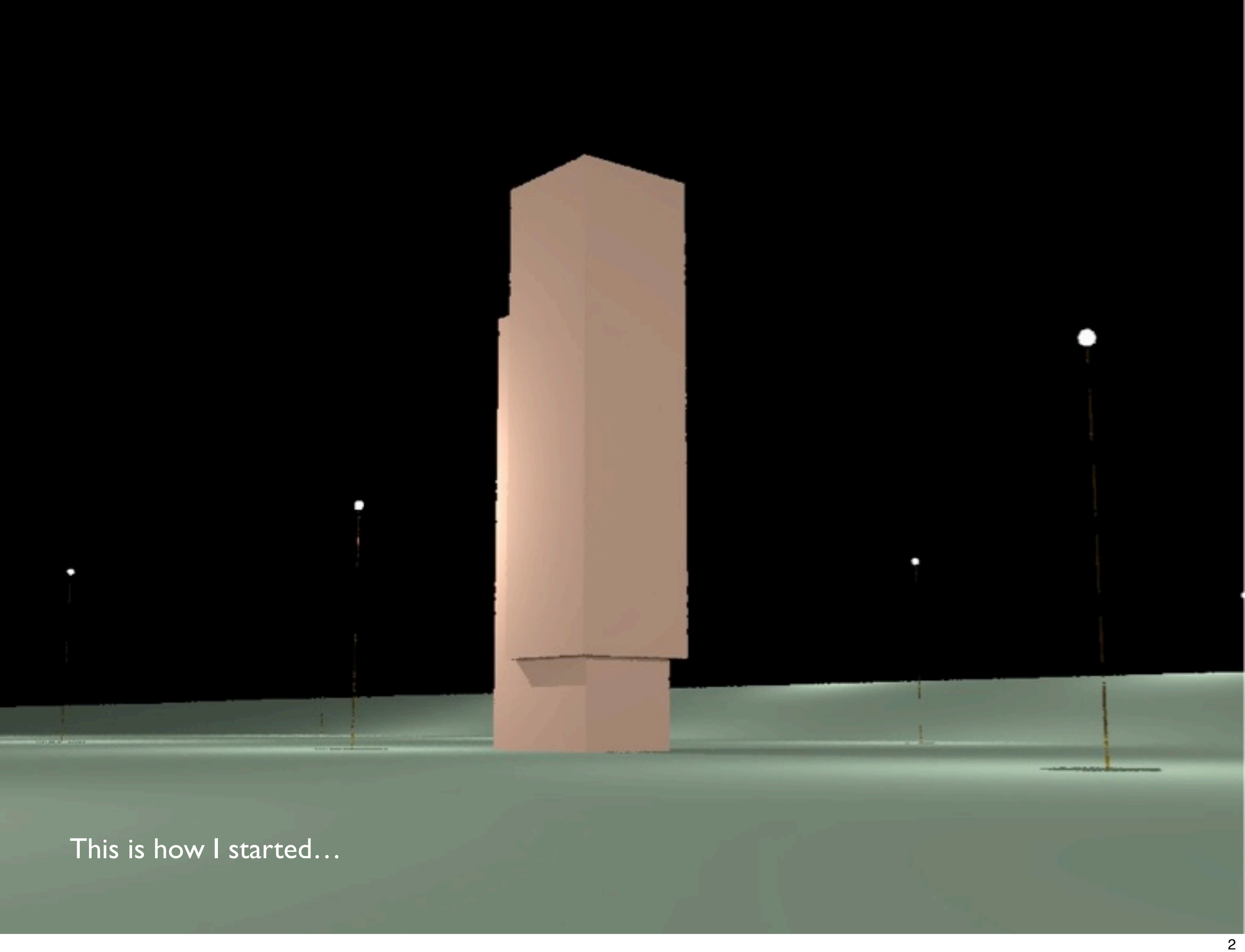
A Decade of Radiance Tricks

Mark J. Stock
mstock@umich.edu
http://markjstock.com/

Harvard Graduate School of Design
Apr 7, 2011

As new users to Radiance, and possibly to rendering programs themselves,
You're going to go through a learning curve.
Radiance seldom makes that easy.

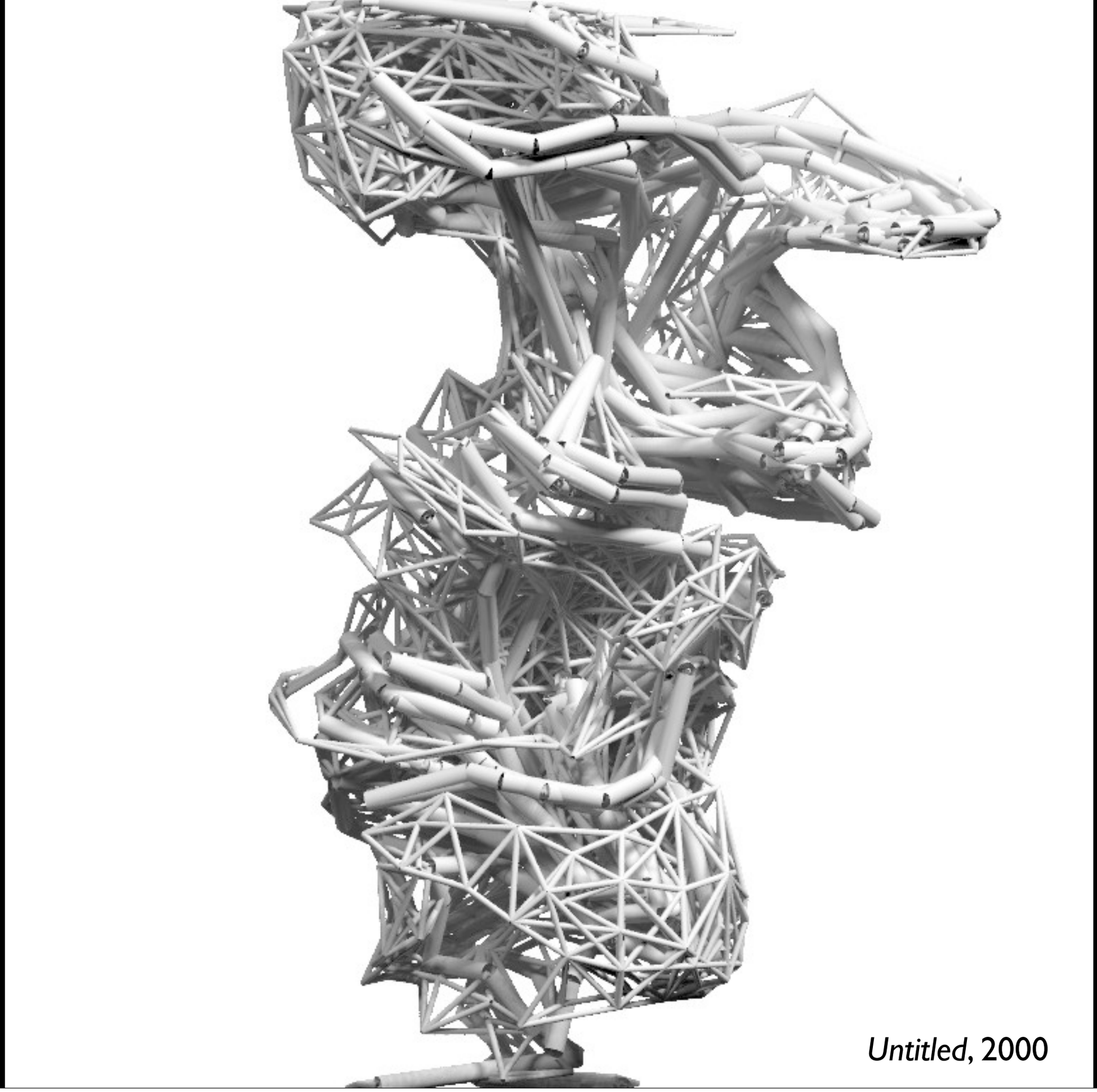
At the very least, I'd like this talk to solve some of your problems and open some doors into Radiance's deep capabilities.



From such humble beginnings!

I’m not sure if your first rendering looked as bad as this. I hope I’ll present enough hints and tricks for you to completely skip this step.

But in a few short years of pursuing rendering as a hobby, I was able to turn this...



Untitled, 2000

...into this. And in the decade since then, I've learned a little more.

Let me pause here for a moment to tell you why I use Radiance.
[interested in 3D rendering, other computer simulation,
studied computational fluid dynamics in school,
first job out of Master's program and I was stuck...

This image propelled me into making science-based artwork,
which has brought me some success.

Outline

- Your model: colors, large models
- Environment: land, sky, clouds
- Rendering: parameters, effects
- Post-processing

This is a rough outline of what I hope to cover today…

...It loosely corresponds to my workflow when using Radiance.

Throughout the talk I'll repeat various points,
and at the end I'll pull them together with some use cases, then an animation.

Your model

You probably have this part under control...you’re architects-in-training, models are your life, right?

I don’t know anything about CAD and GUI modeling, but I have played with triangle meshes, so I’ll start there...

Model geometry

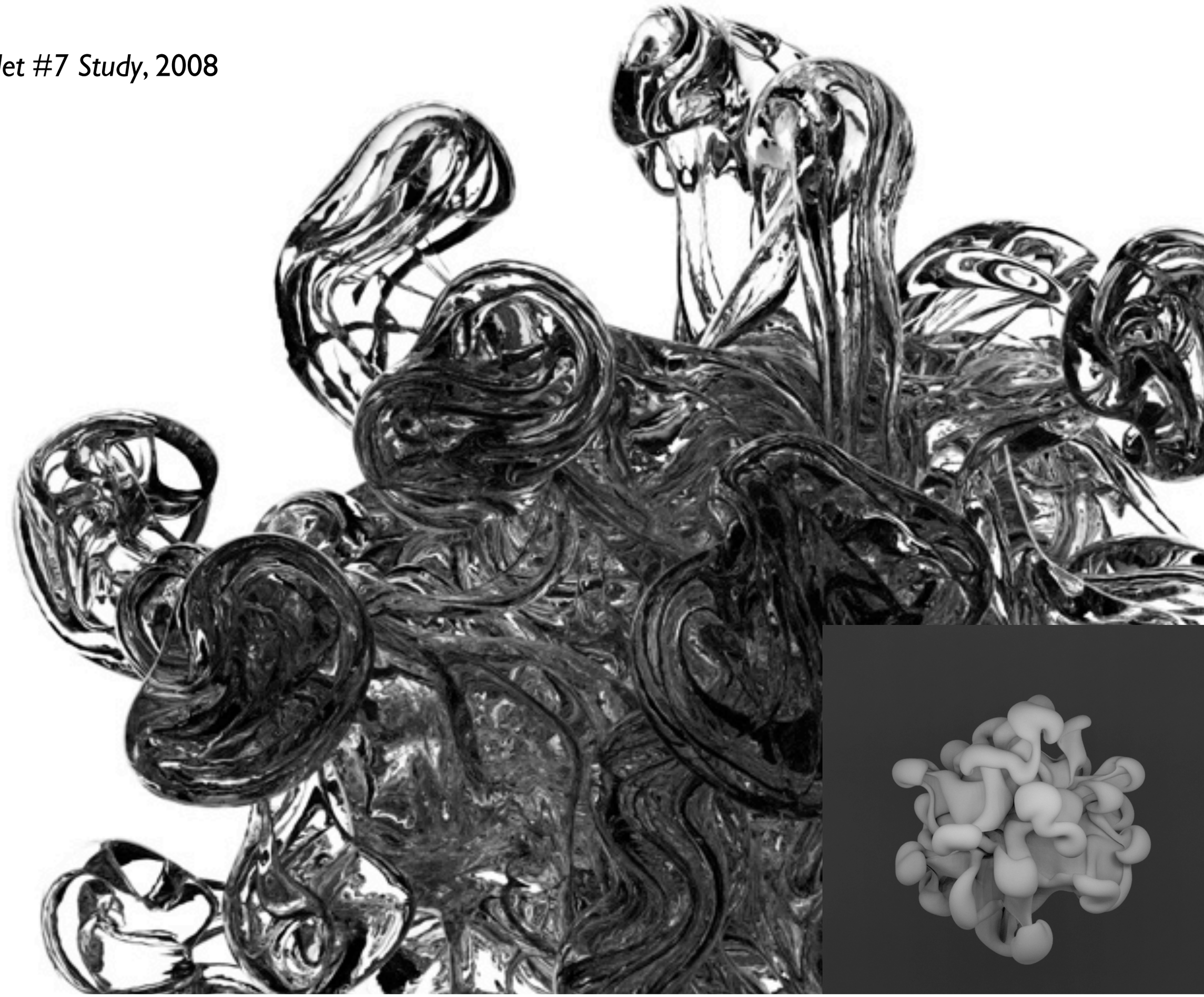
- I like modeling in triangles
- With normals
- Use *obj2mesh* to make Radiance mesh
- Instance the mesh “mod mesh id”

I hope triangles are still the defacto modeling primitive.

Computational fluids people are still mainly in love with quads, but they are also masochists for meshing.

