Deep Image Compositing @ Nordic TDForum 2011

Presented by: Colin Doncaster

Introduction

- this course is meant to introduce the concepts of deep image compositing
- provide some background that will help when you start to use deep images in production
- and share some insights into how it evolved and was adopted with-in a few large scale productions

The Catalyst

- dealing with depth passes in “post” was always hit and miss
- any sort of convolution that relied on this Z information failed more often than not due to incorrect Z values or discontinuous results
- integrating multiple passes via depth compositing was less than desirable
- this was becoming a bigger issue as there were more and more fully CG shots with data generated from multiple sources

Evolution

- there was already a multi-pass re-rendering engine but that added more complexity than desired so we started to see what else we could utilize
- deep shadows seemed ideal - although they solved slightly different problems in “light space” they provided us with the information we wanted
- it was straight forward to quickly test our hunch, it was a matter of adding a new display line to the rib file to generate a deep shadow from the camera
- we didn’t have to invent a new format (yet) to make use of them as Pixar provided the dtex libraries with all of their distributions

A Guinea Pig

- first tests were visualizing deep images in Shake which quickly led to extremely nice hold out mattes
- this was a single reader node and a hold out node
- DESS (2008) was in production and had a massive volume cloud with elements that needed to be integrated, iterations were slow due to volume rendering (and re-rendering) requirements
- Arieto gets wind of the deep image testing and volunteers to be a guinea pig

DESS Truck Swarm

The Fallout

- files were HUGE, although it was a lesser of two evils it wasn't ideal - this led to slow renders in the comp, way to much network traffic and at that time the deep image library wasn't thread safe
- PRMan undocumented z threshold RiOption to the rescue, this compressed the deep images by throwing away samples but this was a lossy process which meant a lot of experimenting and at times it introduced more artifacts (noise!)
- the files were still HUGE

The Success

- although there were issues we felt we were on to something which led to an effort to properly integrate deep images into Shake
- Peter Hillman @ Weta became the R&D lead and integrated deep compositing into the core of Shake
- easy to implement on a larger scale (ie. one line added into the RIB)
- at that stage deep shadows were “promoted” to deep images

A Much Larger Guinea Pig

- Avatar was in production and the director wanted an atmospheric, "under water feeling" for Pandora's jungle
- initial tests of depth fog with deep images proved positive, traditional issues of motion blur causing artifacts weren't visible and iterations could happen at the compositing stage fairly interactively

Improving the Workflow

- re-rendering issues with characters and jungle, Weta likes to get everything right in lighting and render as few passes as possible.
- Characters animation and look were changing at a different pace than environment
- the solution was to break out characters and jungle and re-assemble in comp via deep image holdout

A nice variation on mattes...

- how is this different from matte objects? Matte objects still require all of the elements, ie. you'd still have to render a matte jungle with characters and vice versa.
- this led to a nice side effect, memory consumption was a lot less as large elements could be rendered separately (although storing the deep samples meant more space needed for the frame buffer)

Deep compositing rolled out...

- the depth fog and holdouts became the main use for deep images
- it was adopted for many Avatar shots with deep images being "switched on" more often than not (due to an approved look)
- at this stage we were still only storing deep images with an alpha

What else was possible?

- due to having projection matrices we started to experiment by extending the volume integrator to take non camera based deep images for light integration.
- we were also able to inverse project the samples to create point clouds for visualization purposes, surface derivation and more...
- another nice side effect was very interesting “making of” elements

What is a deep image?

Pre-History

- a frame buffer is a flat data structure that stores any number of values at pixel locations, this can be an image, open gl texture or screen
- when a renderer rasterizes geometry it internally stores a lot more information than just the final RGBA values
- the idea of making this data available post-render isn't new - IPR rendering, RLA, Lpics and even AOVs for relighting in a compositing application are all examples of this...

Pre-History Cont...

- depth information is extremely important when rasterizing
- at some point a smart person decided to store the this depth information
- followed by another smart person storing the projection matrices to access the data for shadowing purposes
- although each pixel contained a depth value there are many occasions when this isn't enough and these traditional depth passes were (and still are) laden with issues

Problems with traditional Z

- there are many ways of generating a Z pass
- z, 1/z, P
- unfiltered or filtered (!?)
- no transparency
- what depth to use? min, max or average?
- biasing
- all of this generally lead to a lot of artifacts and a lot of value tweaking to get it right

A Solution!

