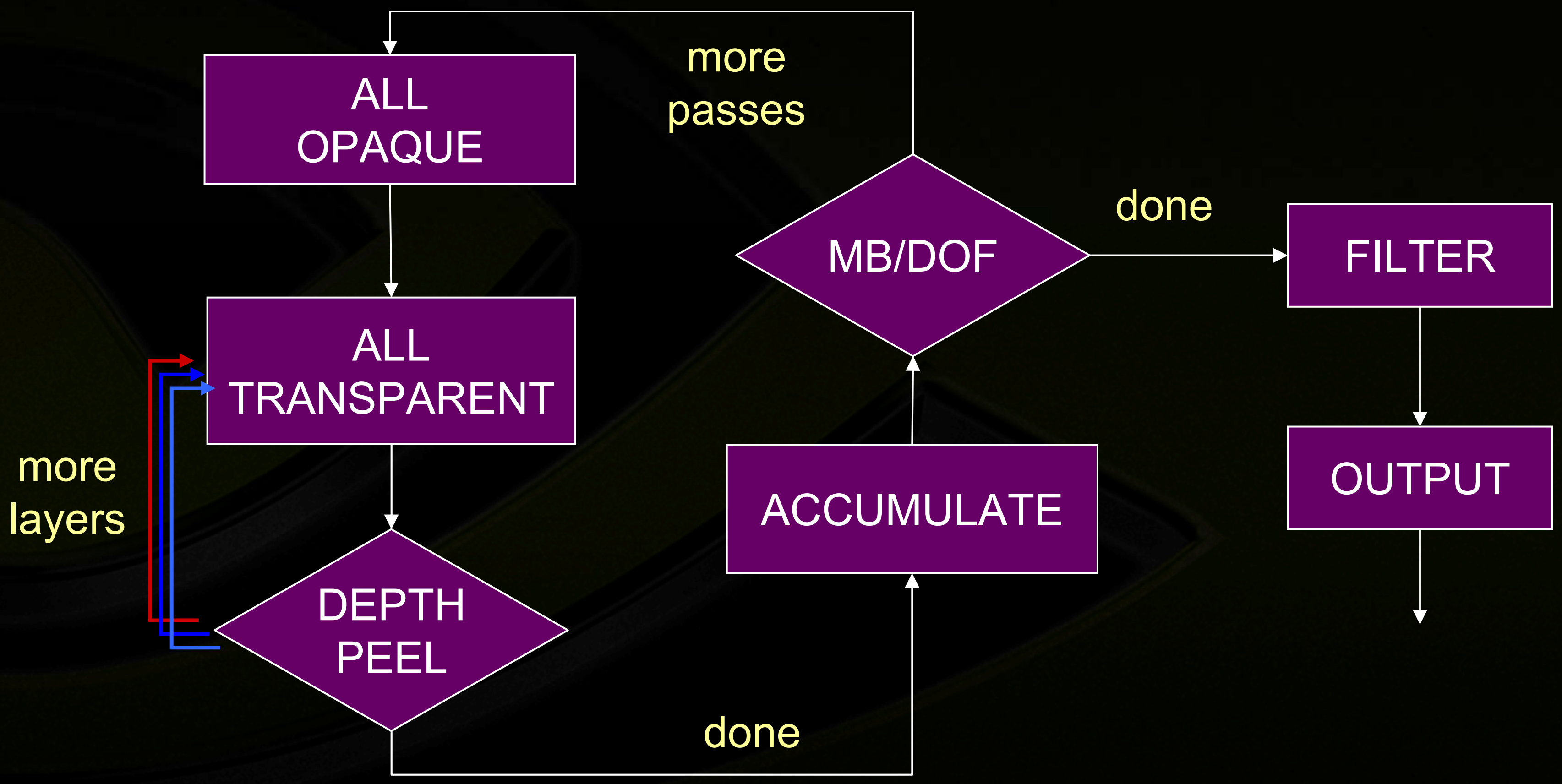
Dicing = Tessellation

- Dicing runs the displacement shader
- Displacement shader may page texture from disk
- Worth doing bbox tests to avoid this
  - bbox expanded by max displacement

