The Ins of Production Rendering
At Animal Logic

Daniel Heckenberg
R&D Supervisor - Graphics
• Detail
• Complexity
• Reference structure
• Motion
• Camera & Imaging

• Interactivity
• Edits
• Upstream rendering
SOME HISTORY
The RenderMan Interface (RI)

- Introduced in 1988 (Pixar)
- Hierarchical state model
- Programmable shading (RSL)
- Serialization (RIB)
- Time-sampled state (motion)
- Structure through referencing
- Delay-loaded procedural
- Late-stage edits (RIFilter)
MayaMan (Animal Logic)

- Translation from Maya to RI
- Time sample animated state
- Stream from Maya to RI on disk
- Renderer streams from RI on disk

- Proxy objects in DCC application
- Geometry caches with direct translation
- RIFilters for edits without retranslation
MayaMan (Animal Logic)
MayaMan (Animal Logic)
Streaming Rasterization

- Spatially coherent scene access
- Ordered passes (shadow maps)
- Working set typically smaller than full scene
- “Shade once”
Interactivity

- Path tracing: single pass
- Working set typically entire visible scene
- Memory pressure, but suits editability

- Glimpse Scene Stream, **GSS**
  - ref: Arnold Scene Source (Solid Angle)
  - ref: Nodal Scene Interface (3Delight)
  - ref: Riley (Pixar)
  - ref: Hydra (Pixar)
Glimpse Interactive Model

- **GSS** API supports in-memory editing and updates
- DCC native objects bridged to GSS objects
- propagate only fine-grained changes
- Proxy objects in DCC to avoid duplication
- GSS and Glimpse renderer for viewers and editors
Glimpse

Maya → GSS → GlimpsePT

Σ
Glimpse

GlimpseMaya

Maya

GSS

GlimpsePT

Glimpse
Memory & multiple representations

1. Serialized on disk (e.g. Alembic file)
2. Geometry cache (e.g. Alembic in RAM)
3. DCC representation (e.g. Maya mesh)
4. GSS representation
5. Render representation (e.g. subdivided mesh)
6. Acceleration structure (e.g. Glimpse BVH)
Principles to reduce duplication

- Prefer **GSS**
- API must be rich and editable, e.g. scene hierarchy
- 1:1 bridge objects when using DCC editing
- Fine grained incremental edit propagation
- Discard intermediate data, but maintain structure
Reducing latency

- Minimise work required before render!
- **GSS** as primary scene source
- Multithreading
- Avoid stack / context paradigm of **RI**
- cf bindless OpenGL
- Parallelism through structure
  - Objects
  - References (files)
  - Procedurals
Instancing

- Ray-tracing’s great party trick
- Minimises memory and work

- Late-stage deduplication vs explicit?

- **GSS**
  - nested instancing
  - hierarchical state
  - path-based instance overrides
Expressiveness

- Concise edits are efficient for users
- Direct representation usually simpler and faster
- Balance frequency and complexity of operations

**GSS**
- hierarchical state
- path-based instance overrides
- defer some user operations to higher layers
  - e.g. wildcard edits
GLIMPSE AND USD
From GSS to USD

- USD is more general and richer
- High performance load and processing
- USDC (Crate)

- Some areas not yet standardised / supported
- Render, imaging, camera settings
- Materials and layered assignment
- Proceduralism
USD and Hydra
Maya and USD and Hydra

AL_USDMaya

Maya

USD

Hydra

RenderDelegate

MayaToHydra
Current Structure

- Hydra Render Delegate?
- AL_USDMaya
- Filament (procedural USD editing)
- GlimpseUSD

- Custom schemas and approaches
  - Render, imaging, camera settings
  - Materials and layered assignment
  - Proceduralism
Maya and USD and Glimpse

- **Maya**
- **USD**
- **Filament**
- **Hydra**
- **Storm**
- **GSS**
- **GlimpsePT**

Diagram:

- AL_USDMaya, Filament
- Hydra
- Storm
- GSS
- GlimpsePT
Challenges ahead

- USD / Hydra support for final frame rendering
  - Hydra Render Delegate
  - Materials and layered assignments

- USD proceduralism

- Memory
  - Speed of some edits
ACKNOWLEDGEMENTS

Glimpse Team
Michael Balzer, Curtis Black, Luke Emrose, Erik Pekkarinen, Jakub Jerzierski

AL_USDMaya Team
Eoin Murphy, Rob Bateman, Eddie Hoyle, Fabrice Macagno

Pixar USD Team

Luma Pictures

Weta Digital
Johannes Hanika, Luca Fascione