- Deep shadow maps were introduced via Tom Lokovic and Eric Veach @ Pixar circa. 2000
- these were a major breakthrough in lighting at the time
- multiple depth samples are stored via an opacity extinction function (how much light was penetrating to that sample) and provided a means of storing partial pixel coverage into the image
- this improved hair self shadowing, introduced motion blurred shadow maps and increased shadow quality at much lower resolutions

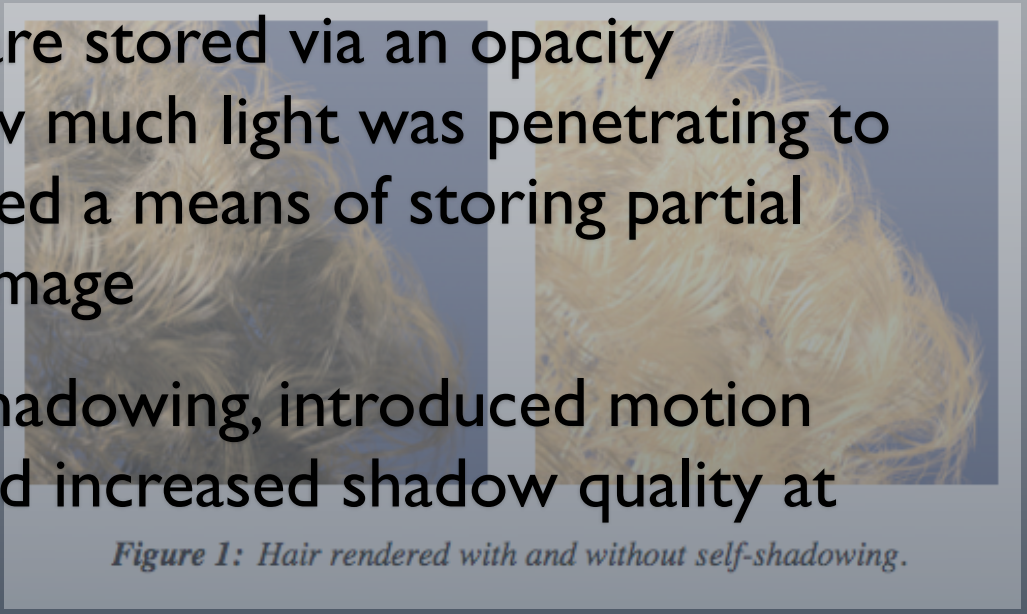


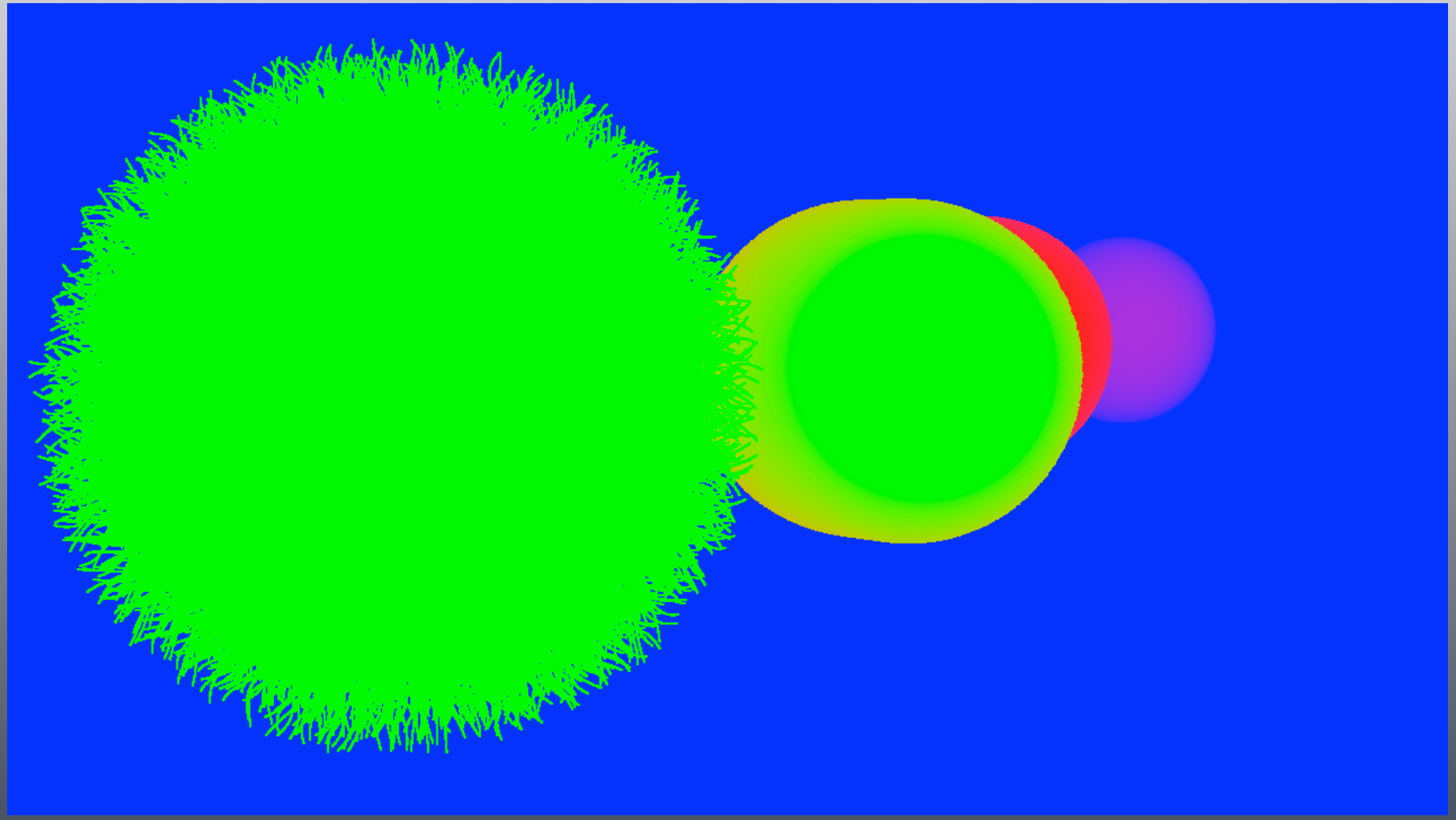
Figure 1: Hair rendered with and without self-shadowing.