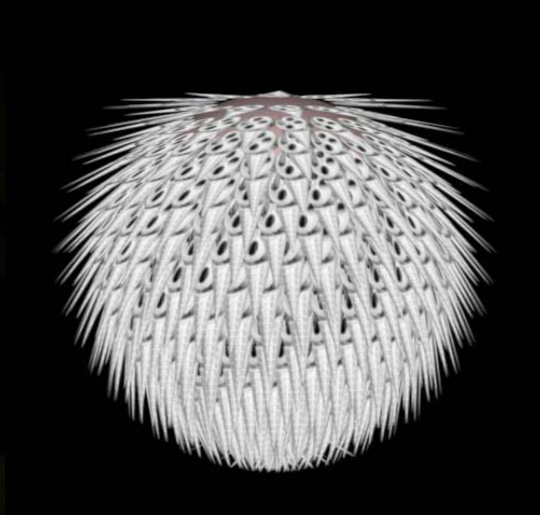
REYES Algorithm

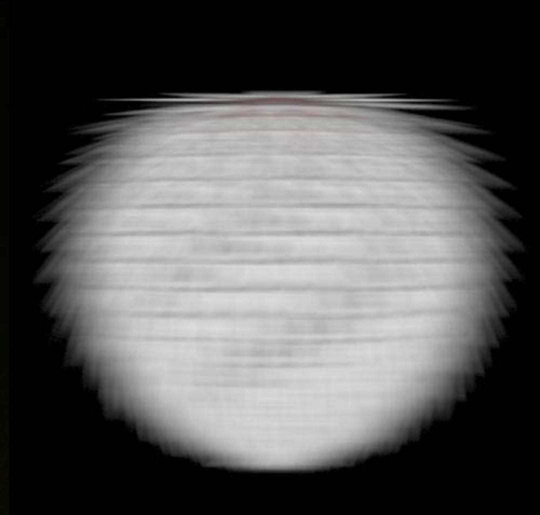


Hiding Algorithm



Poor Performance Cases



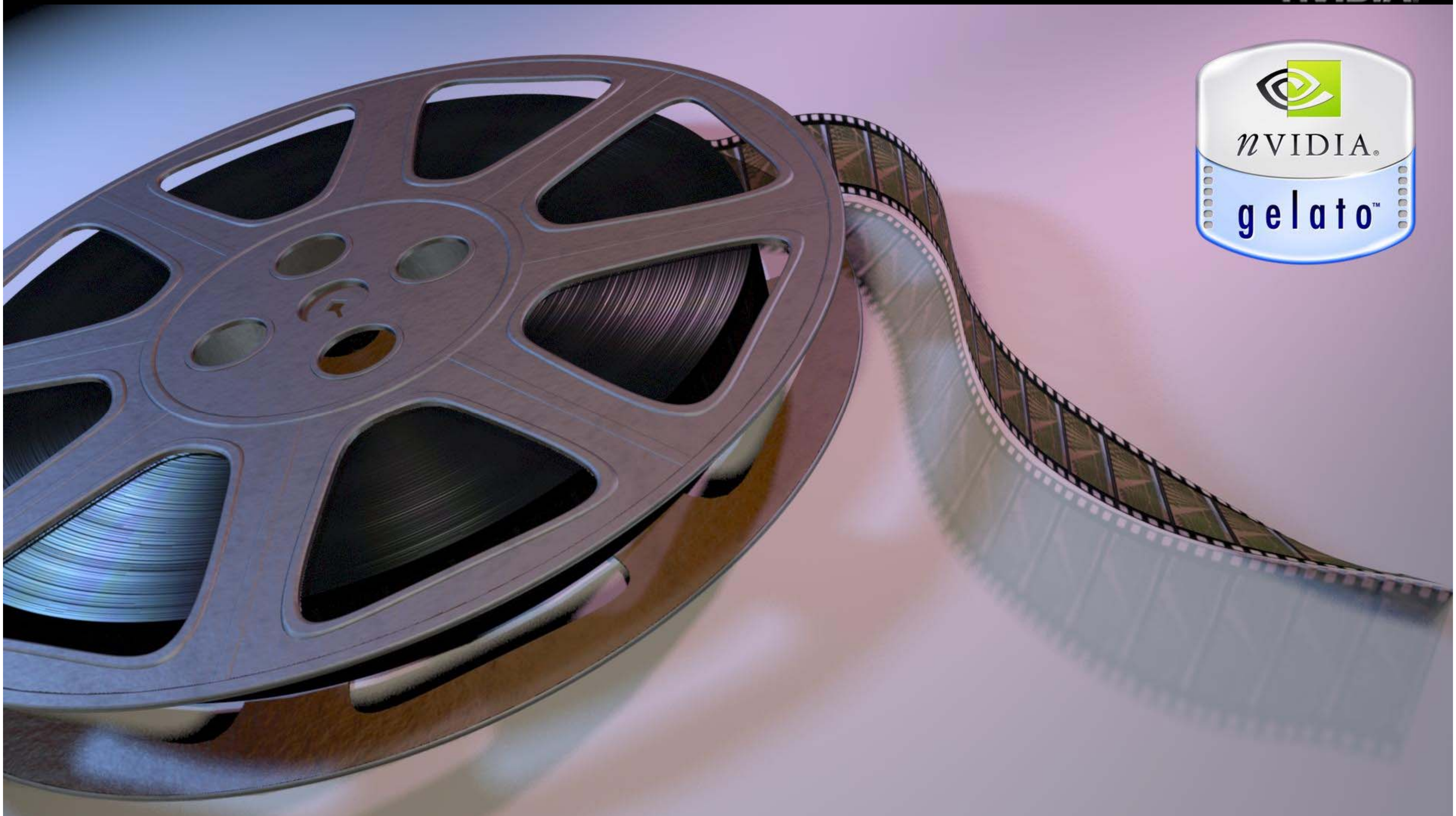


Total Passes = (# Depth Peel) x (# Motion Blur)

High-end Rendering = ?

- No distracting artifacts
  - Space
    - Good filter (2 pixels wide, smooth)
    - Accurate silhouette → Adaptive tessellation
  - Time
    - Motion blur
    - No pops, no chattering
  - Shading
    - AA texture lookup
    - good derivatives
  - Order-Independent Transparency
- Richness
- High level description

