SONY
make.believe

Production Renderering

Trials and tribulations generating those pixels

Jonas Gustavsson : Advanced Application Labs
Technology Research
Sony Mobile Communications

How did I get here?

- Started out writing renderers in 1983 on a Vic64
- Started doing Commercials in 3D around 1988
- Interned at Colossal Pictures worked on Aeon Flux(Liquid Television)
- Went to work at Xaos and did TD tools on “Toys”
- Spent a bunch of years freelancing in the US, UK, Japan on film, TV, Games, UI´s and real-time graphics
- Started a company with a few friends doing custom software
- Did tools and effects for Digital Domain, Rhythm & Hues, Glassworks, MPC

How did I get here?

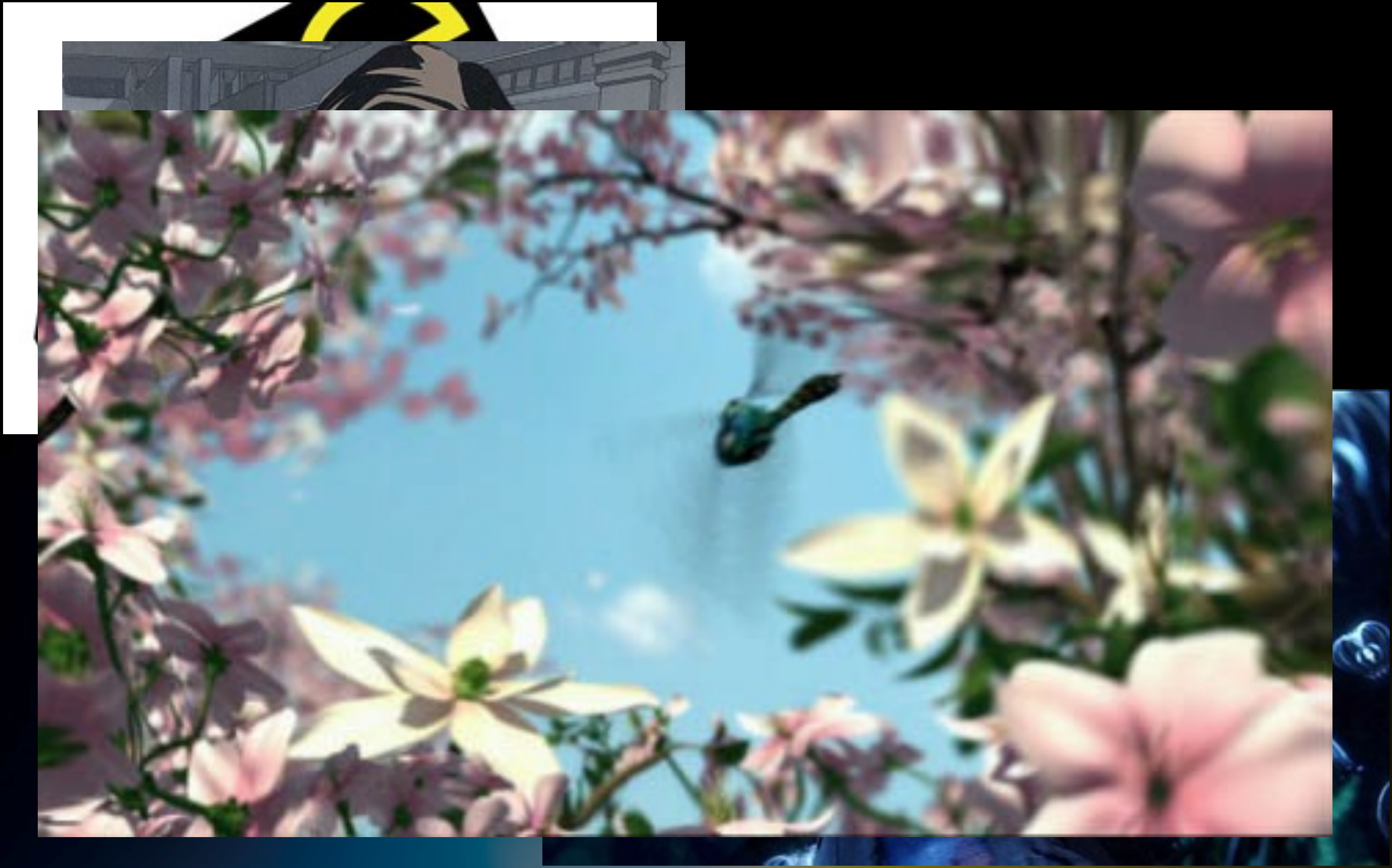
- Company assets and people(including myself) bought by NewTek, makers of LightWave3D
- Ran product development for LW3D for +5 years with Jay Roth(Electric Image, Play, Sony Socratto) and Mark Granger / Allen Hastings doing the renderer
- Moved to UbiSoft to do Animation and Motion Capture Supervision
- Moved to Sony Mobile Communications Technology / Research