disclaimer: I'm simplifying a lot of this, of course

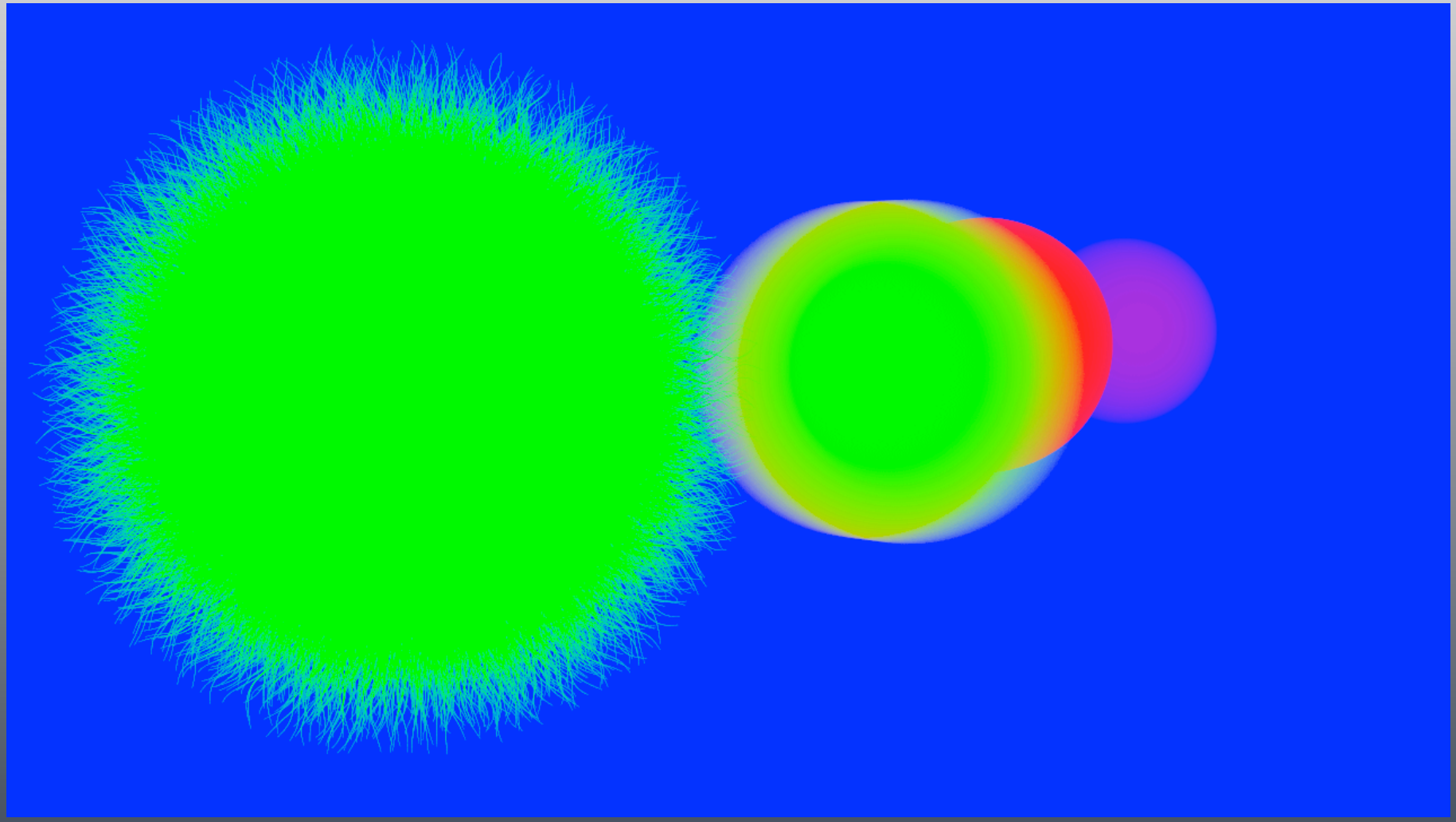
Deep Images

- Just an evolution of deep shadows
- Deep shadows were created to solve issues inherit with traditional Z depth images and shadow casting, deep images extend that to the camera view
- Each pixel is a list of samples with depth information vs. a single depth entry
- each depth sample can store Alpha, RGBA or other arbitrary values (PRMan 16 and OpenEXR 2.0)
- the elegance comes from how these deep images are interpreted