NVIDIA



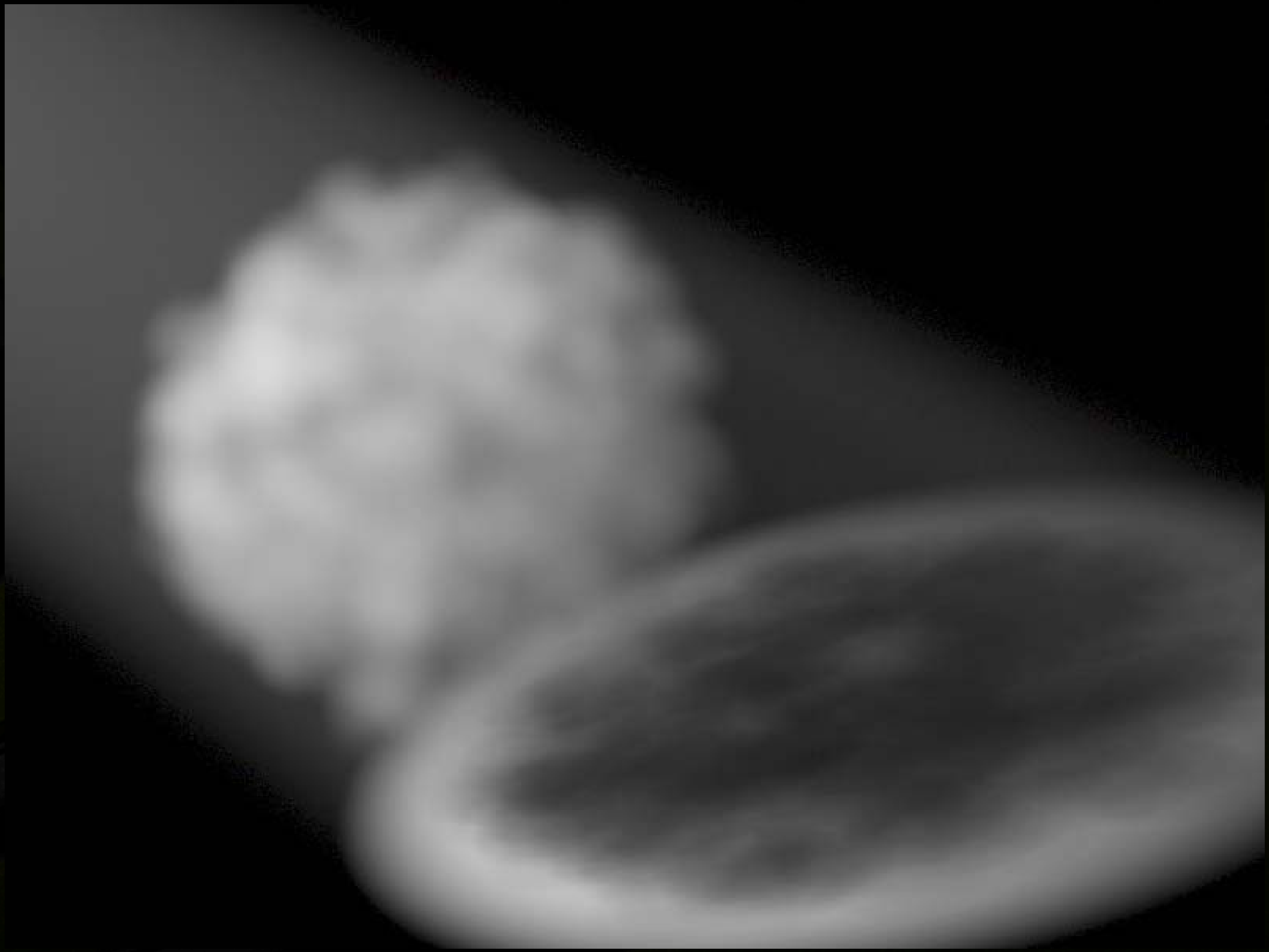
Gelato demo

Gelato = Rendering engine

NVIDIA

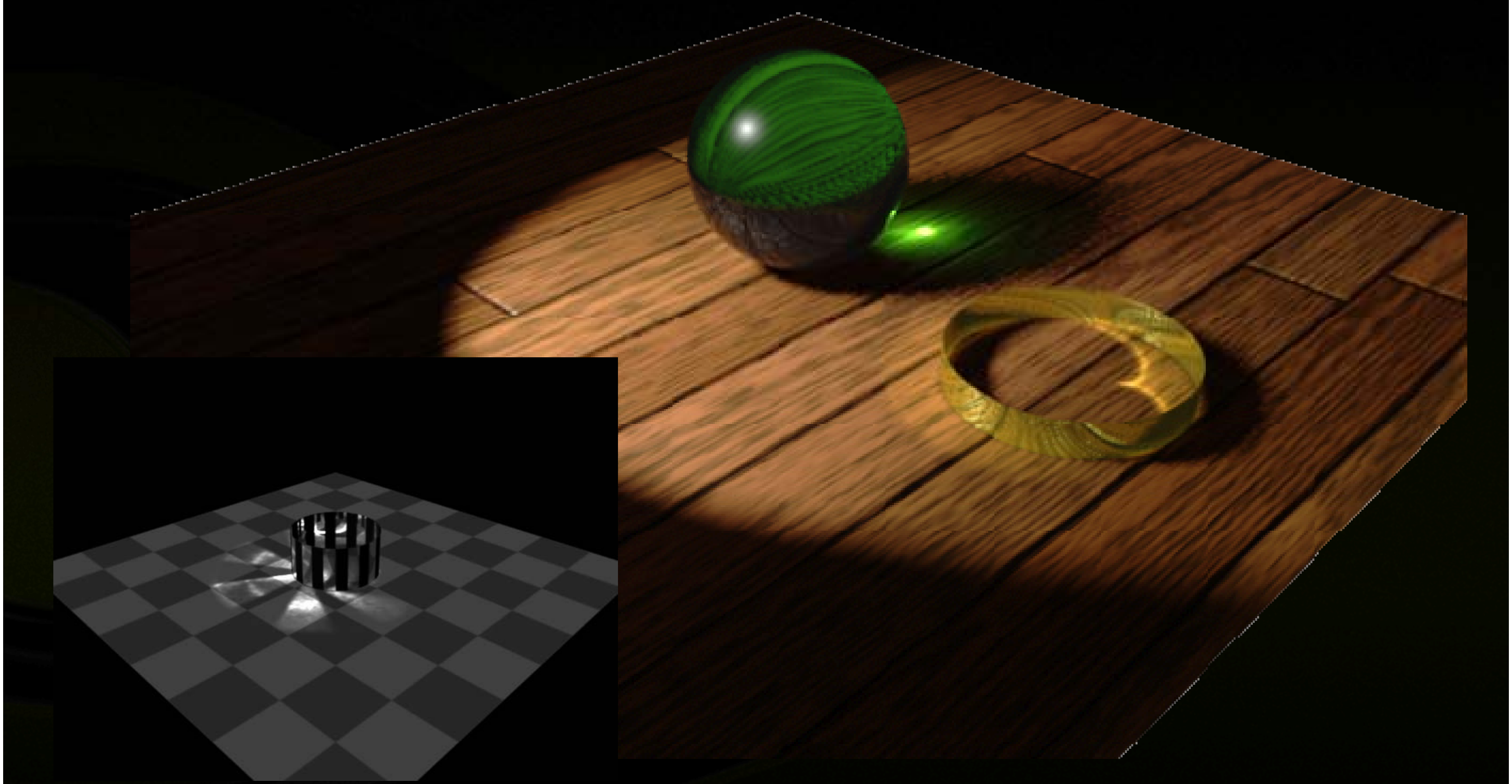
- GPU-accelerated off-line renders, beyond native hardware capabilities
  - all the features & quality
  - half the calories
  - hide HW details
- Free!
  - www.nvidia.com/get_gelato
  - Maya, 3ds max, Python, C++
- Gelato Pro
  - multi-threaded, networked, 64-bit, support, etc
  - relighting (Sorbetto) (free trial)

