Schedule

• 9:00am
  - Introduction to OpenVDB, Ken Museth (DWA)

• 9:20am
  - OpenVDB in Houdini, John Lynch (SideFX)

• 9:40am
  - Adoption at DreamWorks Animation, Jeff Budsberg (DWA)

• 10:10am
  - Advanced applications of OpenVDB in production, Dan Bailey (DNeg)
Course Material

• Main site: http://www.openvdb.org
  - Course slides from 2015 & 2013
  - Coding cookbook and FAQ
  - Example files: vdb & Houdini hip
  - Google group: “OpenVDB Forum”

• Technical paper on VDB: http://ken.museth.org
Dry Facts

• History of development
  - Starting in 2007 (DB-Grid/DB+Grid/VDB)
  - Open source in 2012 (v0.9)

• OpenVDB core developers
  - K. Museth*, P. Cucka, M. Aldén* and D. Hill

• Source code
  - Tar-balls: http://www.openvdb.org
  - GitHub: https://github.com/dreamworksanimation/openvdb_dev

• License
  - Mozilla Public License version 2 and CLA
Commercial Adoption

- Pixar's RenderMan
- MODO
- REALFLOW
- Guerilla Render
- arnold
- Solid Angle
- Houdini
- 3D Animation Tools
- 3delight
- ChaosGroup
- V-Ray
- clarisse iFX
- STOKE MX
- octaneRender
- Maxwell Render
- Autodesk Maya

VES Reference Platform
Library Versioning

- **Patch:**
  - No change to API, file format or ABI of Grid or its member classes

- **Minor:**
  - Change to API but not Grid ABI; backward-compatible file format

- **Major:**
  - Change to ABI of Grid or non-backward-compatible file format

- Library is namespaced on the complete version number
  - No release guarantees complete ABI compatibility!
  - Guaranteed Grid and file compatibility with fixed Major
Terminology

• Voxel \[= \text{Volume} + \text{Pixel}\]
  - Smallest addressable unit of index space
  - Resides at the leaf node level

• Tile
  - Larger constant domain of index space
  - Resides at the upper (non-leaf) tree levels

• Active state
  - All values (voxels and tiles) have a binary state
  - Interpretation of state is application-defined
Properties of VDB

- Unbounded
- Sparse
- Fast access
- General

7897 x 1504 x 5774
Technical Features

- **Unique variant of B+Tree**
- **Inverted tree traversal**
- **Bit-mask techniques**
- **Hierarchical acceleration**

[K. Museth, SIGGRAPH / ACM TOG, 2013]
CPU Analogy

Memory Hierarchy
- Memory
  - L3 Cache
  - L2 Cache
  - L1 Cache
  - Core

Capacity

Latency

VDB Hierarchy
- Root node
  - Internal Node 1
  - Internal Node 2
  - Leaf Node
Random Access: Top-Down

tree.getValue(x,y,z)=?
int cacheKey[3] = {x & ~((1 << sLog2X) - 1),
                  y & ~((1 << sLog2Y) - 1),
                  z & ~((1 << sLog2Z) - 1)}
Changes since 2013

Version 3.1.0 - In development

New features:
- Added tools::DensityAdvection for sparse advection of non-level-set volumes.
- Added tools::extractEnclosedRegion to detect topologically enclosed (watertight) exterior regions (cavities) that can arise as the result of a CSG union operation between different level set shapes where at least one of the shapes has a concavity that is capped.
- Added a preconditioned conjugate gradient solver.
- Added a Poisson solver for incompressible fluids that operates on a given domain. Often procedural animation of objects (e.g. characters) interacting with liquid will result in boundary conditions that describe multiple disjoint regions: regions of free surface flow and regions of of trapped fluid. It is this second type of region for which there may be no consistent pressure (e.g. a shrinking watertight region filled with incompressible liquid). The unit test TestPoissonSolver::testSolveWithSegmentDomain demonstrates how to use tools::extractEnclosedRegion in conjunction with the Poisson solver.
- Added util::PagedArray, a thread-safe, dynamic linear array data structure with fast O(1) value access (both random and sequential).
- Added LeafNode::Buffer::data(), which provides direct access to a leaf node’s voxel value array, avoiding out-of-core overhead.
- Added tools::Sampler, which provides a unified API for both staggered and non-staggered interpolation of various orders.
- Added equality and inequality operators to Metadata and MetaMap.
A Few Highlights …

- Accelerated ray-marching (HDDA)
A Few Highlights …

- PointPartitioner
- PointIndexGrid
- PointDataGrid
A Few Highlights …

- Delayed loading (out-of-core processing)
- Blosc LZ4 multi-threaded file compression
- Python scripting support
- Poisson solver ($\nabla^2 \mathbf{x} = \mathbf{b}$), segment trapped liquids
- Density and non-level-set advection
- Level set morphing, measures, clipping, and sphere packing
- Diagnostics tools for level sets, Fog volumes, etc
- Maya nodes (to/from poly, read/write, filter, vis, convert, transform)
- Numerous optimizations, eg Grid copy and topology operations
Houdini SOP Nodes

Advect Density, Advect Level Set, Advect Points, Analysis, Clip, Combine, Convert, Create, Diagnostics, Fill, Filter, Filter Level Set, Fracture, From Particles, From Polygons, Metadata, Morph Level Set, Noise, Occlusion Mask, Platonic, Prune, Rasterize Points, Ray, Read, Rebuild Level Set, Resample, Sample Points, Scatter, Sort Points, To Polygons, To Spheres, Transform, Vector Merge, Vector Split, Visualize, Write
Looking further ahead...

- MultiResGrid: MipMaps for VDB
- Liquid particle decimation
- Improved liquid particle surfacing
- Improved adaptive mesh extraction
- More multi-threading of grid methods and tools
- More focus on fluid related tools
- More cloud modeling tools
- Continue work on particle storage and processing
- Improved adaptivity of tree structure
- C++11 & cleanup of API
- Vectorization: SSE/AVX
We’re hiring!

Ken.Museth@DreamWorks.com