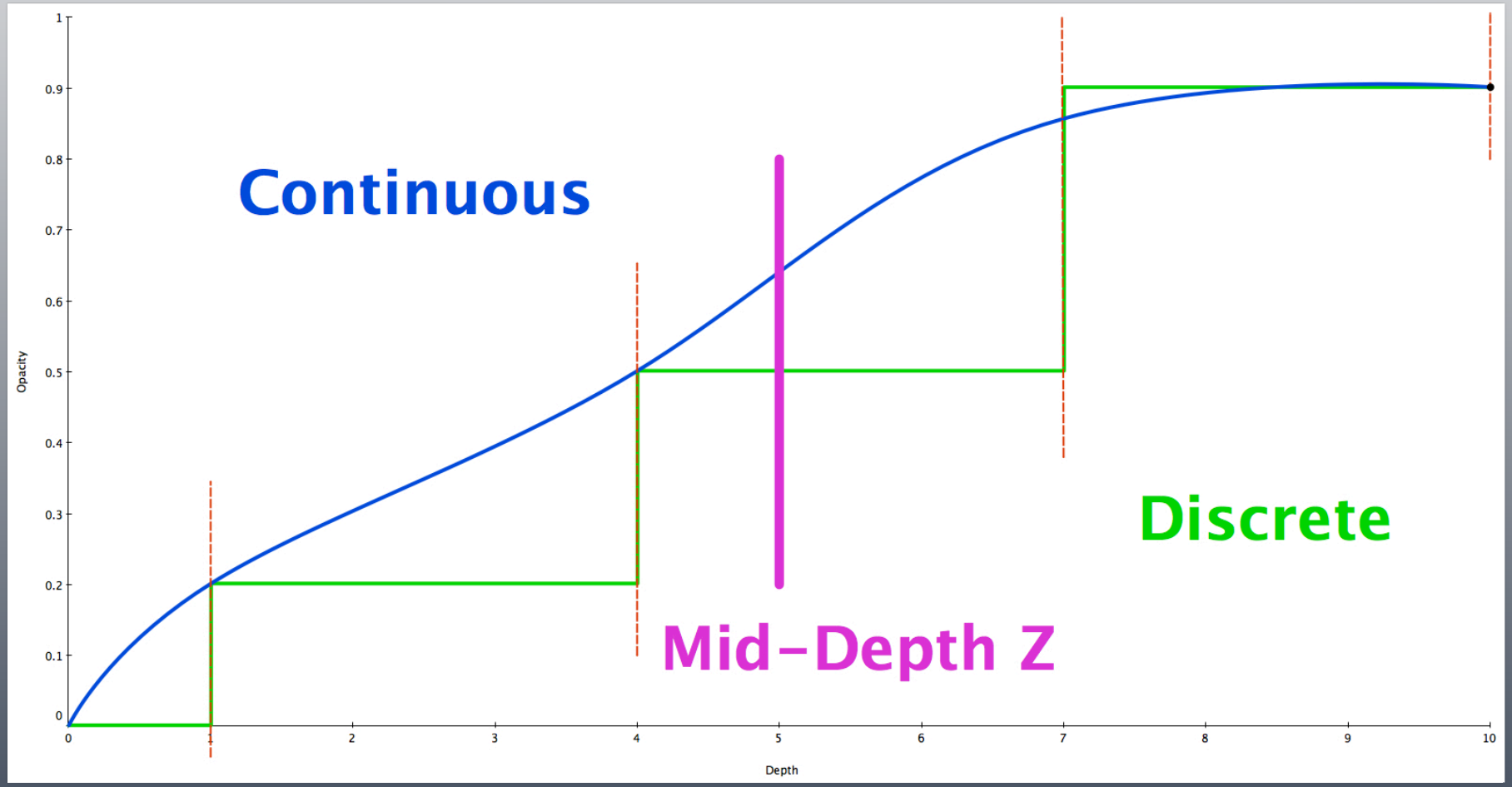
Traditional vs.