Gelato : Volume Shadows



Gelato : Ambient Occlusion

Gelato : Caustics



Gelato : Subsurface Scattering

Gelato

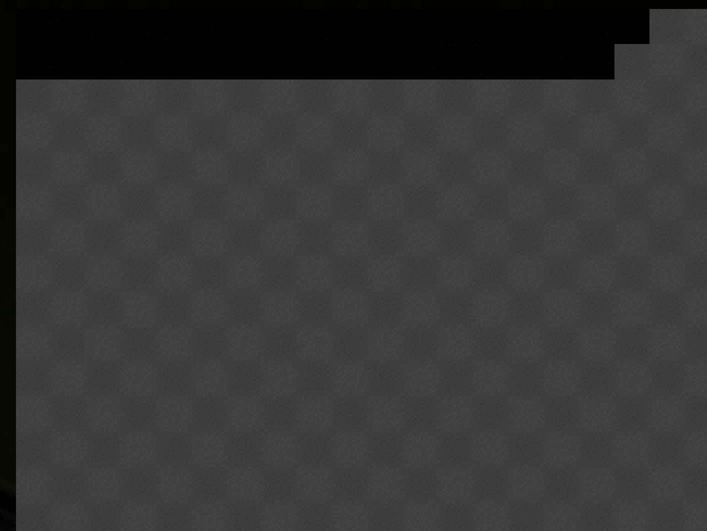
Ethan Summers & Shiew Yeu Loh

Gelato

Frantic Films 2006

Extensions

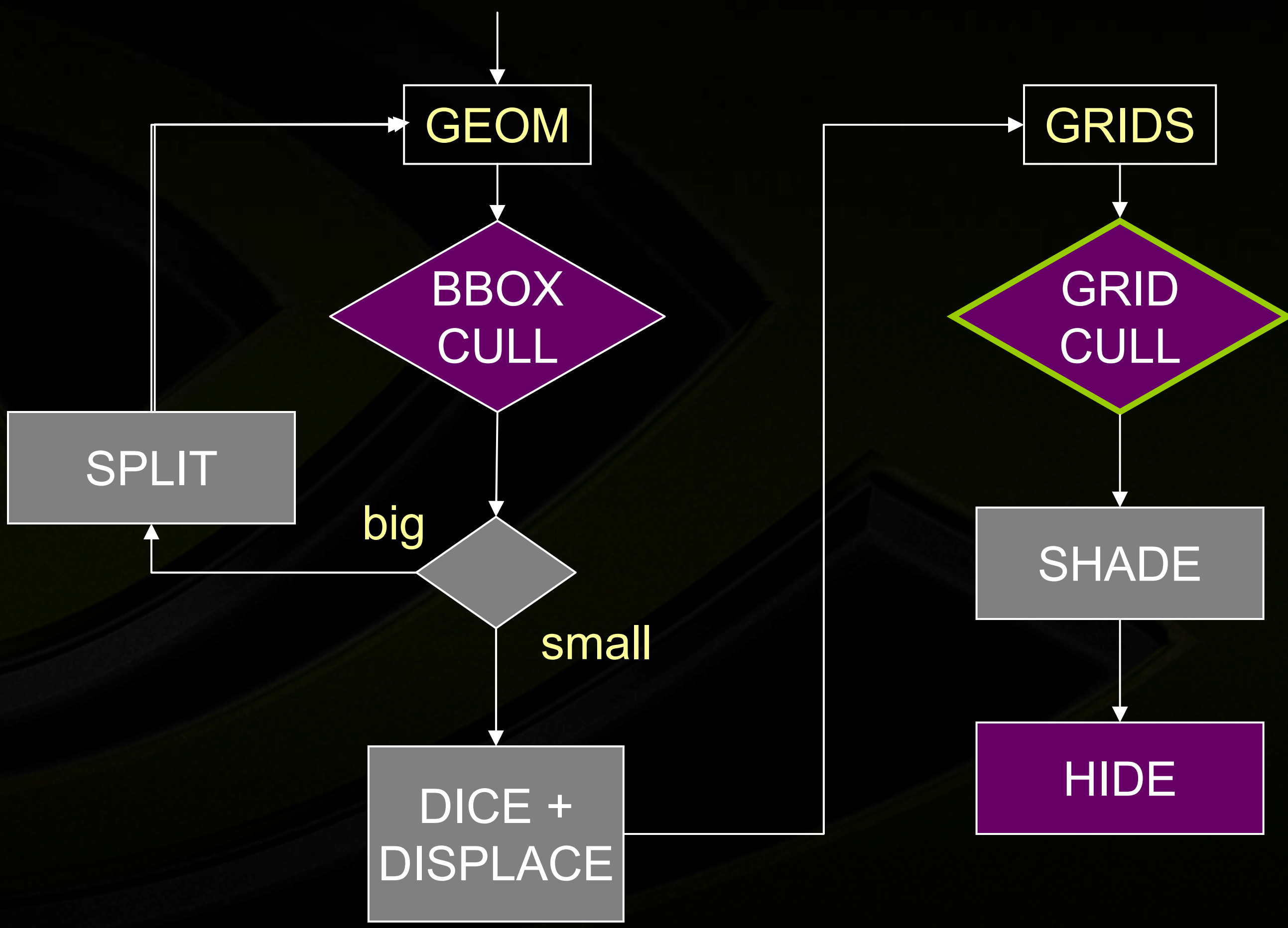
- Depth peel for average-z
- Volumetric shadow maps
- Stereo rendering
- Multi-camera rendering