Because my geometry is usally fluid in nature, I spend some time smoothing and refining my triangle meshes, and *need* to maintain node normal data. Radiance uses this to good effect…

Droplet #7 Study, 2008



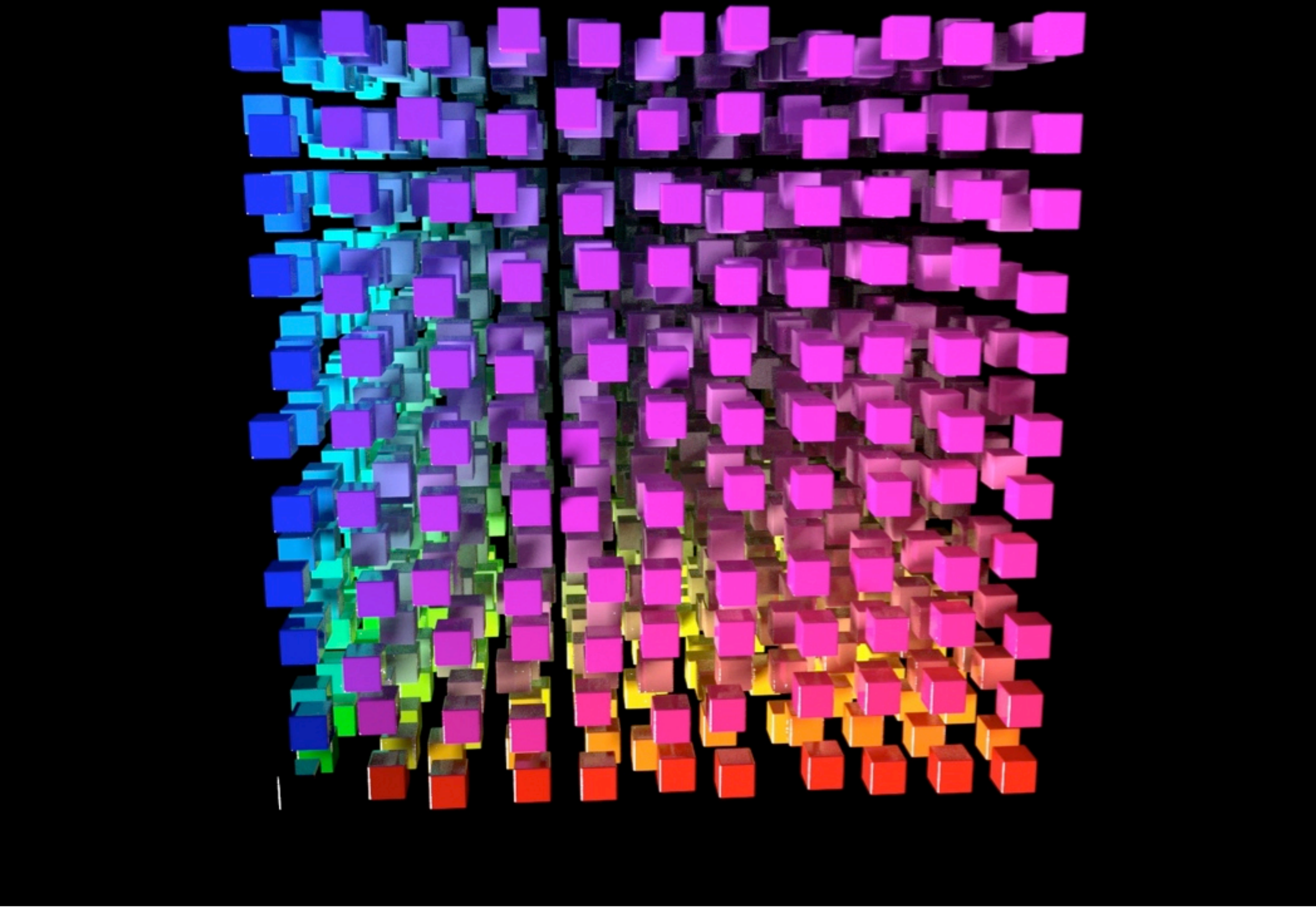
Large triangle mesh geometry
To use the trans material to get reflections and refractions
I needed to have a very smooth surface

Speaking of surface properties...

Choosing colors

How do you choose colors?

What am I thinking? Architects always use white, right?

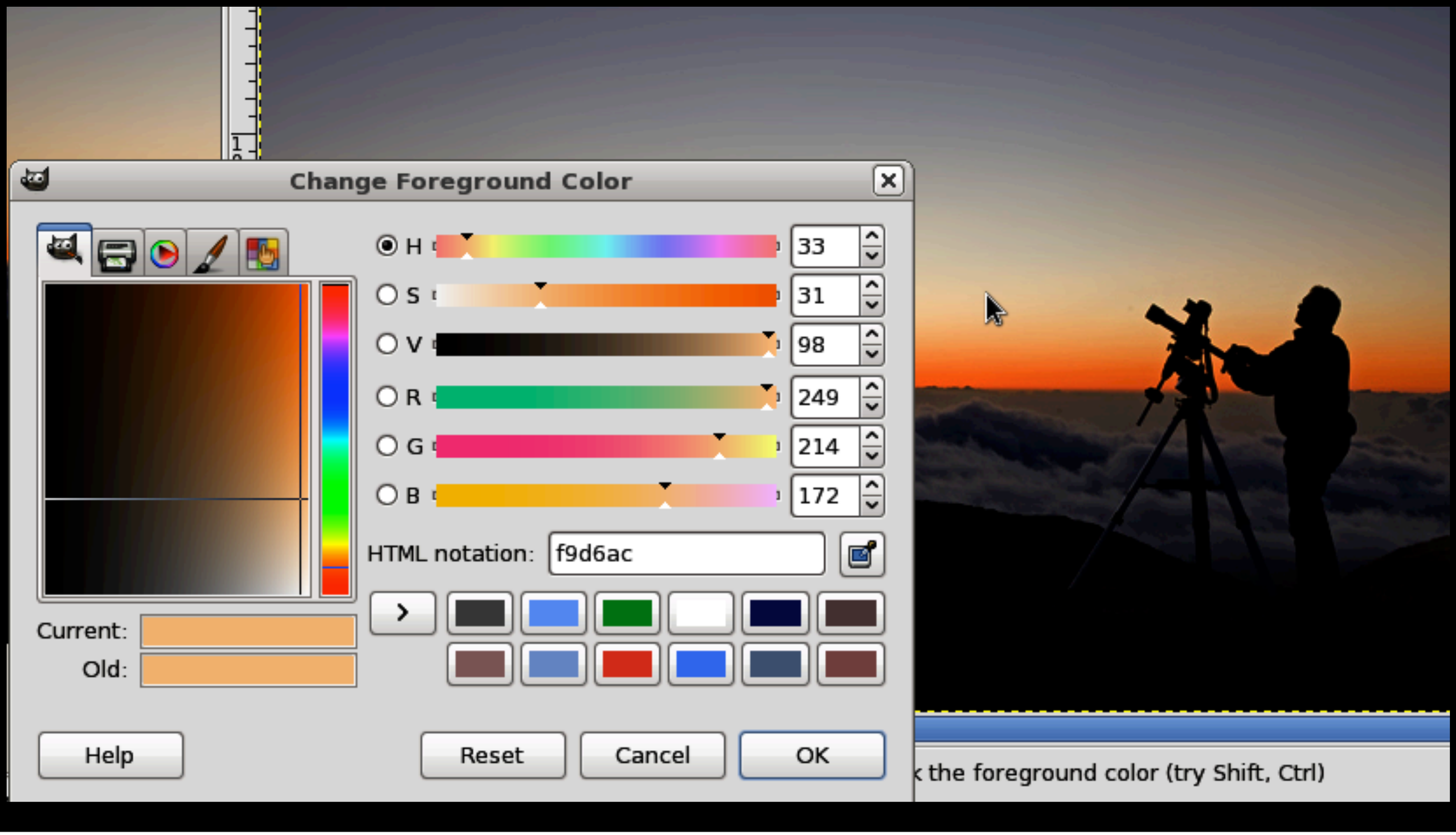


In case it hasn’t thrown you off already, Radiance uses reflectivity and not “raw” color. RGB from Photoshop just doesn’t work.

This is a linear color cube, 0 to 1, r,g,b are x,y,z. Most of the interesting stuff is crammed in the bottom left corner.

But color guides always work in the 0..1 or 0..255 range...

R,G,B estimate



So when I can’t find “albedo” values for my surface properties, I resort to this hack.

Start with your desired color. In this case, a color from the sky...

R,G,B estimate

- Remember Radiance uses reflectances so 18% grey card is 0.18 0.18 0.18
- Find "traditional" R,G,B (249,214,172)
- Normalize 255 to 0.9 (0.88,0.76,0.61)
- Square (0.77,0.57,0.37)
- When in doubt, plastic with no specular

Then follow this quick algorithm…

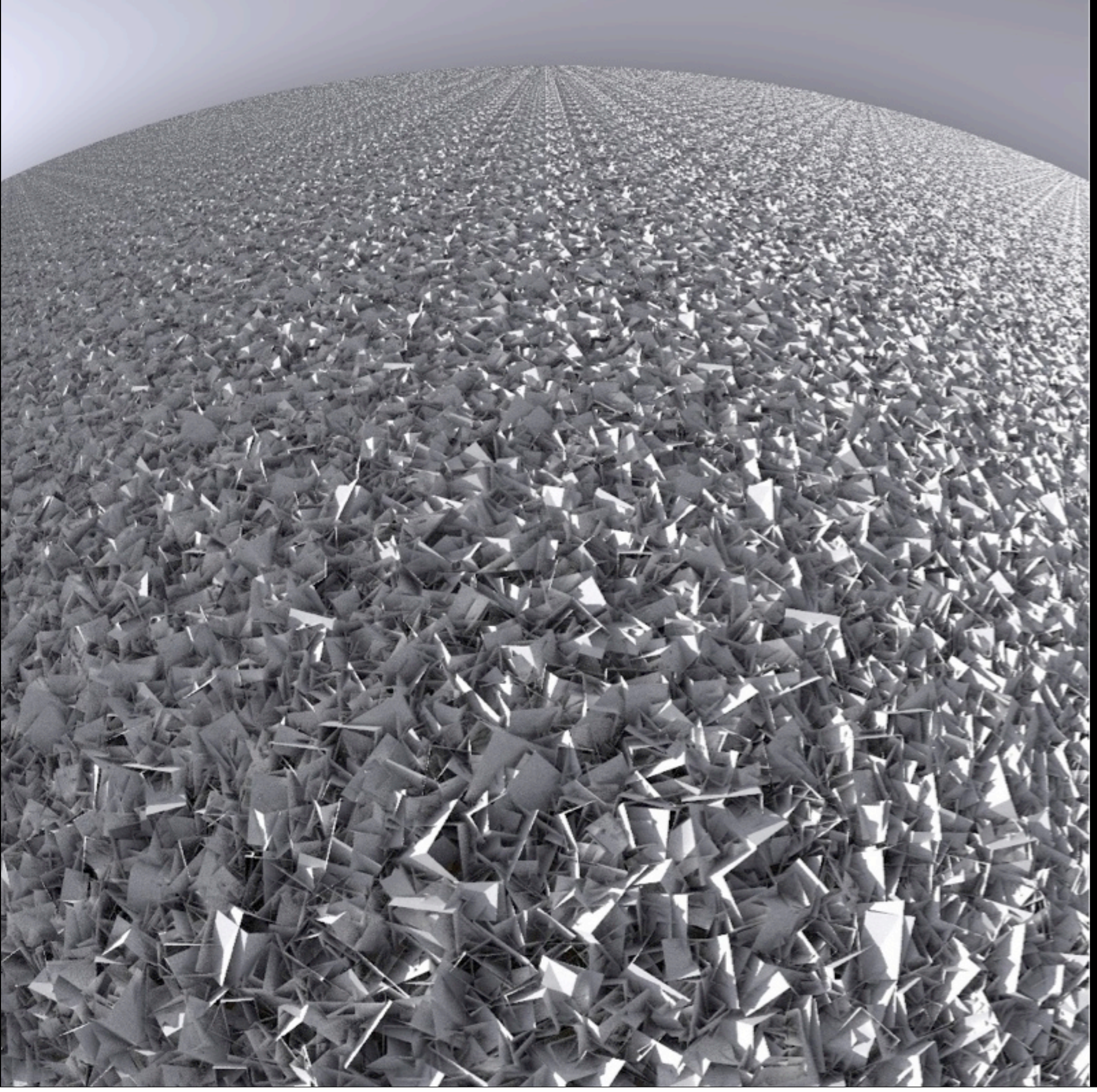
Many of my pieces have been monochrome because choosing specific colors is difficult, but also partially because I am color-blind.

Algorithmic geometry

Naturally, you create much of your geometry by hand, but both of us have a need to create algorithmic geometry.

Both the natural and the built environment are affected by agents, and their actions generate detail at many scales.

Don’t rely on .cal files or “texturing” for detail when you can define detail geometrically.



Randomly-oriented open boxes across mesh

void plastic def 0 0 5 0.8 0.8 0.8 0 0

3976 polygons per tile turned into frozen octree

xform to make 100x100 array of tiles

vh 100 -vv 100 -vta -aa 0 -ad 16 -ab 2 -as 0 -ps 1 -x 2048 -y 2048

This is a quick demonstration of the power of geometric detail.

Because Radiance can accurately compute the global illumination of a very complex scene (in under a lifetime), we should not be afraid to leverage it.

Again, many polygons and cylinders per tree, turned into frozen octree

Place one thousand instances on ground plane

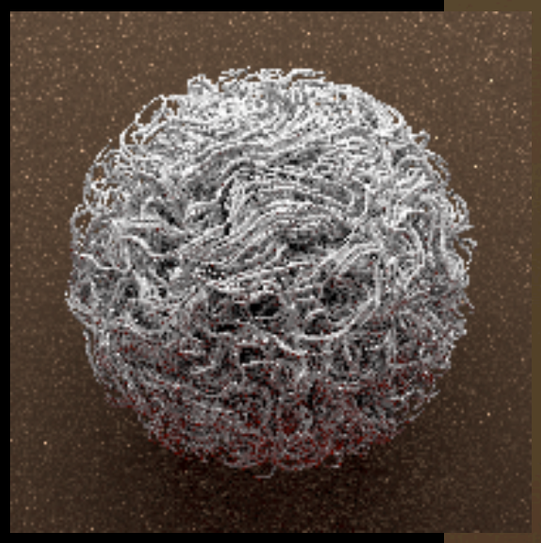
Here is just another example, distinctly natural objects this time.

Scene/lighting

Think like a photographer---Radiance is accurate enough to allow that.

Use large lights, key lights, reflectors, etc.

Dynamo, 2006



1.2M primitives (cylinder, sphere, cone)

tubes and room are plastic (.6 .6 .6 0 0)

Background is plastic (.075 .048 .028 0 0)

Incandescent area light source overhead

-vtl -ds 0.1 -ab 3 -aa 0 -ad 8 -as 0 -ps 1 -u+

36k pixels, 8 rpiece jobs, 38 CPU-days

I used uncorrected incandescent lighting to give this piece a warmer feel.

Also, I almost always use area lights indoors.