Deep

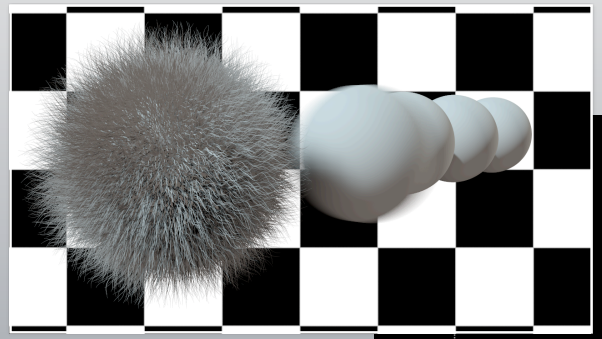


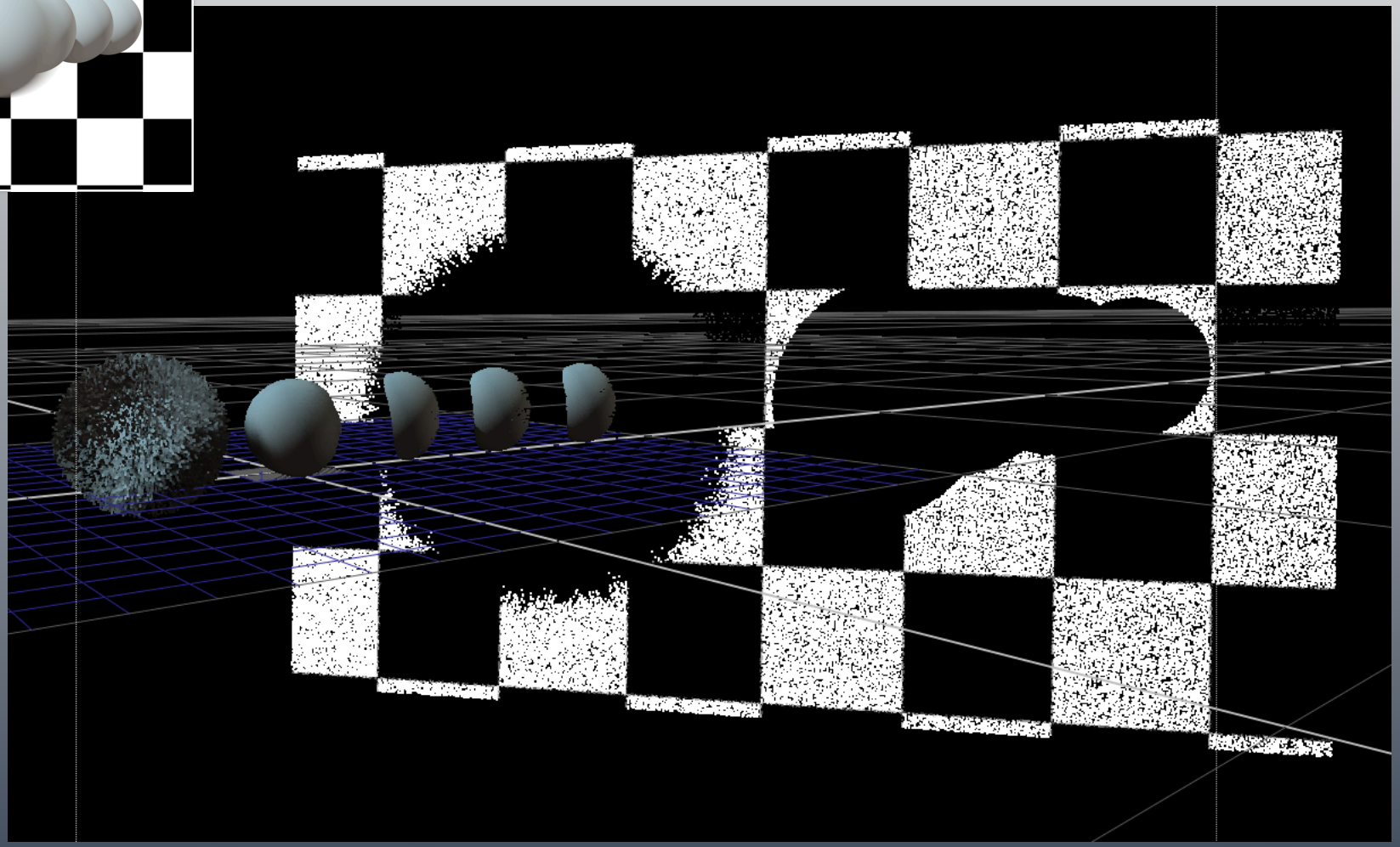
What do deep samples look like?
(Continuous, Discrete and Z)



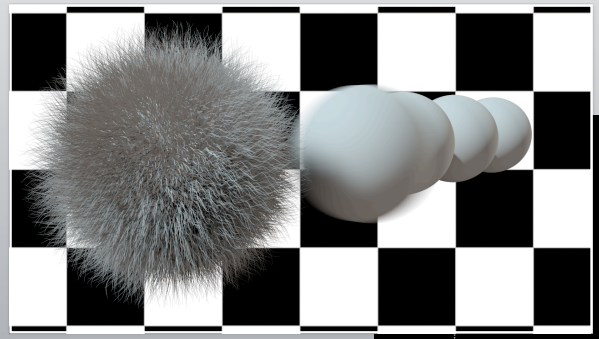
This should really be flipped...