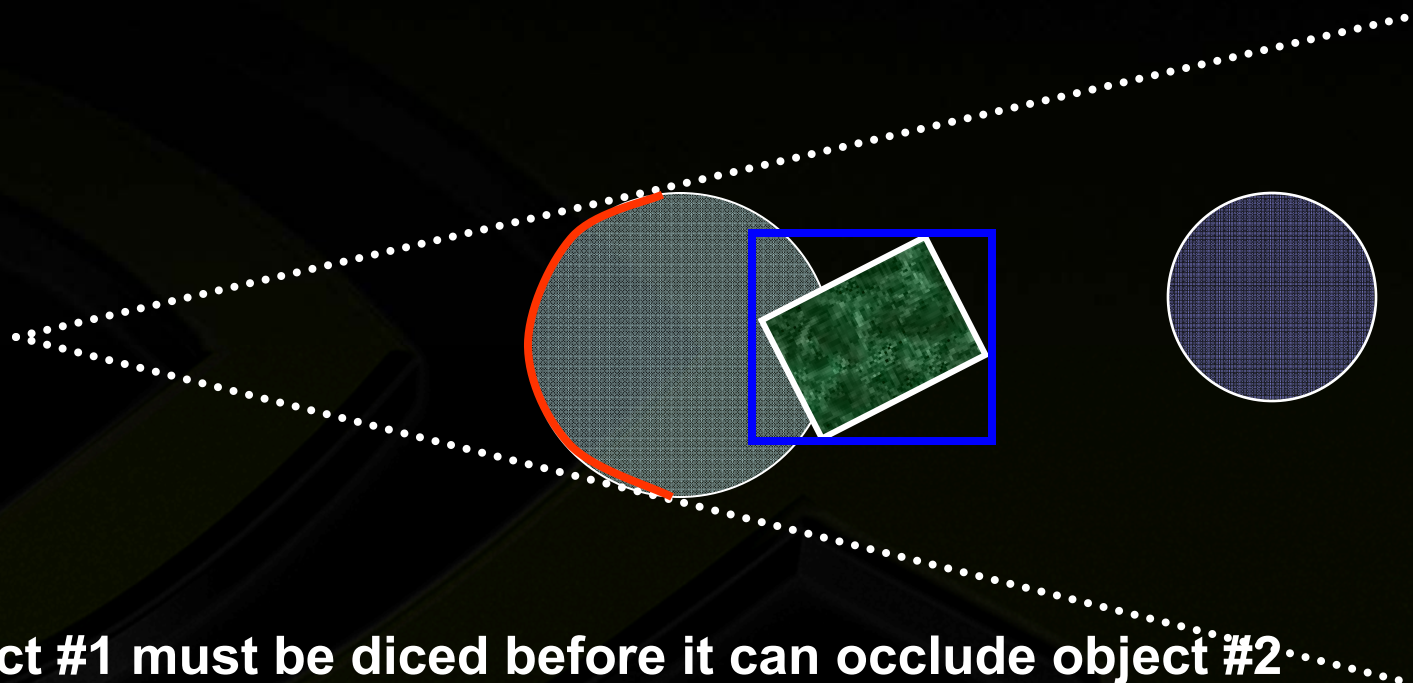
GPU Challenges

- 32-bit floats, including textures
  - slow, not all features work, “temporarily”
- Triangle size
  - At 8x8 samples, 1 shading sample / final pixel, triangle is approx 32 pixels → medium sized
  - But 8x8 enlargement is “temporary”
- Batch size vs culling accuracy