Work Samples

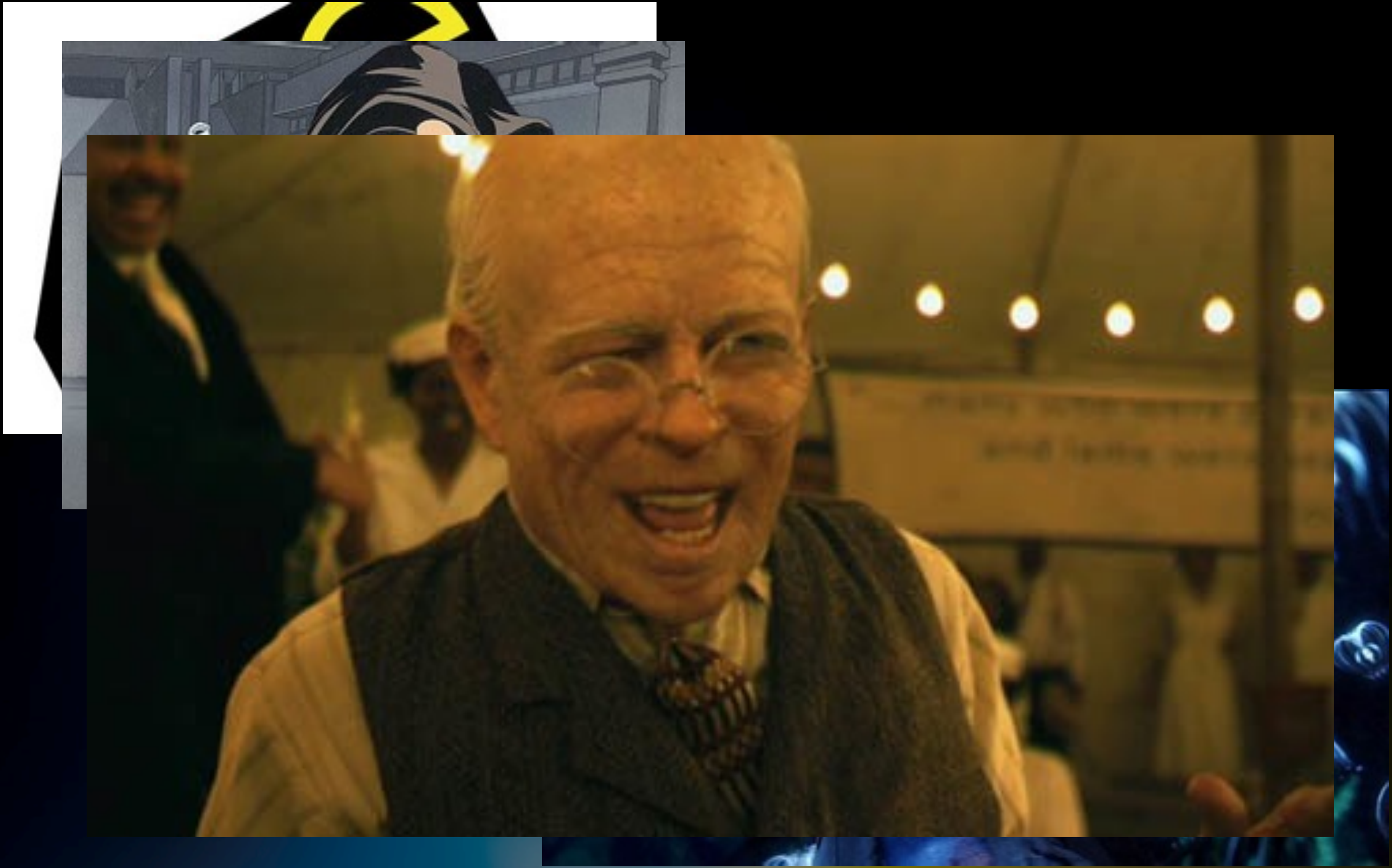
Work Samples

Work Samples

Work Samples



Work Samples



Work Samples

Work Samples

SONY
make.believe

Work Samples

NewTek LightWave 3D 10.0 Demo Reel

Personal view on production renderers

- The basic foundation of a Production renderer haven't really changed that much over the last 10 years, while there have been improvements in how they work the focus have moved towards optimizing the pipeline instead of the actual render algorithms
- A few major building blocks to create a working production renderer:
  - Rendering component
  - Shading component
  - Filtering system
  - Output system
  - Acceleration structure

Renderer

- Out of the rendering algorithms out there most tend to move towards ray tracing or Global Illumination and ray tracing:
  - Mental Ray, Arnold, Vray, Kray, Brazil, C4D, Mantra, LightWave3D, Messiah, Modo, Maya all use this in different forms to render
  - Renderman uses a hybrid Reyes / ray tracing approach
  - Scanline have for the most part fallen out of fashion these days same goes for ray casting outside of basic volumetric renderers. Although some have used ray casting and global illumination quite effectivly
  - One interesting approach that have been coming up lately is various forms of Splatting approaches(photon, marching, casting)

Shading

- While it might be more efficient to directly drive shading through a native component/node the use of a shading language will always yield better long term results
- There are numerous variations of shading languages that can be separated into two groups
  - **Realtime**
    - RLSL
    - HLSL
    - GLSL
    - Cg
  - **Offline**
    - RSL
    - OSL
    - VEX
    - MSL
  - they all tend to solve the same problem