What do deep samples look like?
(Inverse Projection)

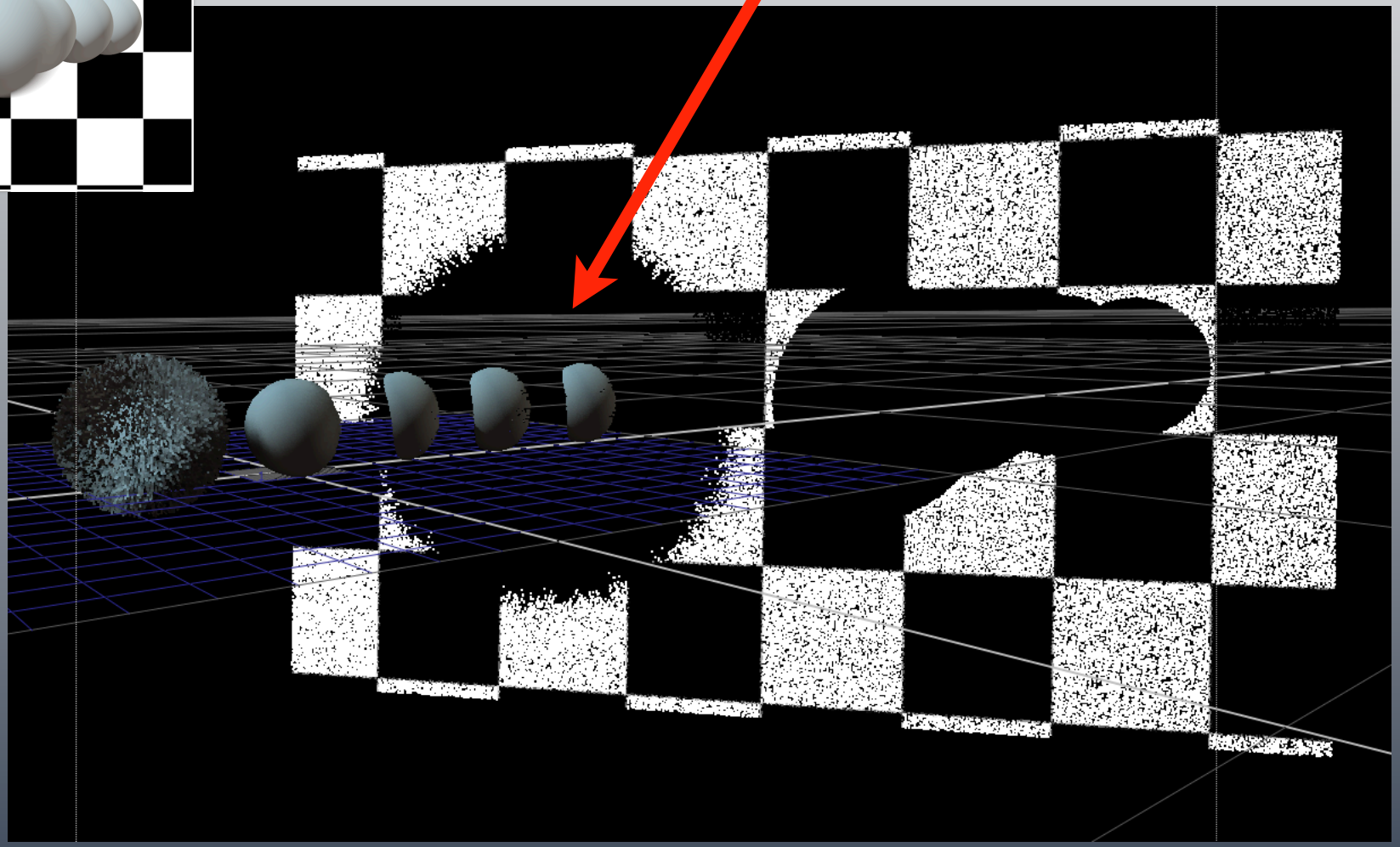




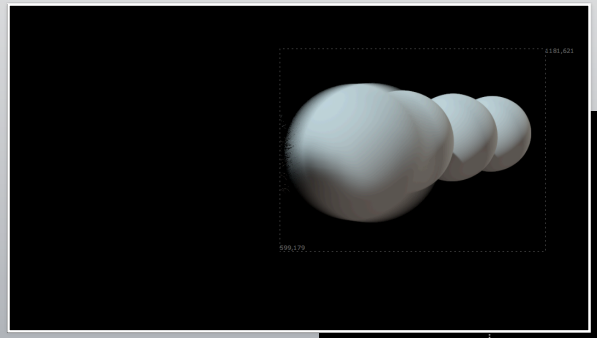
Some obvious misconceptions...