Batch size vs Culling accuracy



Batch size vs Culling accuracy



- Object #1 must be diced before it can occlude object #2
- Bbox test for object #2 must either
  - wait for object #1 to be bbox OQ'd, diced, and drawn
  - or don't wait, and be less accurate → overdice

GPU Challenges

- 32-bit floats, including textures
  - slow, not all features work, "temporarily"
- Triangle size
  - At 8x8 samples, 1 shading sample / final pixel, triangle is approx 32 pixels → medium sized
  - But 8x8 enlargement is "temporary"
- Batch size vs culling accuracy
  - General problem for GPU algorithms (all parallel algorithms?): Result of step 1 could make step 2 more efficient, but then they are serialized.

GPU Mental Model

- No flow control outside a pixel
  - Not much flow control outside a fragment
    - z test, stencil test
- 100x faster at 1% efficiency = no gain
- Small batches bad
  - Because of CPU time, not GPU time
  - Limited draw calls per second
  - OQ latency small for small batches
    - OQ latency ≈ render latency

A Brief History of Coding…

- Von Neumann CPU
- MIMD Multithreading
- SIMD Parallelism
- Future: MIMD+SIMD

A Brief History of Coding…

- GPU's are parallel
  - That's hard to program. But,
- CPU's are parallel now, too
- mCPU: task parallel, memory locality
- GPU: data parallel, memory streaming
- GPU programming is high-performance programming

Game Render vs Film Render

- 0.03 seconds versus 4 hours
- Blinn's Law: "All frames take 45 minutes."

(missing image) (not Gelato)

enterthematrixgame.com/html/screenshots11.html

(missing image) (not Gelato)

from Matrix Reloaded movie

www.hollywoodjesus.com/movie/matrix_reloaded/reloaded5.jpg

Game Render vs Film Render

- GPU is designed for games
  - Large batches and tight inner loop
  - No dynamic allocation
  - No serialized decisions

- But how long does the first frame take?

Rendering the First Frame

Load Data Over Network…

© Blizzard

Rendering the First Frame

Create Internal Objects…

© Blizzard

Rendering the First Frame

- Load
- But during authoring:
  - Build low-res geom from high-res geom
  - Preprocess for visibility, displacement, …
  - Bake out lighting textures

Game Render vs Film Render

- Every film frame is a "first frame"
  - Start with high-res geom
  - Start without pre-computed visibility, displacement, lighting, …
  - (Also animation (IK, skin, …))
  - When these stop changing, artist stops rendering
    - Render once → deliver film
- How can we have fewer first frames?

Relighting

- Cache expensive computations
  - Texture operations *(e.g. shadow maps)*
  - Complex math *(e.g. noise)*
  - Ray queries *(in shader)*
- Accelerate the major lighting tasks
  - Move a light
    - Recomputes depth maps
    - Reshades affected surfaces
  - Adjust shadow parameters

Sorbetto = Re-rendering engine

- Start with full Gelato render
- Change any light parameter
- Final pixels – Not an approximation!
  - same features (motion blur, transparency, …)
  - same assets (shaders, models, …)
  - ≈10x faster full frame time
  - much faster Time To First Pixel
- Adjust hider parameters
  - depth-of-field
  - stereo
  - very fast

Relighting demo

Conclusions

- High-quality hiding on GPU
  - Bbox cull → less dicing
  - Grid cull → less shading
  - Small batches → better culling
  - Regular sampling ok at modern rates (claim)
- Parameter-space shading
  - Better for multi-pass, over-sampled
  - Better for derivatives (claim)
- The “first frame” problem

Acknowledgements

- Gelato Development Team
  Larry Gritz, Dan Wexler, Eric Enderton, John Schlag, Philip Nemec, Jonathan Rice, Eduardo Bustillo
  Interns: Sharif Elcott, Jared Hoberock
- NVIDIA Software, Architecture & DevTech

Fleeting Image Animation