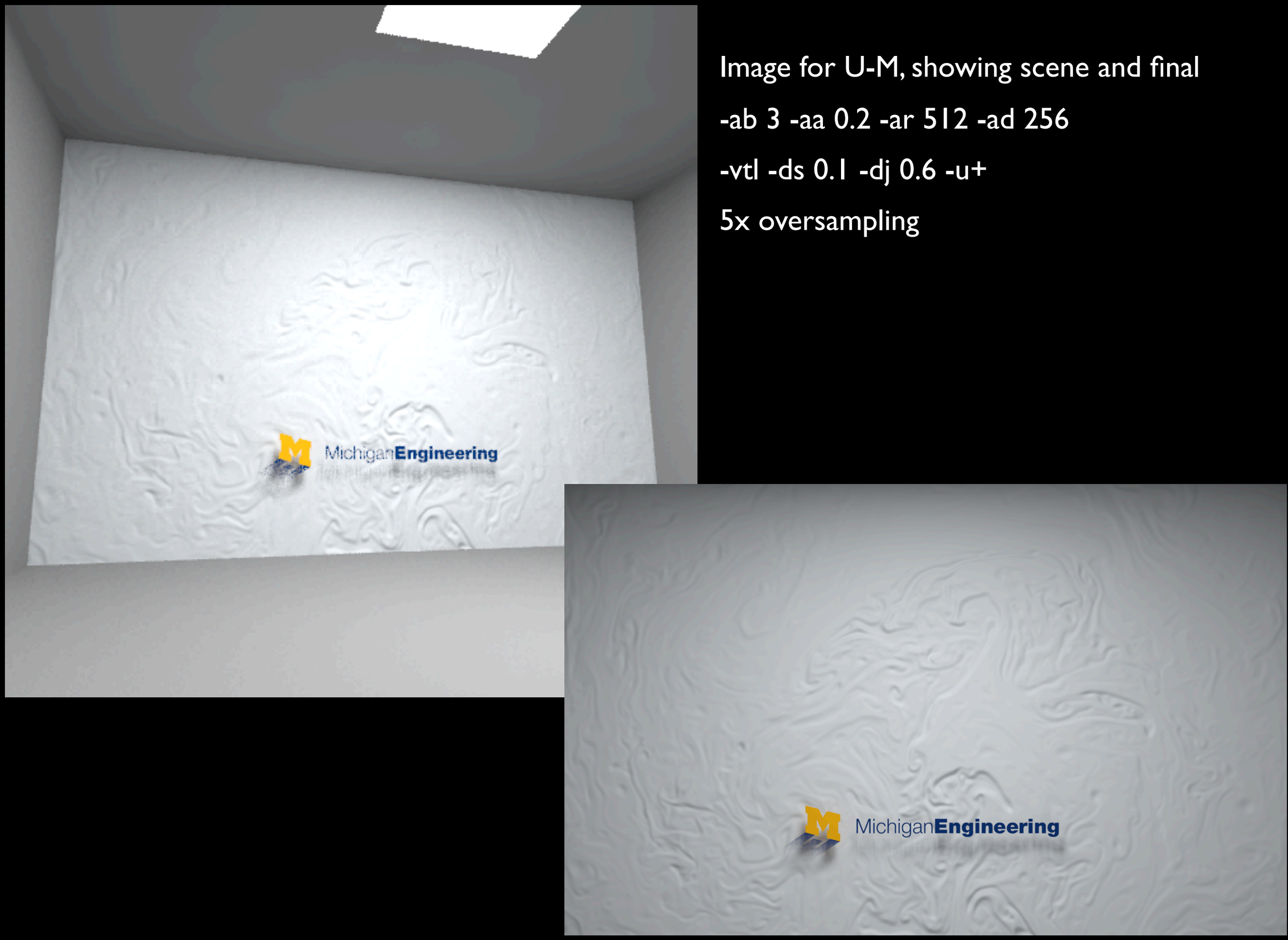


Image for U-M, showing scene and final

-ab 3 -aa 0.2 -ar 512 -ad 256

-vtl -ds 0.1 -dj 0.6 -u+

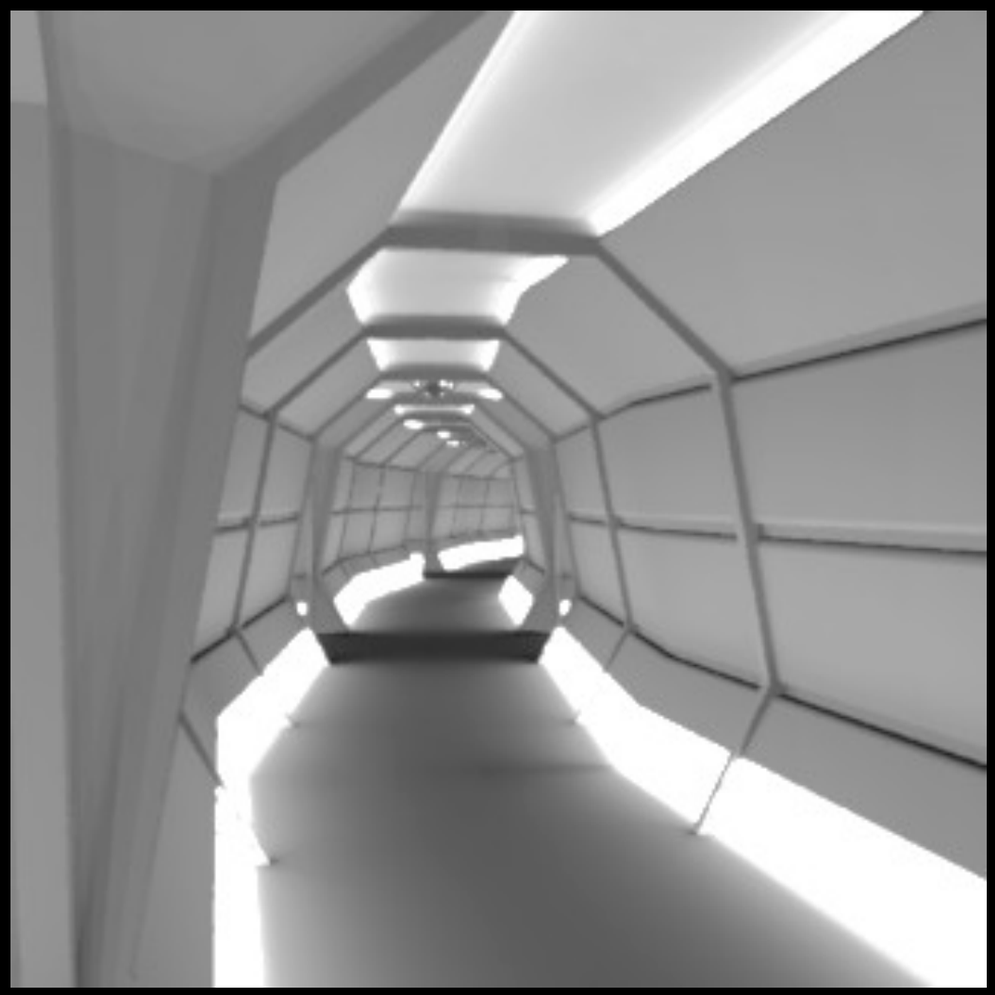
5x oversampling

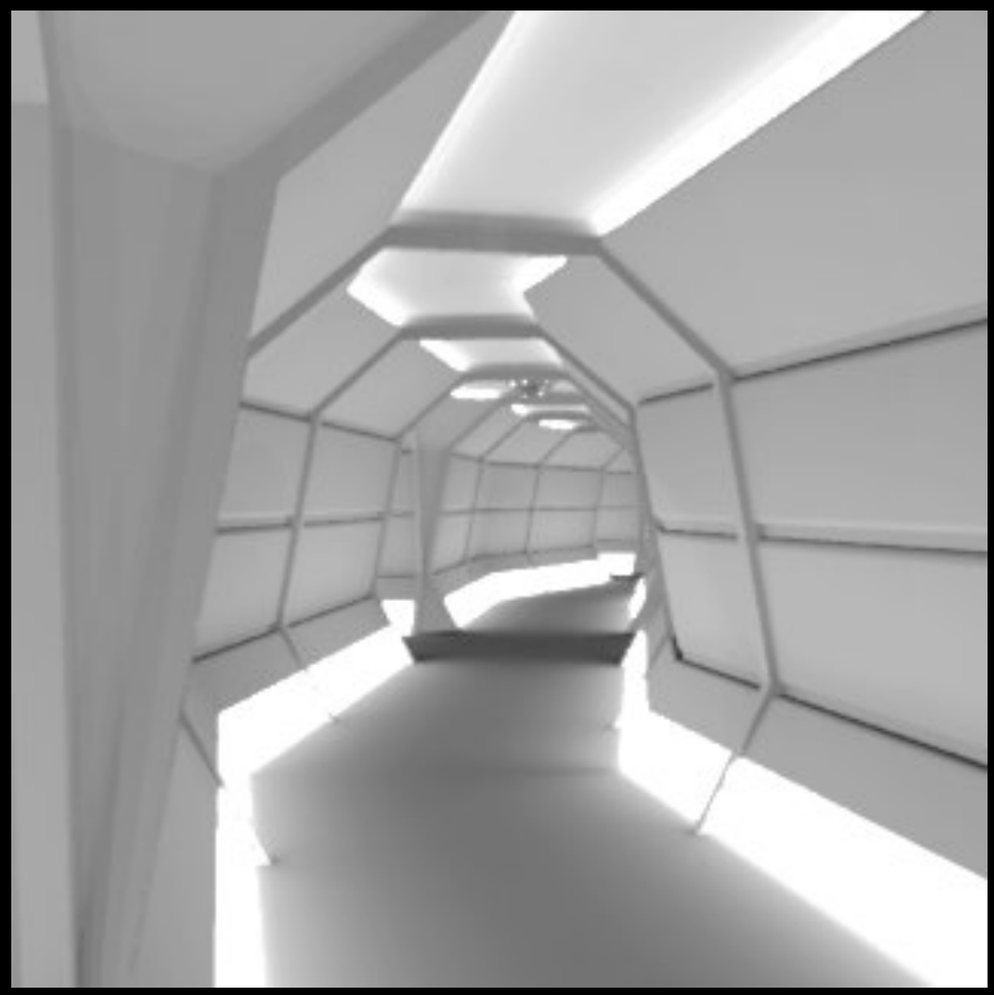
Outdoor scenes are easier to light, as indoor scenes actually require realistic walls, electric lights, etc.

Make note: to use area light sources, you must set -ds (direct sampling) to 0.2 or less!

Mirrors

Mirrors can save on geometry in many cases.

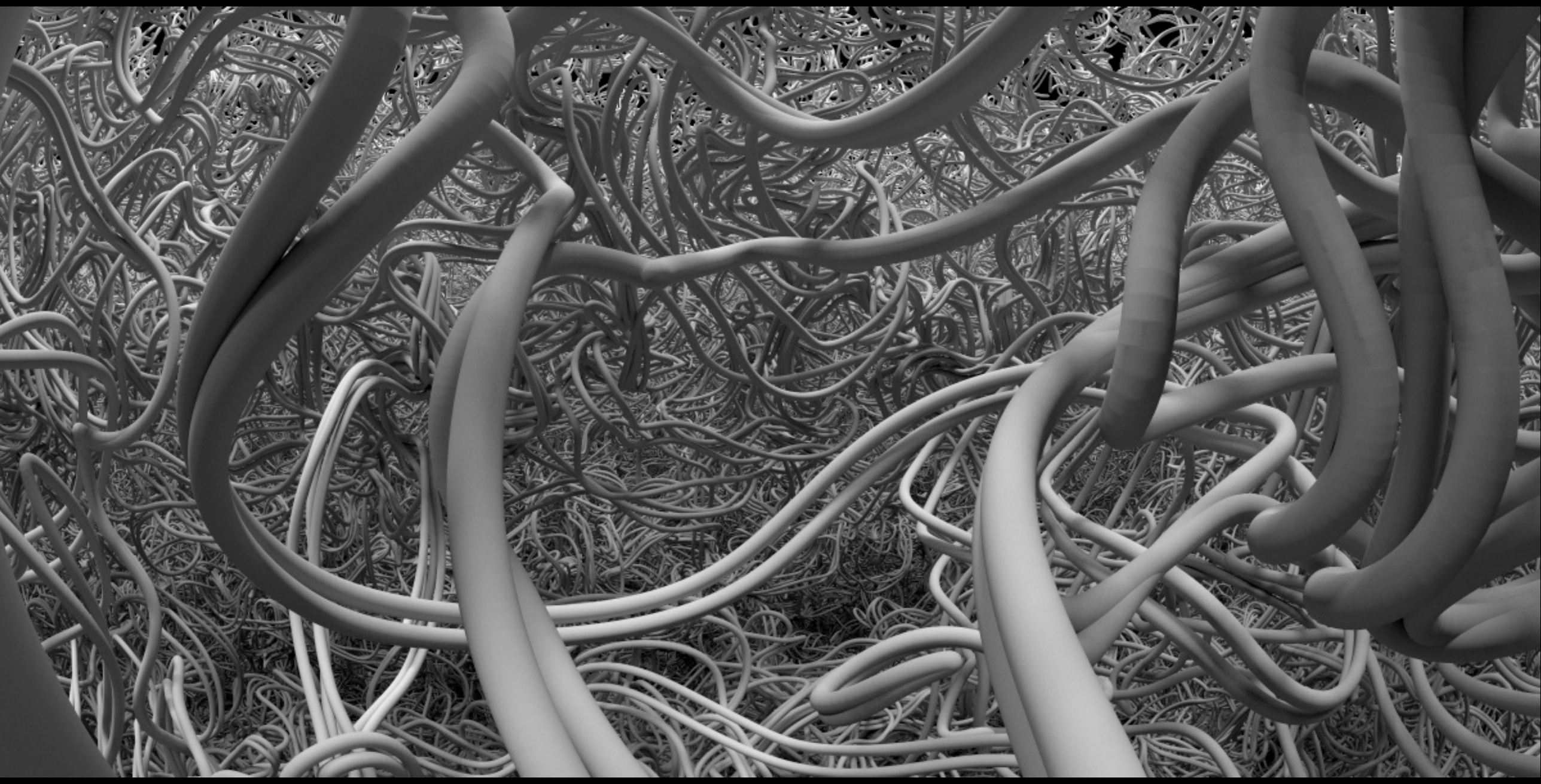




Simple hallway model turned into entire starship with 4 mirrors

Not just a hallway, obviously.

Turbulence Infinite, 2003



42k cylinders + 42k spheres in static octree, instanced 45 times, plastic (.4 .4 .4 0 .01)

One overhead light, plastic (.6 .6 .6 0 0) floor, mirror (1 1 1) walls

-vta -vh 176 -vv 88 -ab 1 -aa 0 -ad 16 -as 0 24k x 12k, 2 days

This is called “Turbulence Infinite” and

Was the first image that did not use ambient caching (-aa 0)

While this may look infinite, there are cleverly-placed mirrors to make 100k segments look like more

Wide field of view hides mirror lines

Very large models

What if you actually have very large models, that can’t just be copied, or mirrored?

oconv is a finicky beast...

oconv hates...

- Scenes with *many* overlapping primitives
- Scenes with >5M primitives, some overlaps
- Scenes with large variations between largest and smallest primitives
- Chains of overlapping cylinders and spheres

While oconv (or obj2mesh) can handle 10M non-overlapping polygons
It thrashes when there are even 100,000 seriously-overlapping polygons

Even with 8 or 16 GB, making some octrees fills up RAM and can take forever.

There is a solution...

One solution

- Partition geometry along coordinate axes as many times as is necessary
- oconv/obj2mesh each partition by itself
- Instance the resulting octrees together

Naturally, I have a script to do this...