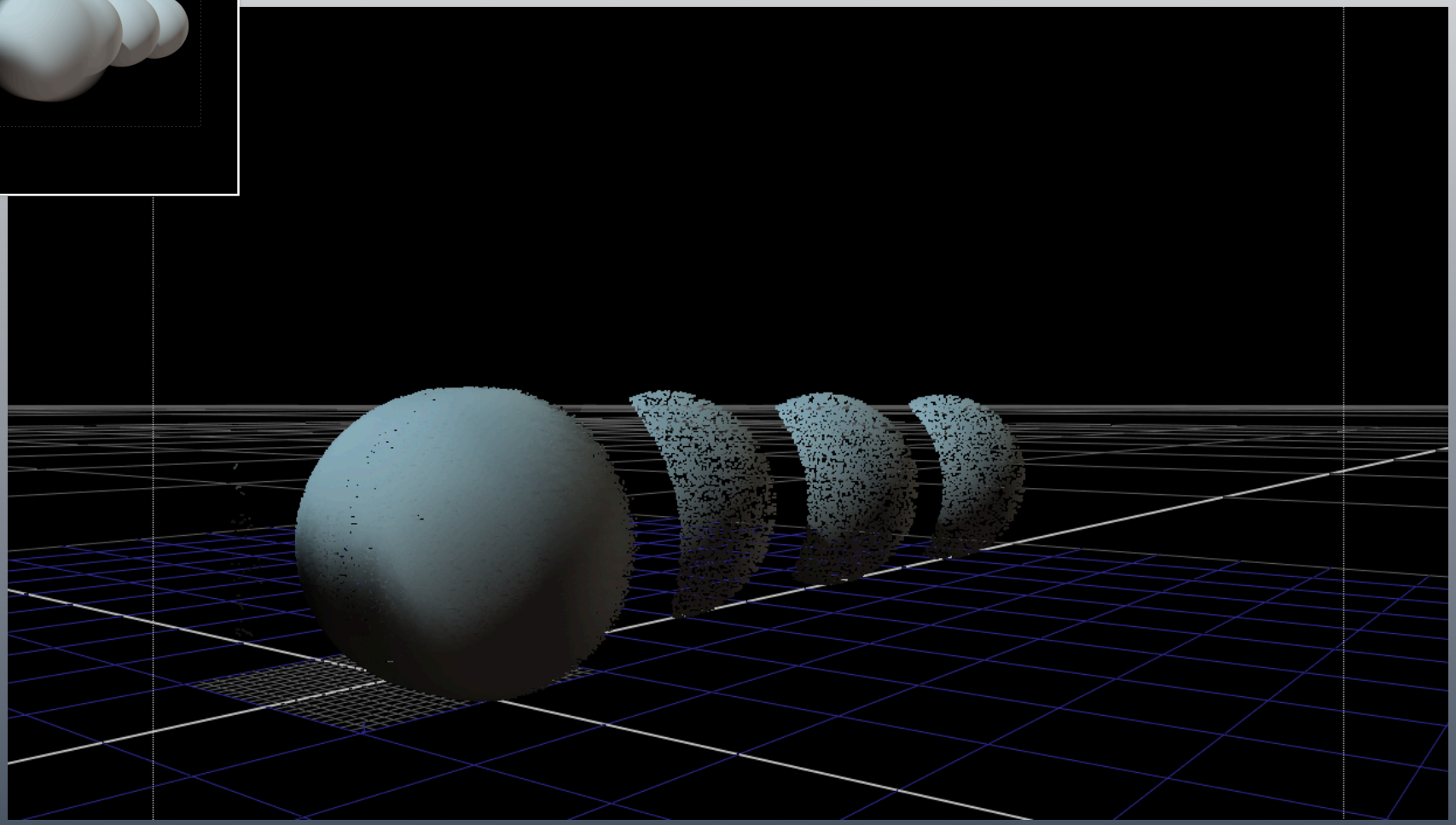


there's data missing!