Shading

- The problem they are all trying to solve is HW/SW abstraction in one way or the other and usually goes through a compile stage(LLVM, Clang, …) this is where the continuous improvement compared to native shader modules comes in.
- They also tend to have some variation of the following shader types:
  - Light Source
  - Surface
  - Displacement
  - Deformation
  - Volume
  - Image
  - Lens
- Most systems tend to add BSDF, BSSRDF, Emissive, Reflective and a lot of Noise functions

Filtering

- Separating texture and render filtering is advisable given that the texture values are known ahead of time while render samples happen at runtime. This also ties into texture cache solutions
- Pre filter sampling or average sampling usually based on illumination differences over the pixel to eliminate flickering while sampling varying brightness over the same pixel, also known as Super Sampling
- Basic jittered sampling is by far the easiest to accomplish but also the most error prone
- Stochastic sampling is probably the highest quality but suffer from IPR issues in a commercial environment
- Possion disk sampling can be high quality but suffer from higher computational load
- Another option would be to use iterative fixed sampling pattern, blue noise, ellipsoid weighted average

Output

- This would be the render out put backend that is becoming extremely important in today's production environment
- In most cases you need the following buffers to enable a good workflow with Comp:
  - Beauty pass
  - Alpha / Occlusion
  - Specular
  - Diffuse
  - Reflection / Refraction / Translucency
  - Shadow
  - Primary illumination per light source
  - Secondary illumination both Ambient and bounced light

Acceleration

- Given the load on a render system in production one of the most important parts is the method selected for acceleration
- Most common approach would be a KD-Tree (Ingo Wald´s implementation is good) especially with Ray tracers, but one can go with Uniform Grid, BVH, MLRTA
- After you have your general structure we can further optimize by using
  - ray bundles(Fat Rays)
  - reshaping the acceleration structure to be more in line with eye rays(simplifies the ray intersection)
  - add multiple acceleration volumes and cloning(helps with motion blur/ enables easy instancing)
  - group/merge/split acceleration volumes to assist with temporal changes(cache a group of objects that move in the same direction instead of recalculating)
  - Add secondary Uniform grid / BVH structure to handle Ray caches for Bucket rendering

Rendering at Sony Mobile Communications

- Most of the rendering work within Sony Mobile is in the real-time space and using the basic scan line HW pipeline found in modern Mobile GPU´s
- Shader work tend to be the overwhelming majority of work done in 3D including:
  - Image processing
  - Encoders
  - Post process
  - Various interface elements
- At research level we are experimenting with different real-time rendering approaches outside of the basic HW pipeline including real-time ray tracing, basic photon splatting approaches.

Rendering at Sony in general

- Sony Imageworks tend to use a lot of Commercially available software(Mental Ray and Renderman) but have moved more and more of their rendering to Arnold(by Marcos Fajardo) using their OSS implementations found here: http://opensource.imageworks.com/
- Sony Computer Entertainment work mostly in the real-time space with the various Playstation consoles they have numerous engines for various game types. But they also use both internally developed and commercially available software to generate cut scenes
- Sony Online Entertainment tend to do work quite similar to SCE
- Outside of this there is quite a lot of research going on within Corp R&D within the computer graphics area

Mo money, mo problems

- Every frame in a modern VFX driven movie uses on average:
  - <100,000 shader instances and between 30,000-70,000 OSO instructions
  - Shader compilation takes around 10 minutes/frame out of < 60 min render
  - On average 1TB of image data is pulled in through shaders per frame
  - Peak geometry load around 20 GB/frame
  - Usually rendered at 4K or 4K stereo
  - Output is usually in the range of 100s of layers/buffers sent to compositing, there are still a few companies that try to do most of the work “in camera”(notably Pixar, PDI) but its becoming extremely rare given the power of a modern compositor like Nuke
- In other words data management is today the most important aspect in production rendering
- Data point coming from the creation of the Spiderman trailer for the movie coming this summer

Temporal/spatial Sampling activity

- Create a temporal / spatial sampler in your renderers
- Adding a weighted average filter allow you to super sample to get rid of flickering

$$I'(x,y) = \frac{\Sigma\,\Sigma\, I(i, j)\, h(x-i, y-j)}{\Sigma\,\Sigma\, h(x-i, y-j)}$$

- By adding multi stage averaging you can get even better results, doing this will yield less sample usage by checking the previous average with the new sample until you get within a tolerance value, can also add sample stepping here to skip pixels if they are within the same tolerance as the surrounding pixels
- Adding temporal sampling over time slices(1 f / 8–64 steps) to further improve the output results

Temporal/spatial Sampling activity(Cont)

- Use any sample approach you can come up with(fixed iterative is a good approach otherwise more random methods such as Possion Disk and Stochastic Sampling might be interesting)
- With the sampler you can experiment with DOF, Motion Blur, AA, Shutter handling, you should have all the tools available with the basic ray tracer and a spartial/temporal sampler

SONY

make.believe