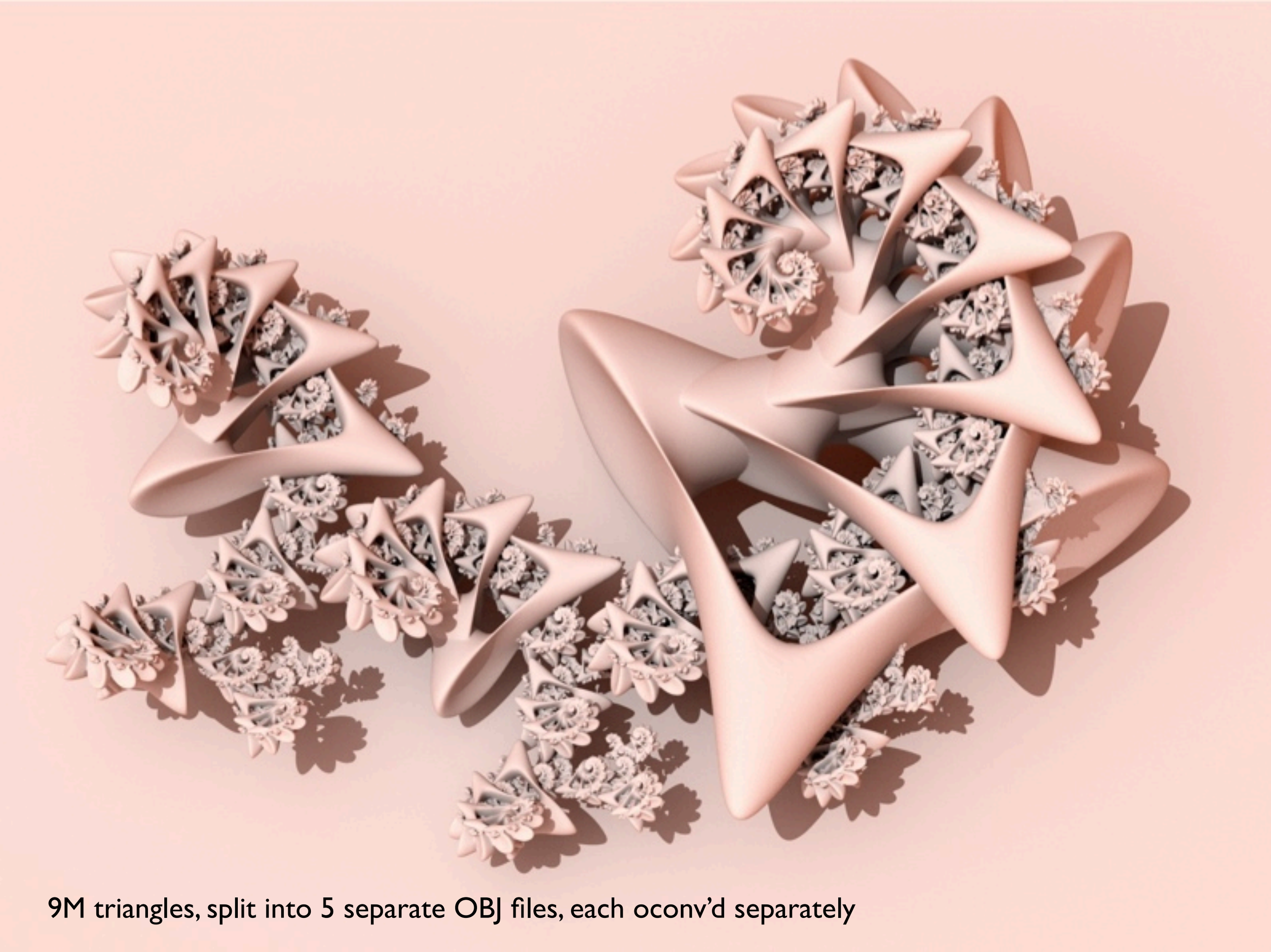
Perpetuity?, 2008; >20M triangles?, plastic 0.03 and 0.8

That extra geometry preprocessing step allows me to build much larger models.

In the corner is the final rendered image

The big image is the geometry with only 1/3rd of the chunks instanced

Again, you see algorithmic geometry, inspired by fluids, but imaged as a solid. I love that tension!



9M triangles, split into 5 separate OBJ files, each oconv'd separately

"Kris Northern" created this geometry for me,
but being composed of nested copies of the same triangle mesh,
it contained many overlapping primitives.
Oconv bombed on the whole geometry,
but the script broke it up into ~100 MB OBJ files, each of which converted fine.
The final oconv takes seconds.

Environment

Models don’t normally exist in a vacuum, though some of mine do.

Let’s look at some methods to define the sky, clouds, land, and participating atmosphere...

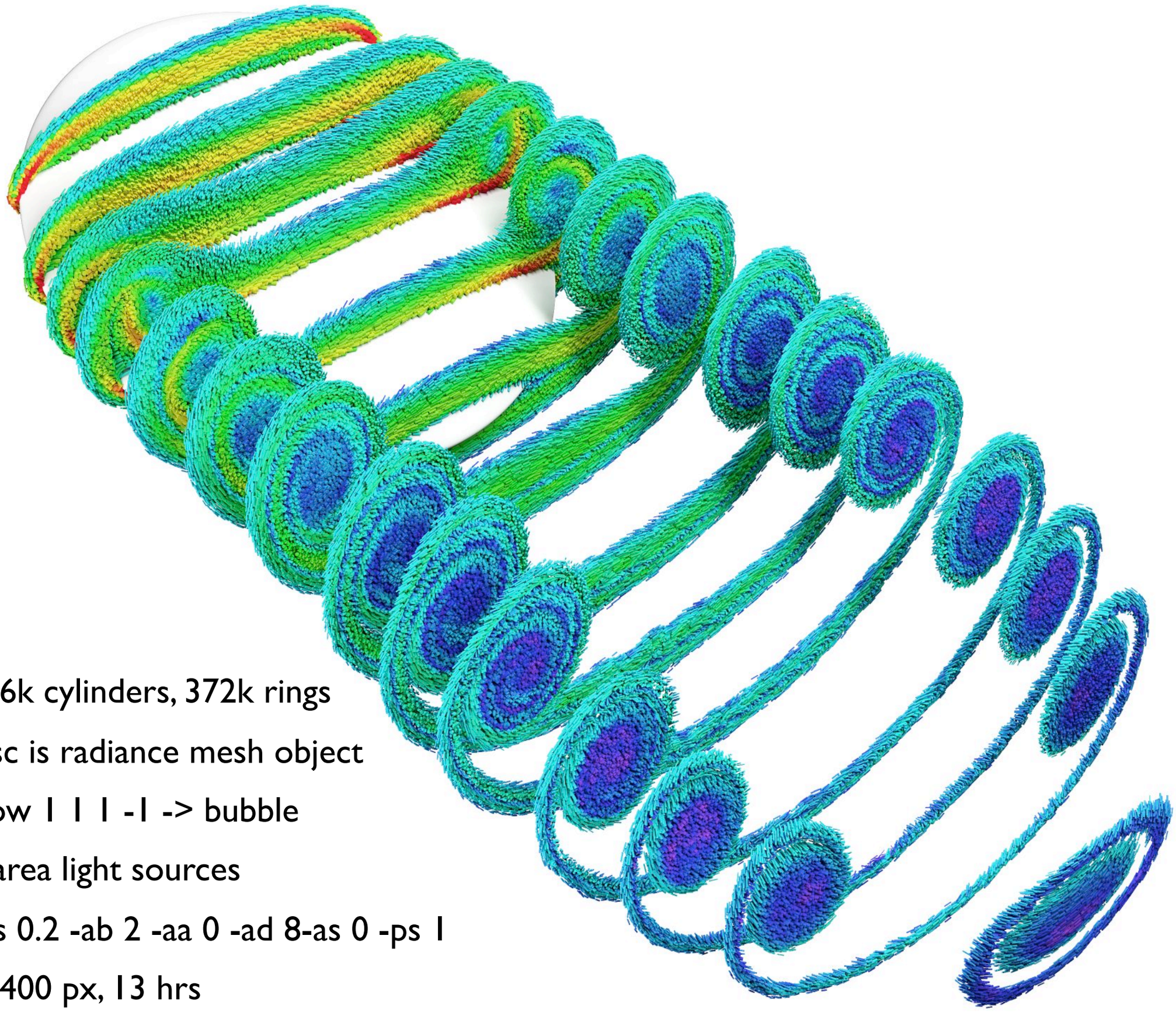
Sky

Arguably the most important part of any Radiance rendering is the sky.

Aside from the default gensky, you have several options

Starting with the simplest, a uniform-radiance “bubble” (opposite of a sphere)...

Image for a publication, 2005



186k cylinders, 372k rings

disc is radiance mesh object

glow 1 1 1 -1 -> bubble

3 area light sources

-ds 0.2 -ab 2 -aa 0 -ad 8-as 0 -ps 1

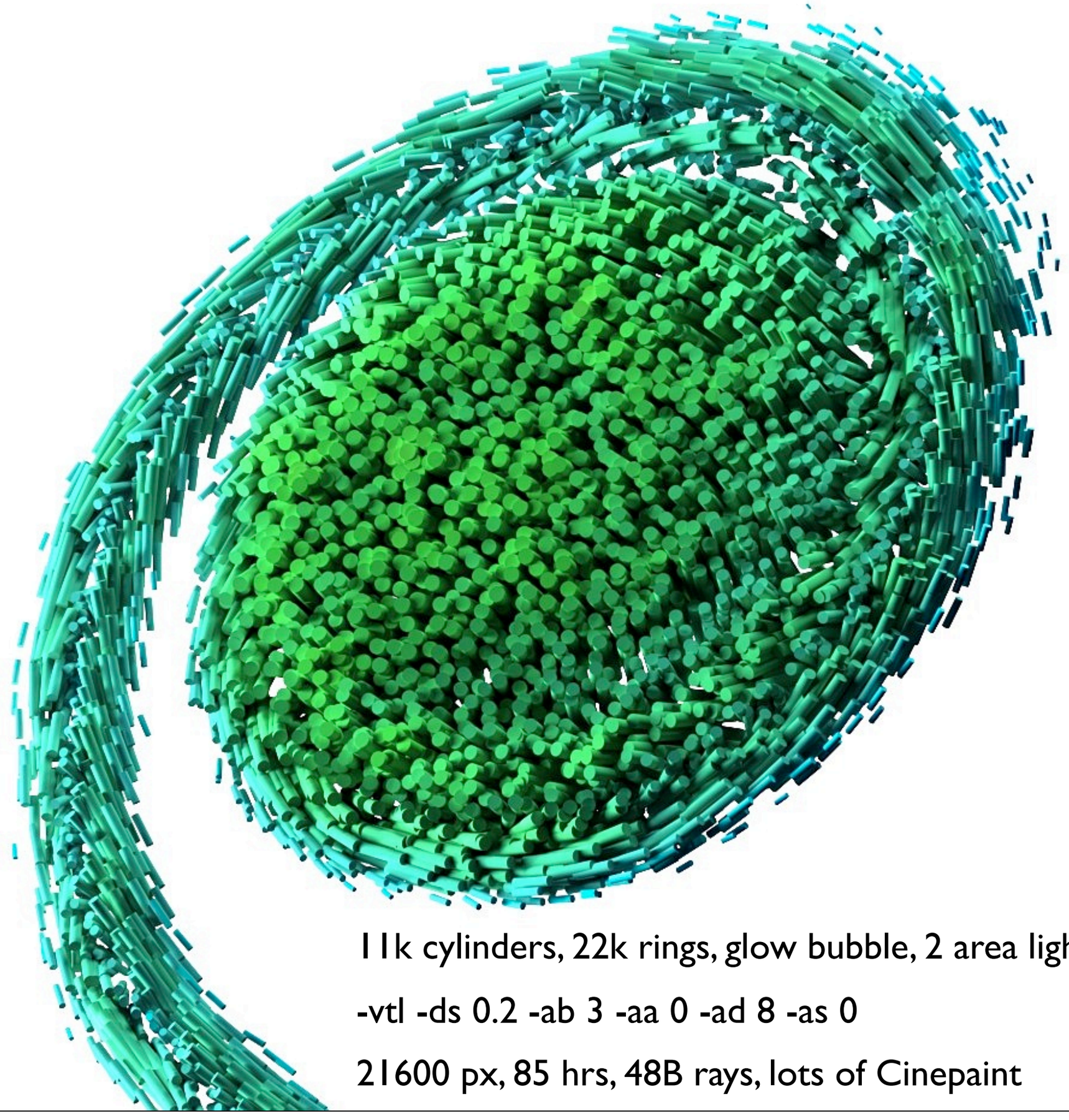
14400 px, 13 hrs

This is simply a geometry in a glow bubble object.

Use radiance because images with light interreflection communicate 3D shape better. This is essential for fluid dynamics!

Let’s take a closer look at one of these swirls...