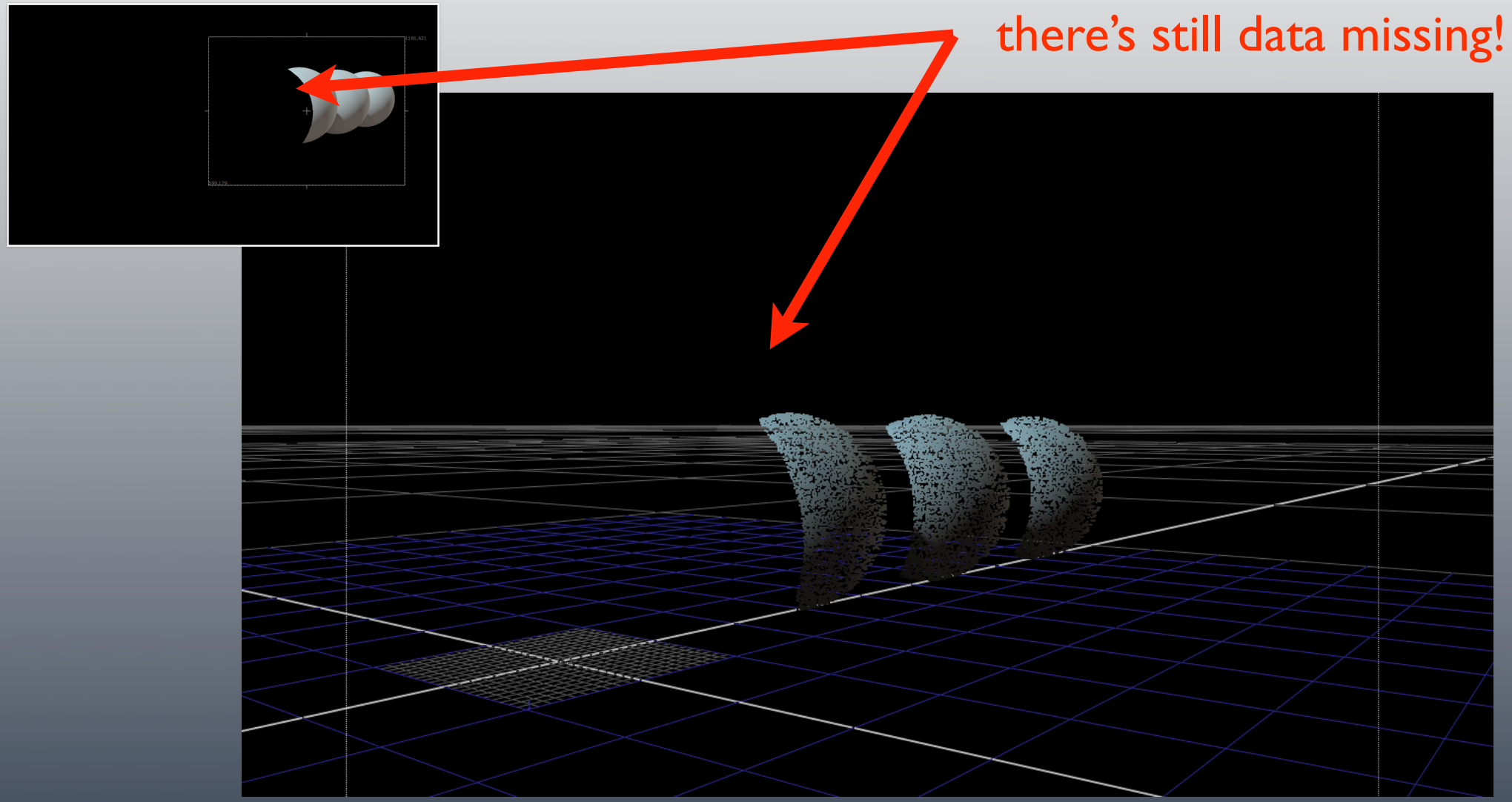
Modifying Deep Data - Z Cropping





Modifying Deep Data - Z Cropping Pt 2.

there's still data missing!



Compositing Operations

- merge (append and interleave)
- holdout
- z-crop
- resample
- convolutions & generators (noise, defocus, etc.)

Be aware of the data...

- the data is even more important than before, get in the habit of inspecting what the renderer has generated (and make sure you have tools that makes this easy)
- building tools to visualize the data is going to be just as important as what you do with it
- as always meta data stored in the header can be very useful!

How to get started? Generating...

- you need a renderer that offers a means of generating deep image data (PRMan, 3Delight, Mantra...)
- Mantra generates deep camera files, PRMan and 3delight generate the deep image files which will be secondary secondary displays of the main camera
- RMS supports this for PRMan, 3delight for Maya will support it but you need to make sure you turn mip mapping off (so you can generate the non power of two images)
- be aware that these are all DIFFERENT formats

Using...

- you need a compositing tool that reads them, right now this means The Foundry's Nuke or a proprietary system if available
- You'll need to have a reader for the deep image format of choice, Nuke ships with PRMan dtex support (source code supplied for this as a reference)
- until OpenEXR 2.0 you need to keep track of all the different files, compounded with Stereo
- don't using for everything, only use it when needed

When should it be used?

- not all the time!
- using and adopting deep image compositing and the impact it has on your pipeline/io depends on your workflow
    - get it right in lighting (element based passes) = good
    - multi-pass rendering and lighting in comp = bad
- integrating volume elements (+ fur, feathers)
    - with other CG passes that have deep data
    - integrating with live action plates

Notes on implementation...

- whether or not you're integrating into a third party application there's still a case for implementing your own tools
- depending on how they were generated samples may not be sorted, so be smart about when and how often you do so
- the sample list can be big, heap sorting is a good balance between speed and flexibility
- convolutions are HARD and slow, resampling is your friend

Some things to consider...

- it's a workflow that's still in it's infancy, things will change!
- Weta's OpenDeepZ -> OpenEXR 2.0 will mean a decent standard is on its way
- hopefully this standard format will make it easier for other applications to generate and use deep data, having spoken to a few vendors most of them are waiting for OpenEXR 2.0
- it's pretty obvious when you think about it!

Future considerations...

- Is this the silver bullet? I don't think so, this is an evolution to a more flexible way of compositing. It solves ***some*** problems but not very elegant
- It breaks a lot of research, GPU acceleration generally is thrown out the door. With that said, there have been some deep buffer experiments on the GPU
- it will be important to leverage standards in the future and work with hardware vendors to provide deep frame buffers
- why stop here? why not just use brick maps or point clouds?

Any questions?

Colin Doncaster - colin@peregrinelabs.com