Green Tendril, 2005

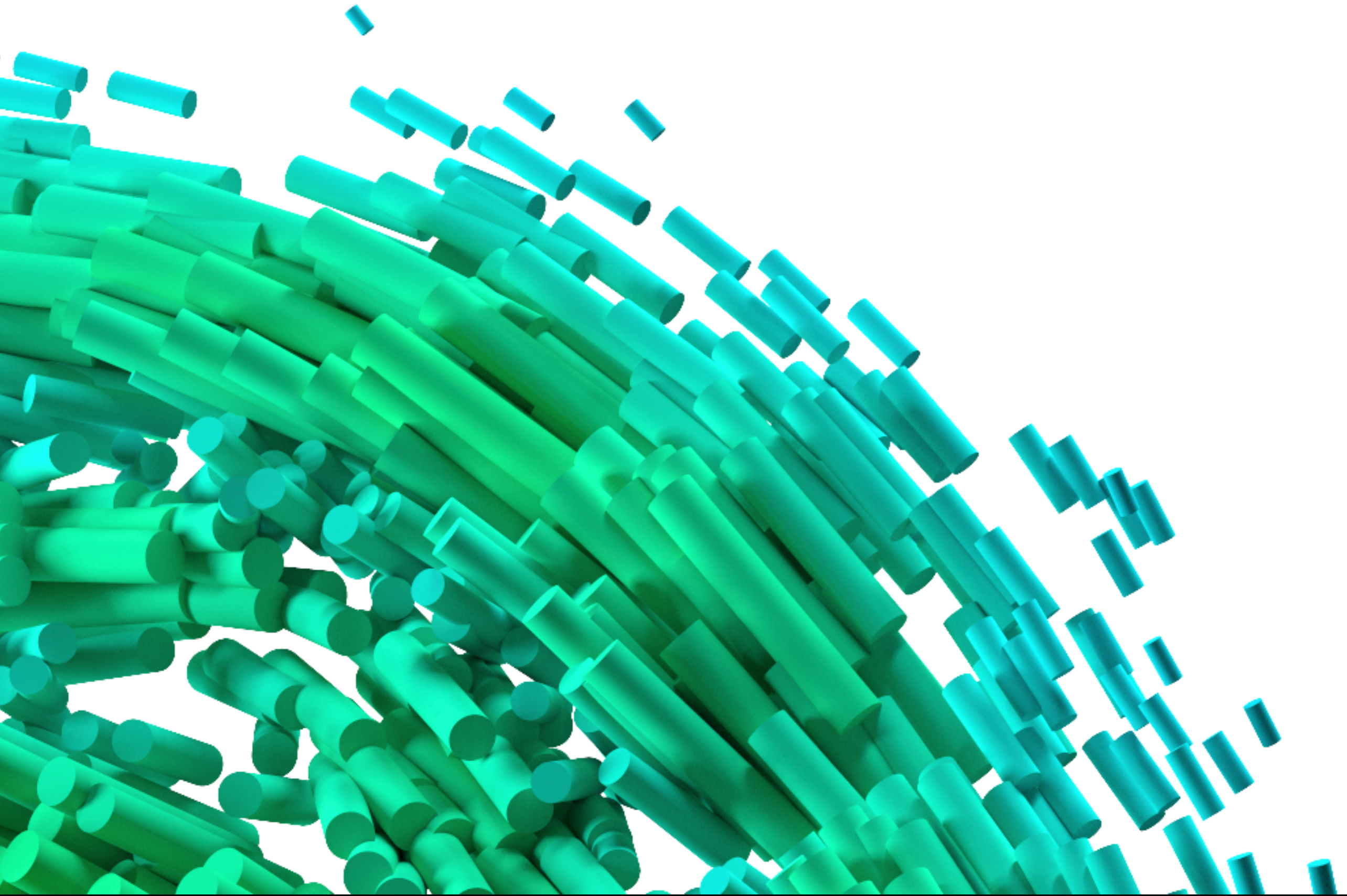


11k cylinders, 22k rings, glow bubble, 2 area lights

-vtl -ds 0.2 -ab 3 -aa 0 -ad 8 -as 0

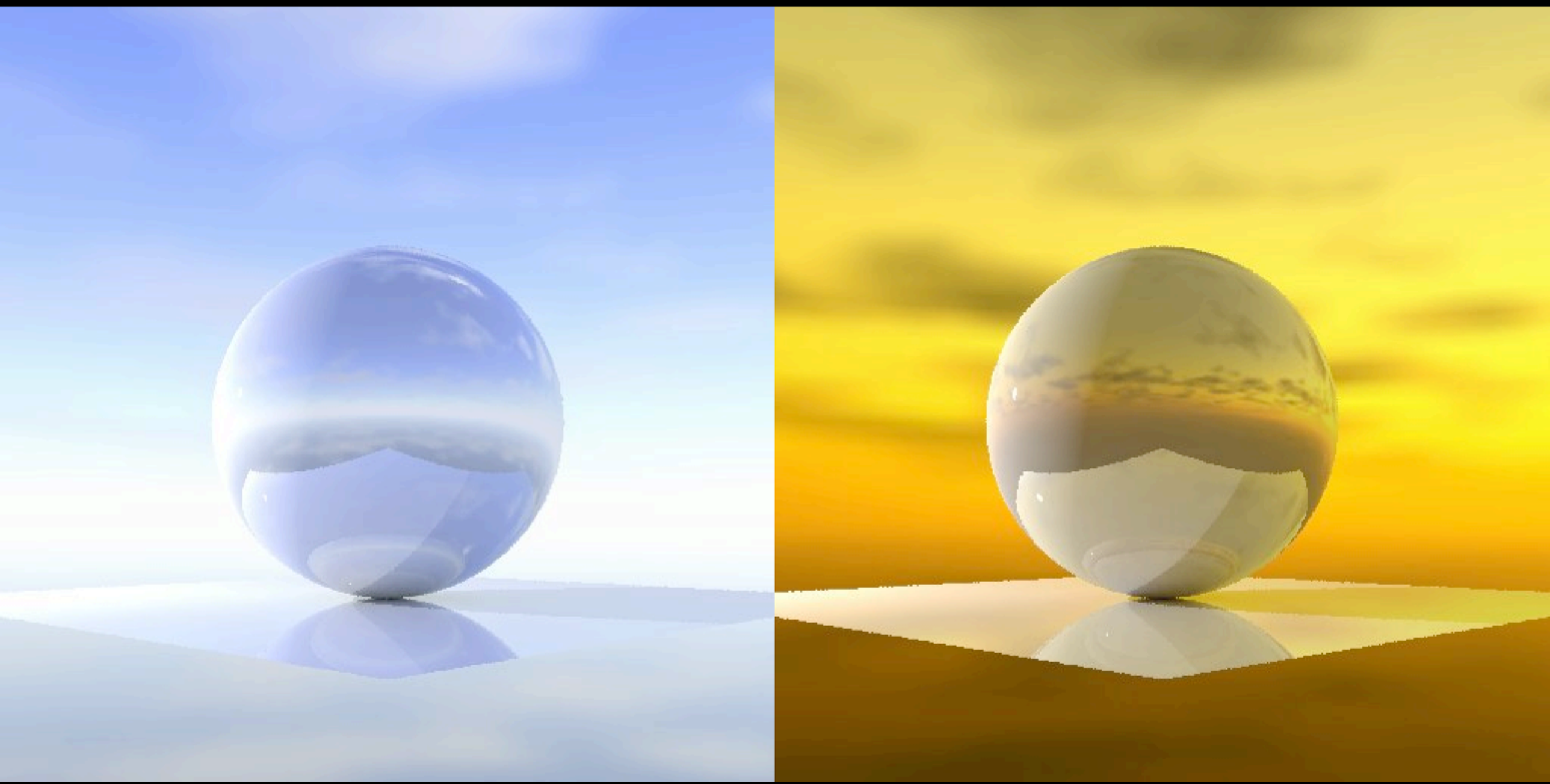
21600 px, 85 hrs, 48B rays, lots of Cinepaint

Looking closer, see individual particles



Here is closeup – one cylinder per particle

Another option is to map an image to the sky dome...



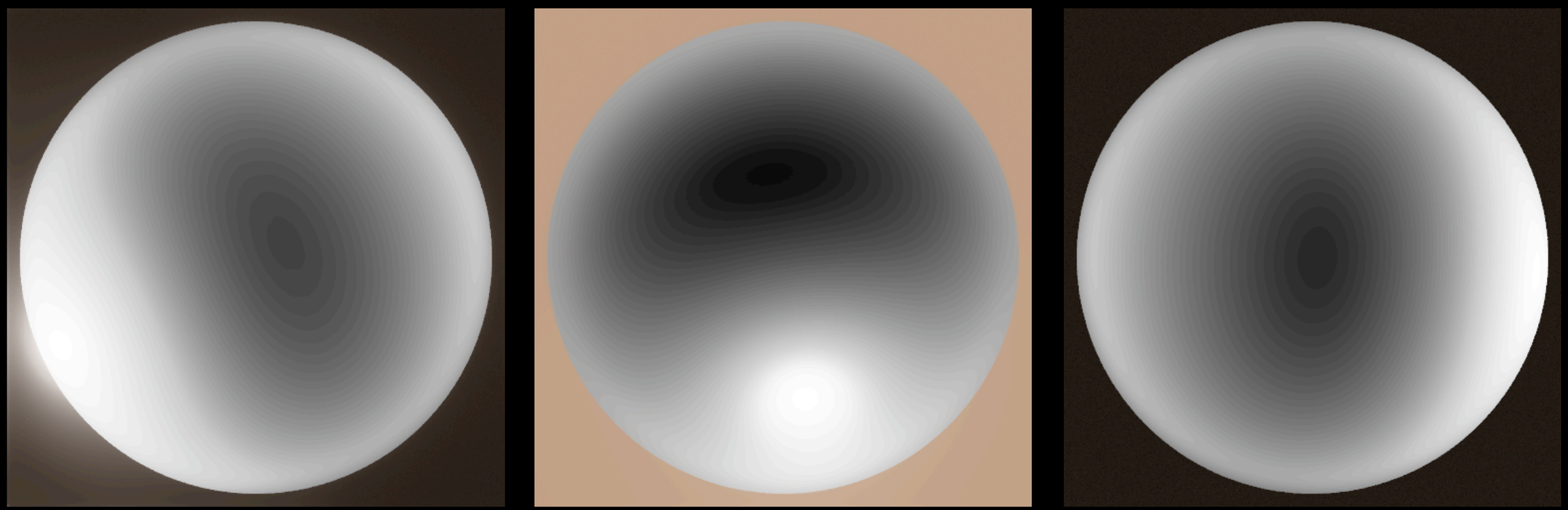
Reflective sphere and plane, HDR image mapped to dome

One way: map a HDR sky dome to your model, ignore gensky altogether

Problem: not much resolution, difficult to find HDR sky images, must add sun separately

There’s some research and code out to assist with that

gensky -t 2



7am 12pm 6pm

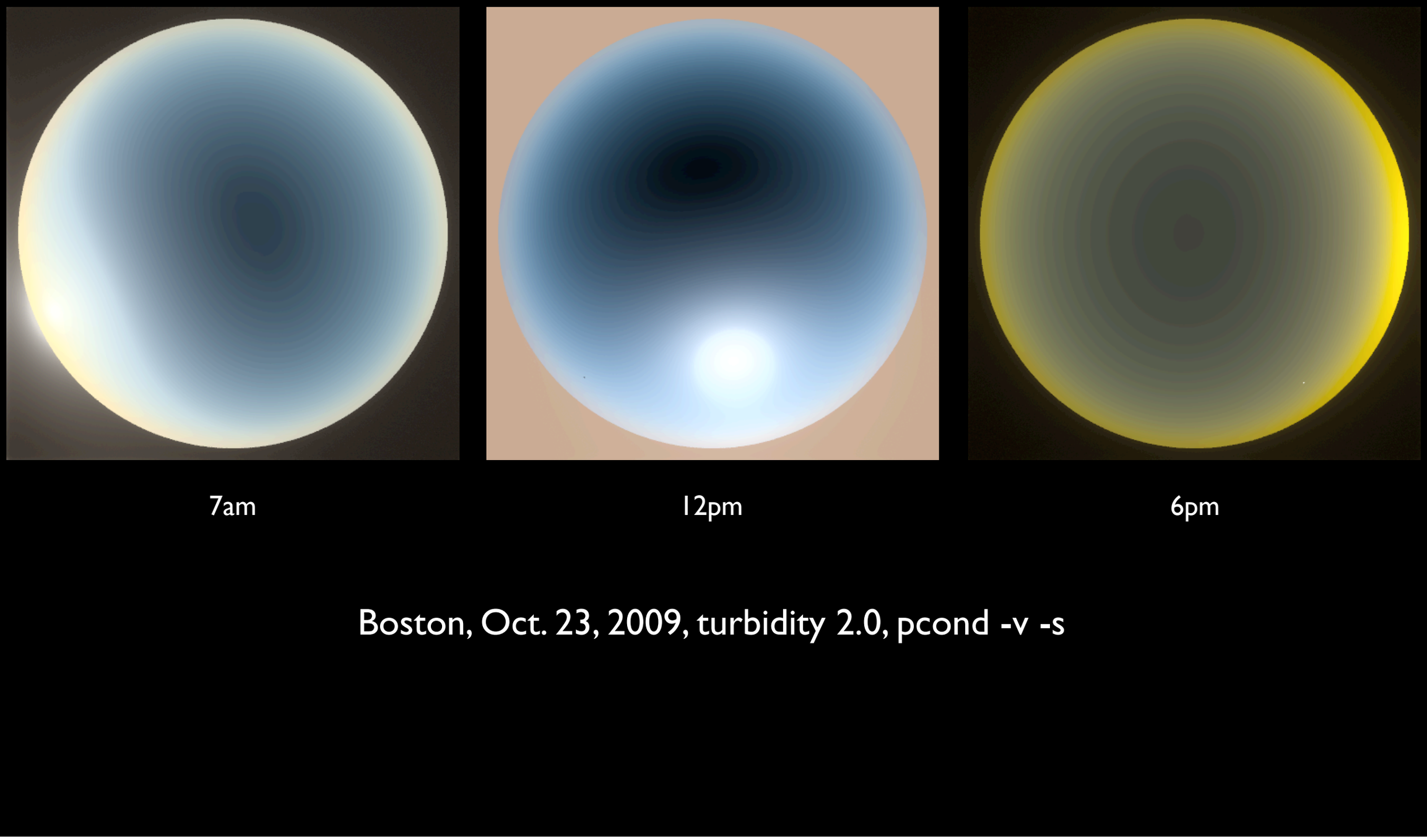
Boston, Oct. 23, 2009, turbidity 2.0, pcond -v -s

Here's the familiar "gensky", with turbidity 2.0

Gensky is only a "brightfunc", so it is monochrome.

Facing up, toes pointed south

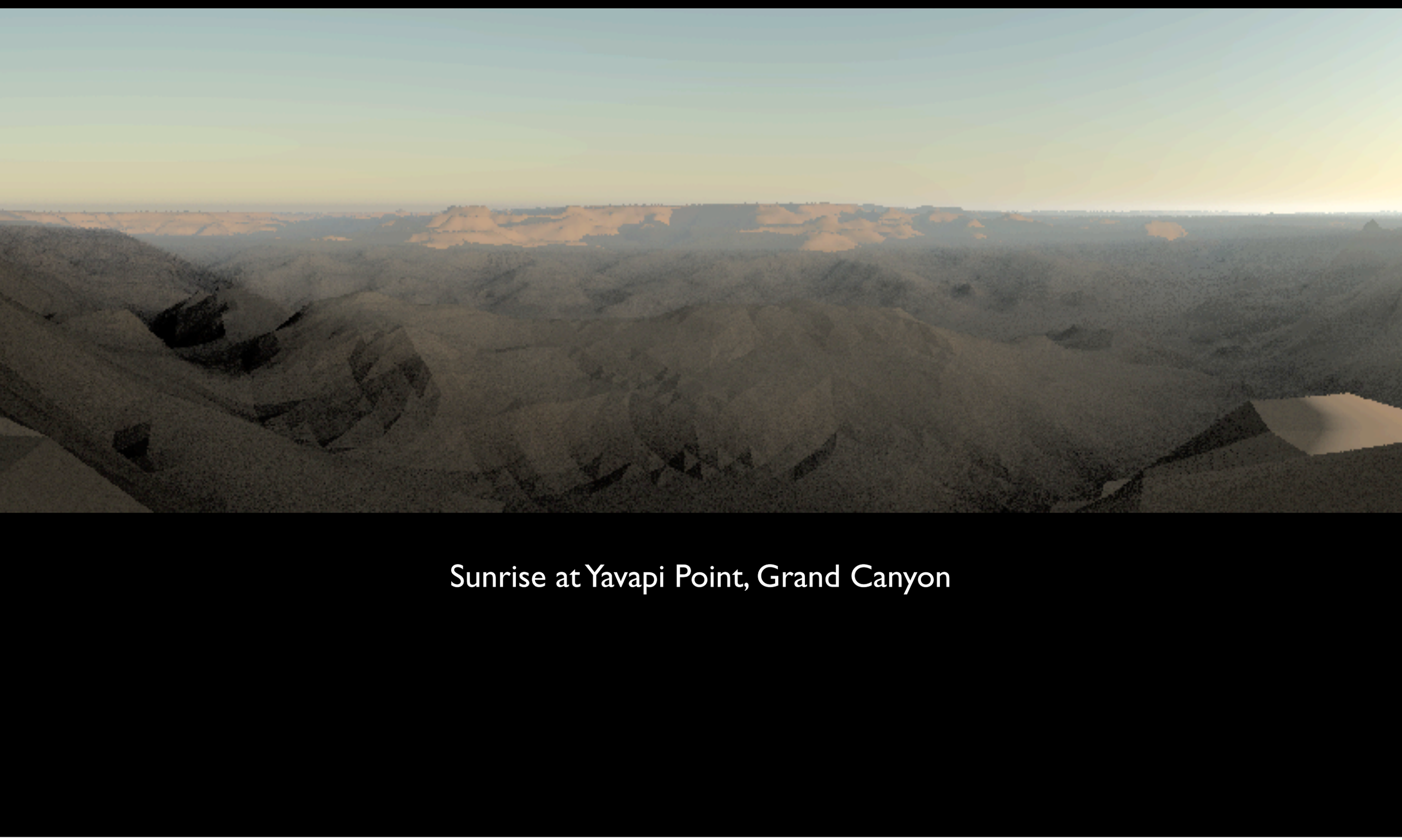
genutahsky -t 2



7am 12pm 6pm

Boston, Oct. 23, 2009, turbidity 2.0, pcond -v -s

Here's something I put together two years ago: the Preetham-Shirley-Smits sky model (curve fit)



Sunrise at Yavapi Point, Grand Canyon

Here are a lot of effects in action:
Sky color model, terrain map, mist to show the sun rays

Get this and other good stuff http://markjstock.org/radiance/

Open House, 2006

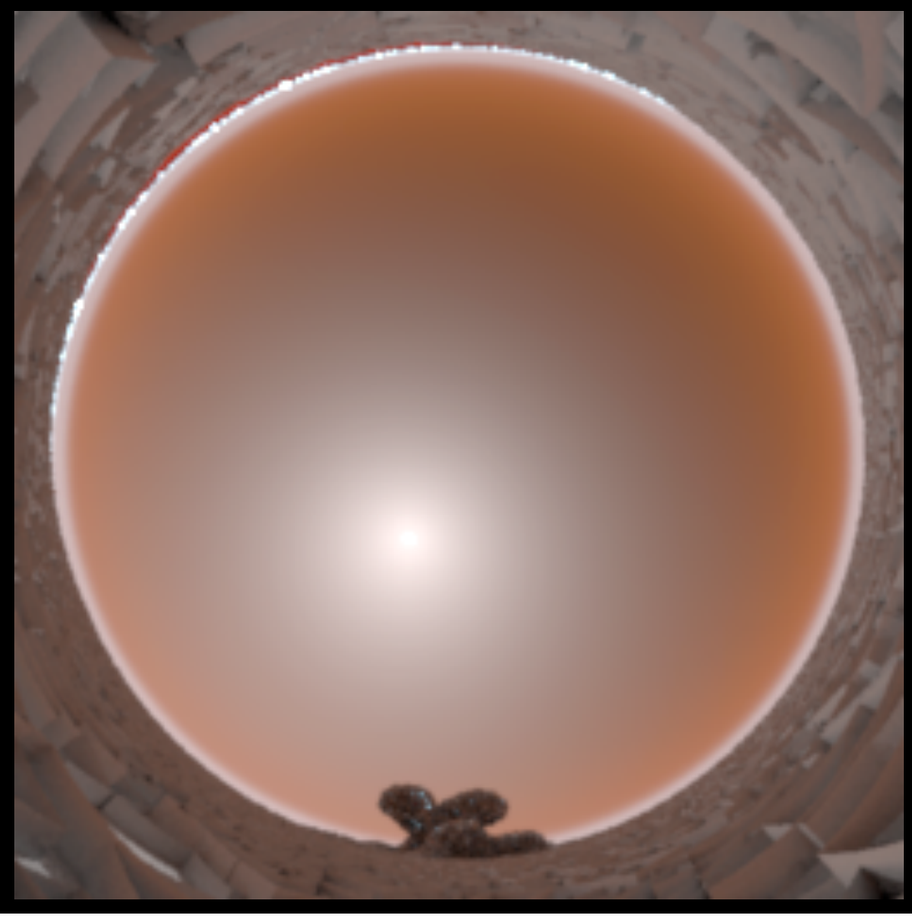
39k triangle mesh,
395k polygons,
all plastic (.7 .7 .7 0 0)

Custom skycolor.cal

-ab 3 -aa 0 -ad 8 -ps 1 -u

-vta -vh 18.75

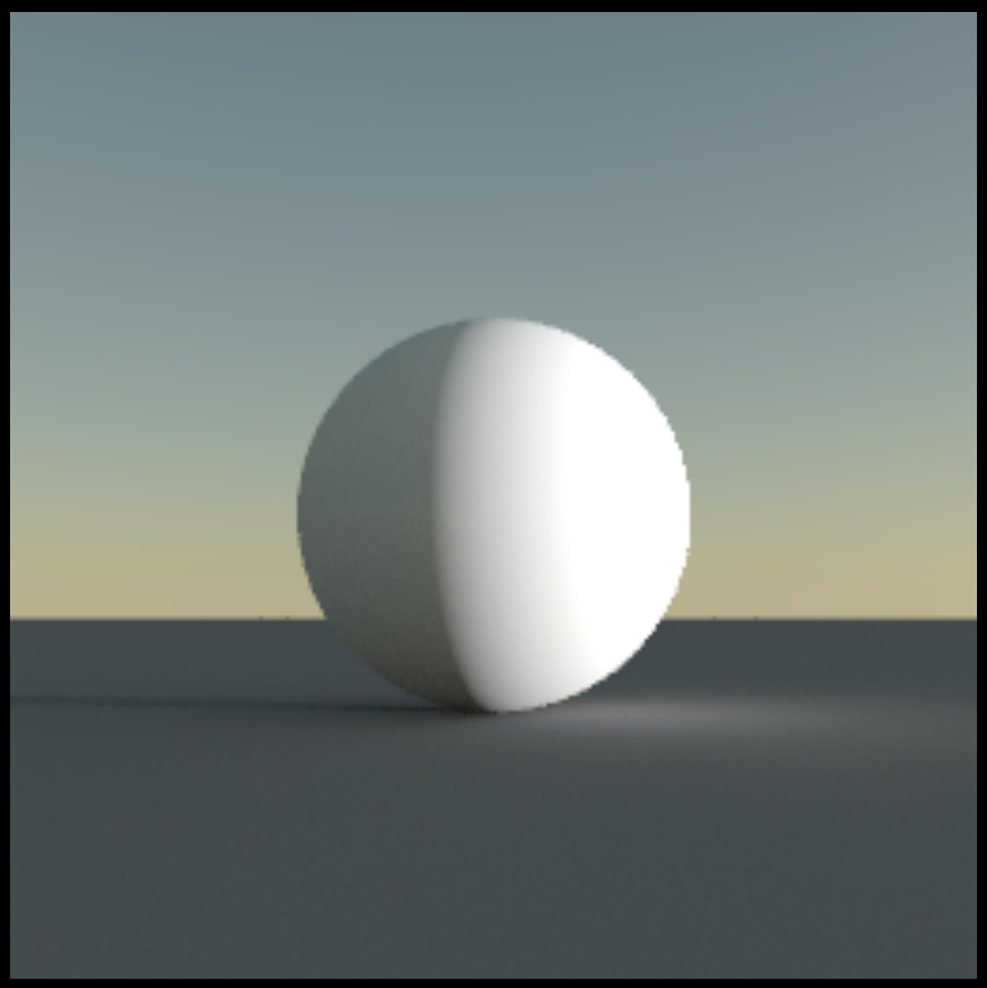
36k x 36k, 25 days



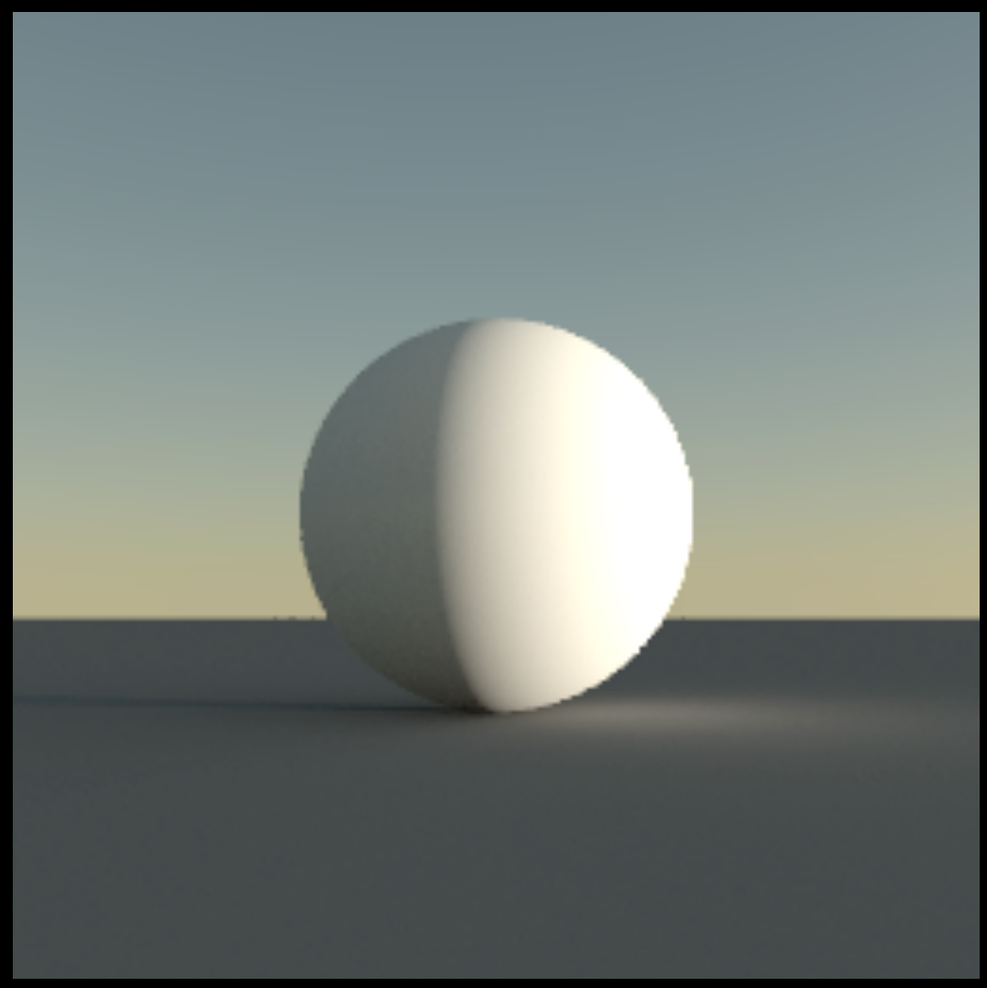
Fiddle with the gensky .cal file to make custom skies...like this pink sky

I copied skybright.cal and added colors to make skycolor.cal

Non-D65 sun



6500K



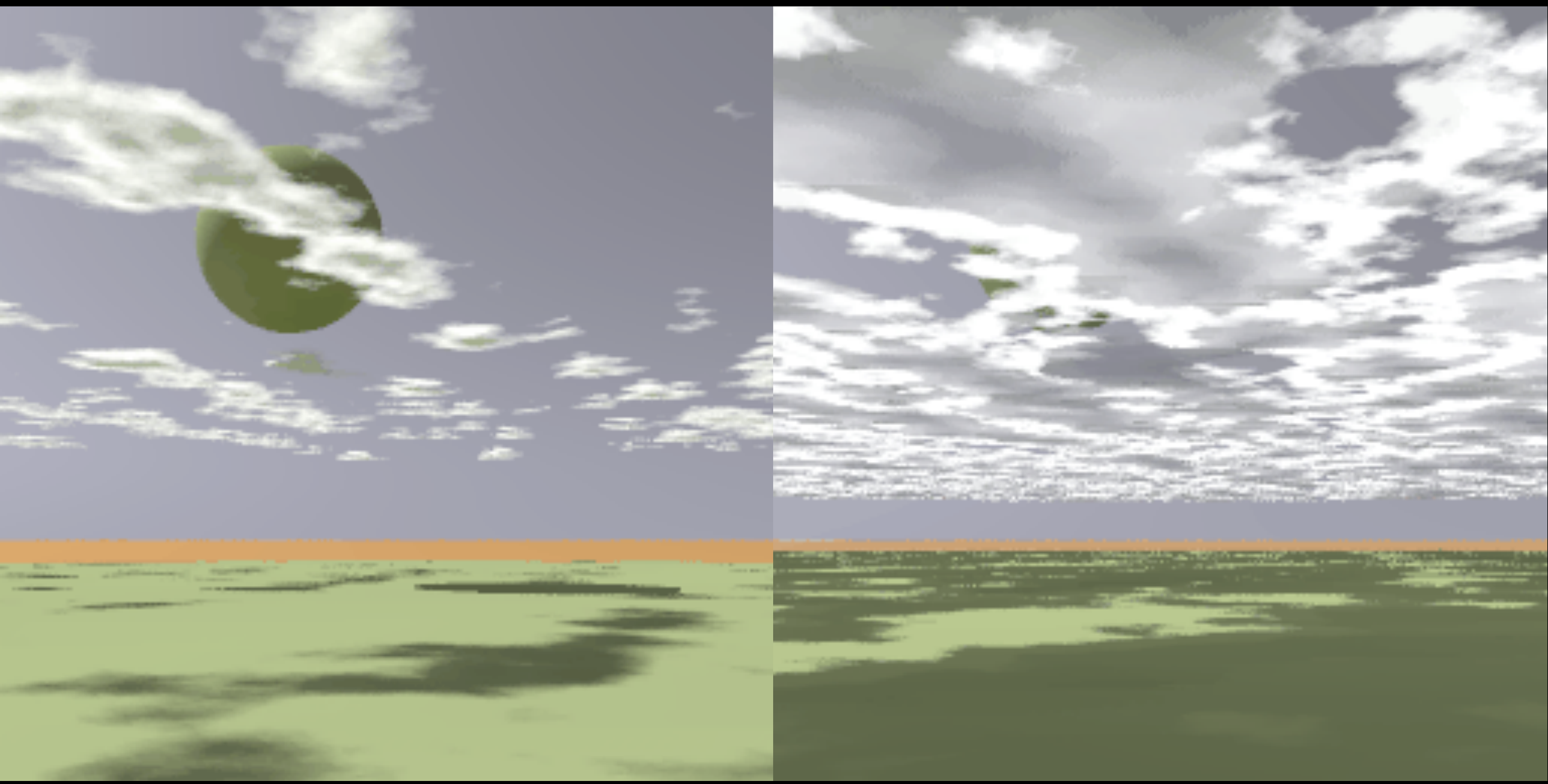
5216K

void light solar 0 0 3
1.80e+06 1.62e+06
1.28e+06

Finally, to add a bit more color to your sunsets, try changing the color of the sun.

Clouds

I have tried every idea that I've had to try to make clouds in Radiance.
It's tough.

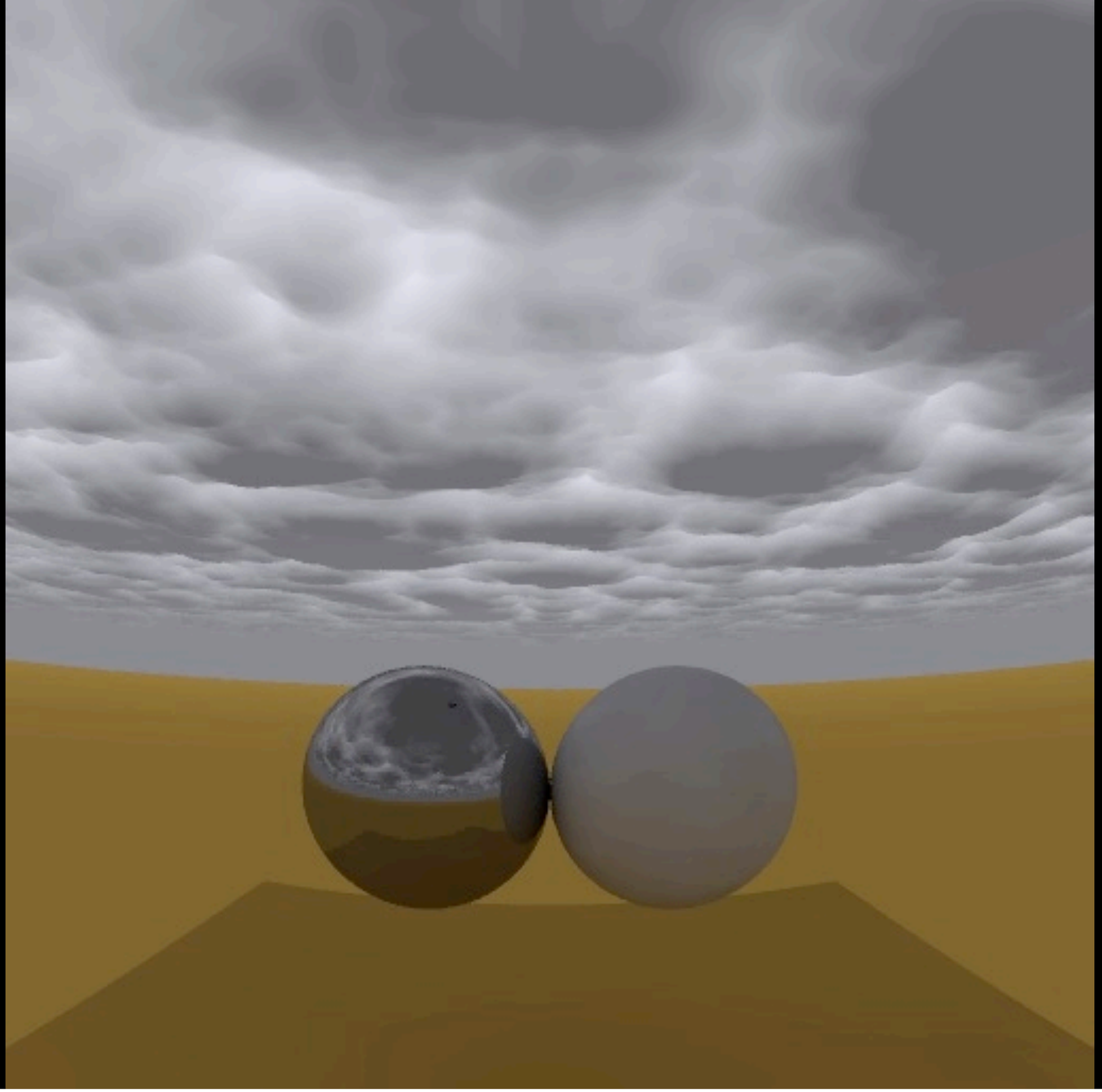


cloud.cal file:

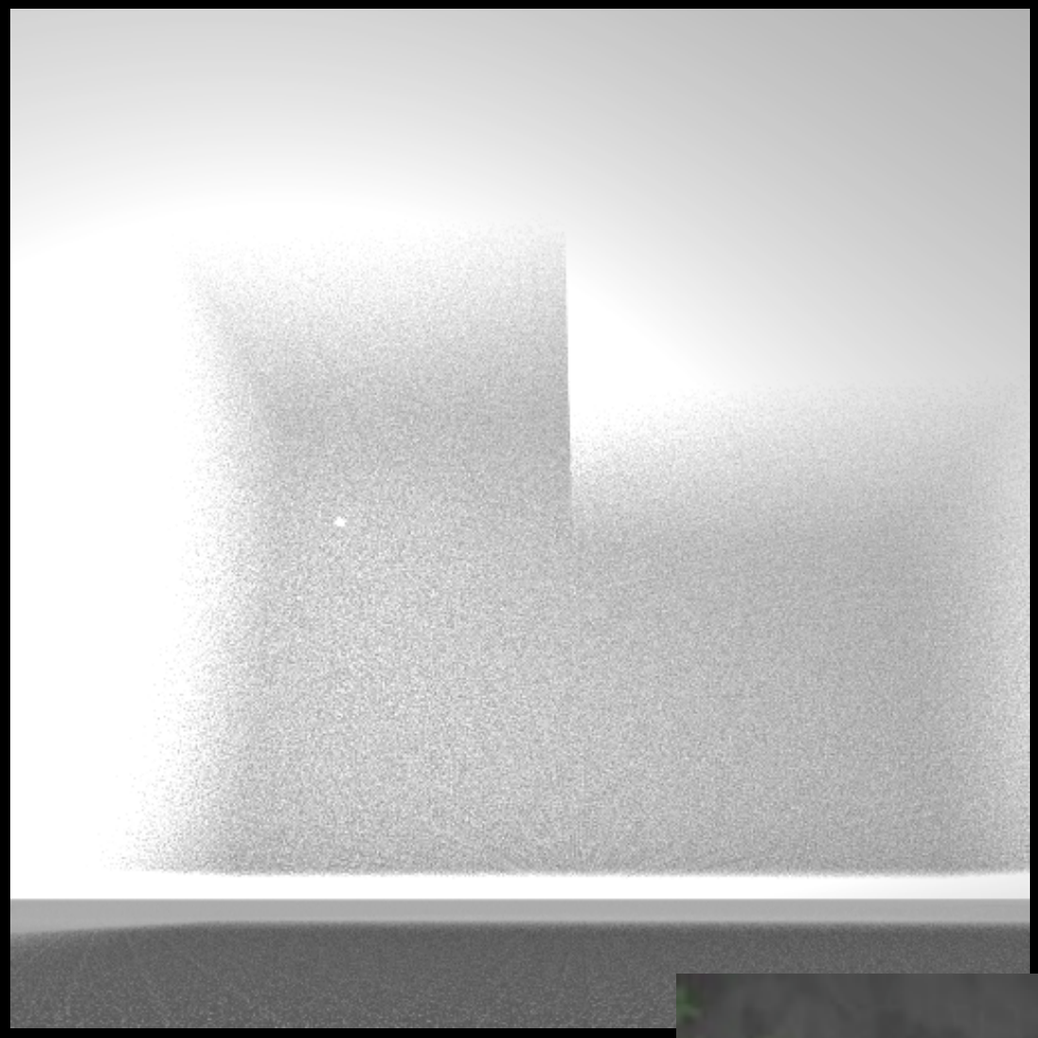
```
cloud = bound(0,(.5*(1+fnoise3(Px,Py,Pz))-A1)/(A2-A1),1);


void mixfunc cloud_pattern
6 cloud_white void cloud cloud.cal -s 400
0
2 0.3 0.4
```

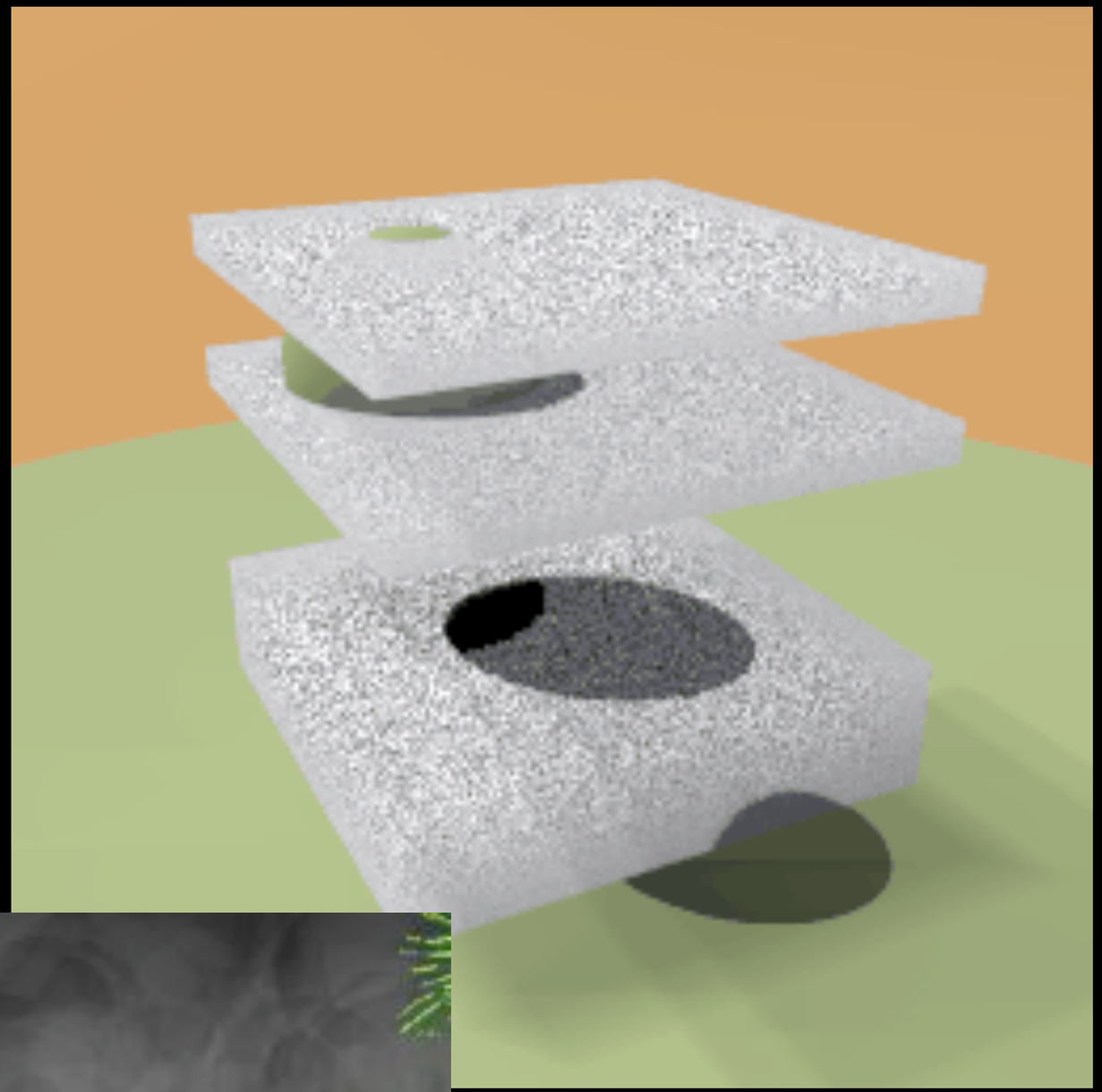

It turns out----you can get “good enough” with a handful of layers of “trans.”
This is an old texturing trick from the 80s
One layer on left, three on right, two ambient bounces



A more complex .cal file can generate a somewhat realistic cloudy day.
This is just a single polygon in the sky, filtering sunlight



Lots of tiny droplets

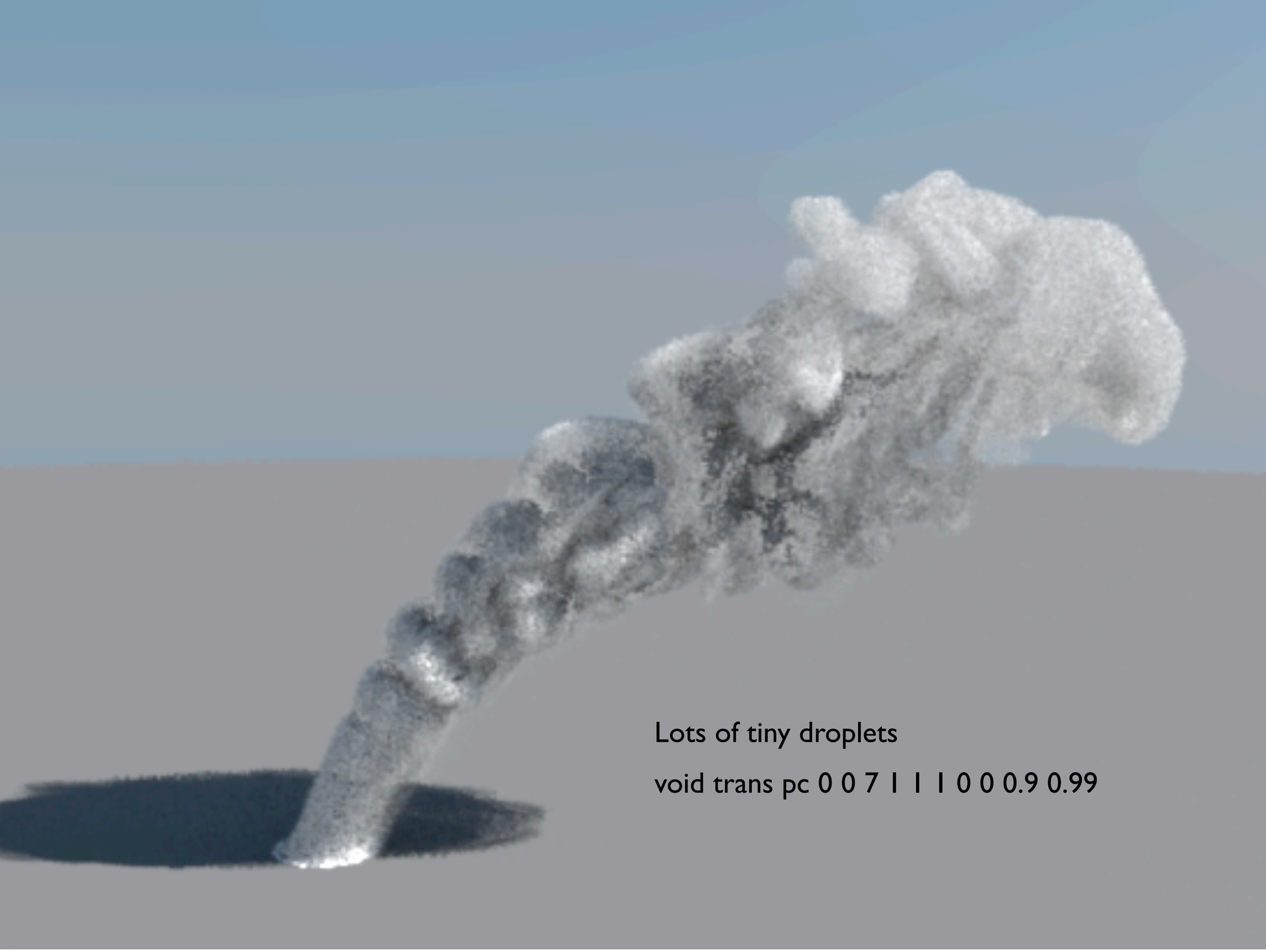


Volumes of mist

What was I thinking?

But, oh, using “mist” to make clouds…

Mist objects cannot overlap with anything else.
You have to use antimatter object to cut them



Lots of spheres of trans. Not bad.

730619 element
Radiance mesh

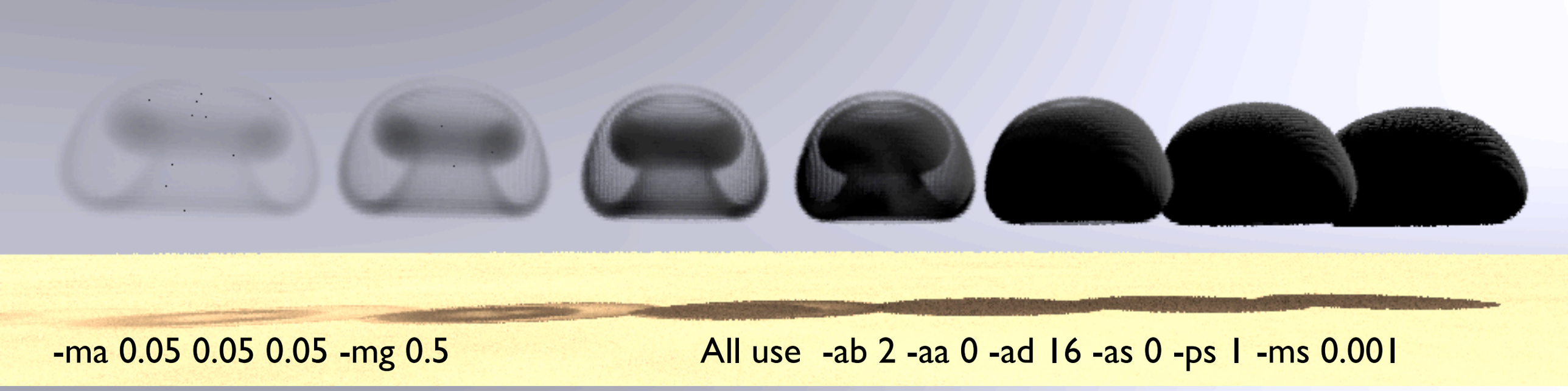
-ab 2 -aa 0 -ad 8

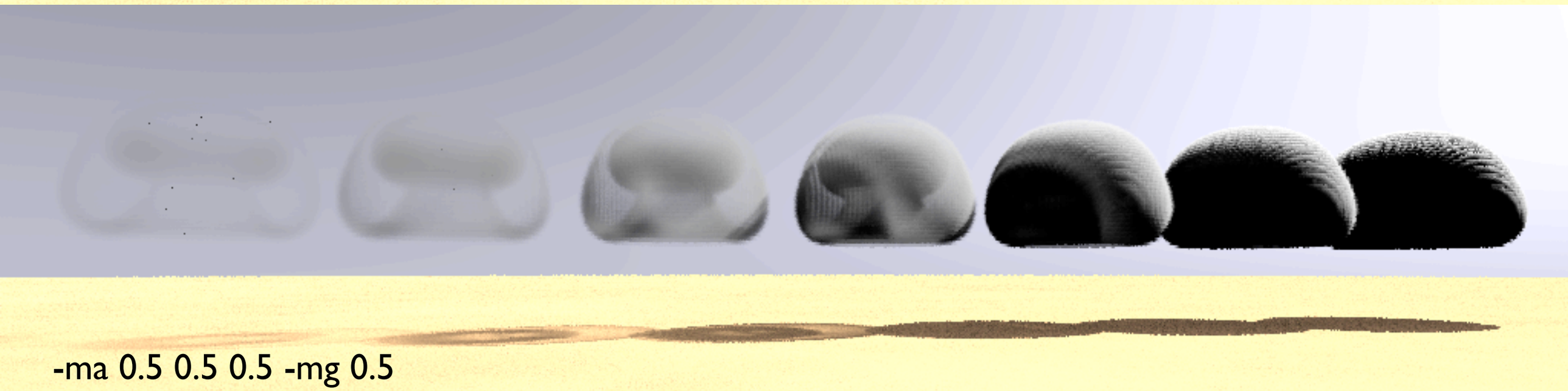
void mist def 2 sun sky 0 7 30 30 30 0.5 0.5 0.5 0.1

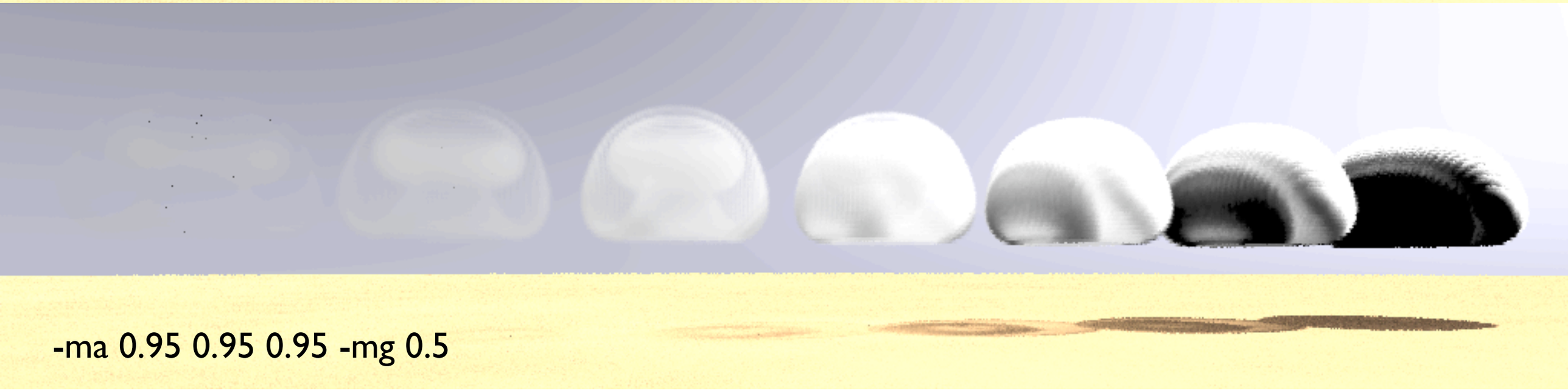
-ms 0.0003 -ab 1 -aa 0 -ad 8

Triangle mesh must be perfectly closed---see the errors on the right?

Extinction coefficient of 30, albedo of 0.5, HG coeff of 0.1, very fine sampling!







Study of mist parameters
Each object made of a grid of *non-overlapping* cubes

Extinction coeff varies 3-fold between adjacent objects, from peak of 1 to 1000
Each object can contain 100-fold variance in extinction coeff.

228k cubes (each 6 polys)

Extinction coeff varies over 1000 levels

-ma .96 .96 .96 -mg .2 -ms .01

-ab 2 -aa 0 -ad 16 -as 0 -ps 1

Standard gensky

4k px, 26 hrs

This is as good as I have been able to get

Each cube can have a different extinction coefficient...

Crepuscular rays

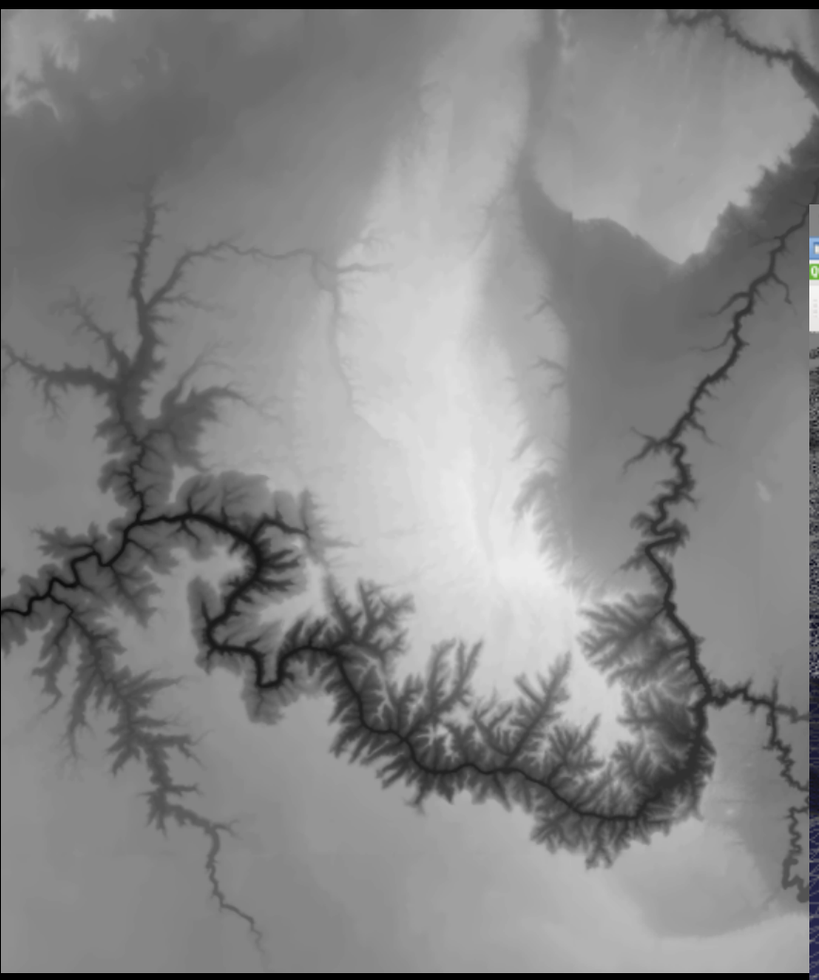
rpict -ds 0.001 -ms 0.1 -ab 2 -af ambfile -x 1536 -y 1536 -ad 64

pfilt -1 -e -4.5 -x /2 -y /2

Light source must be outside of mist volume

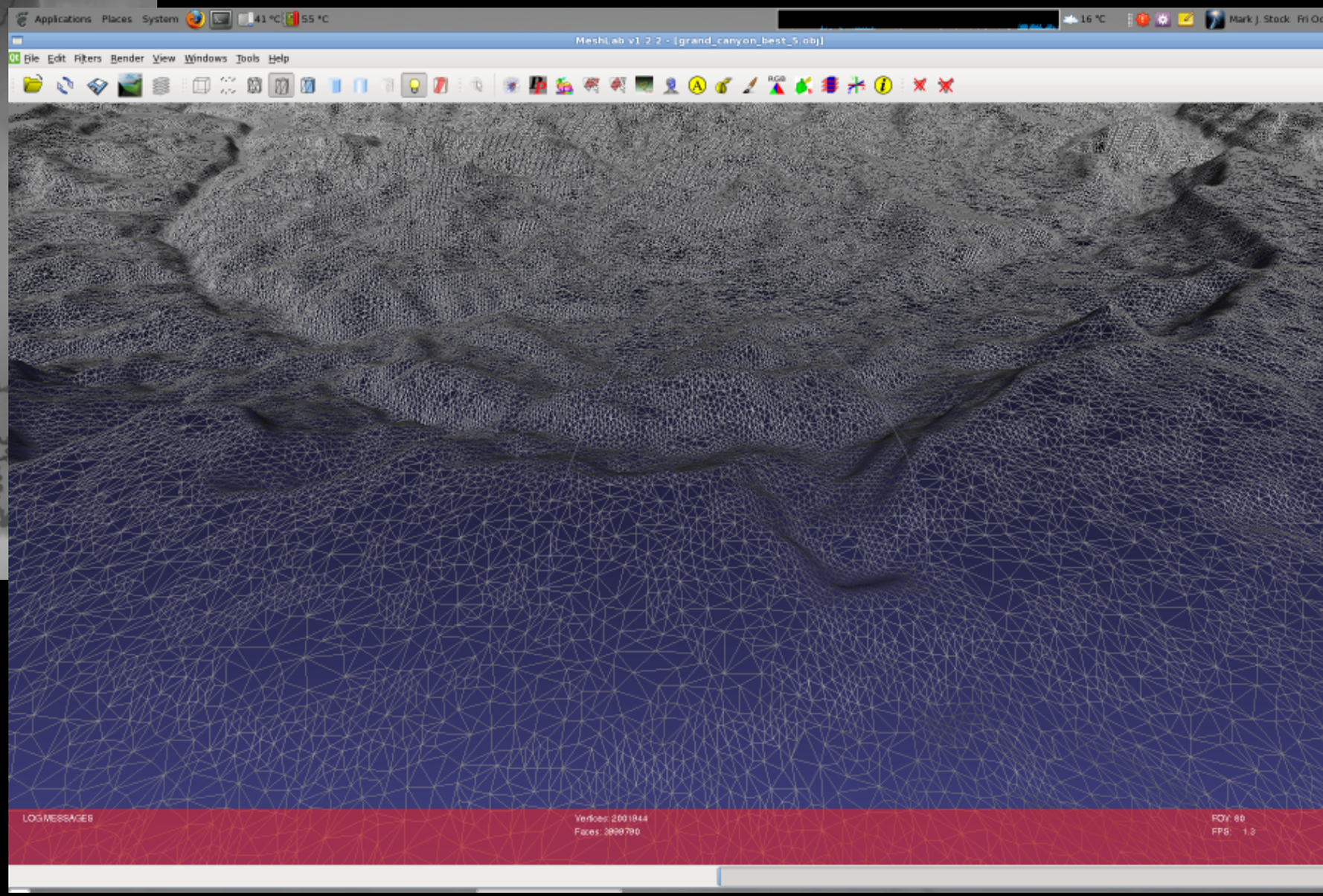
Land

DEMs



From this...

to this...

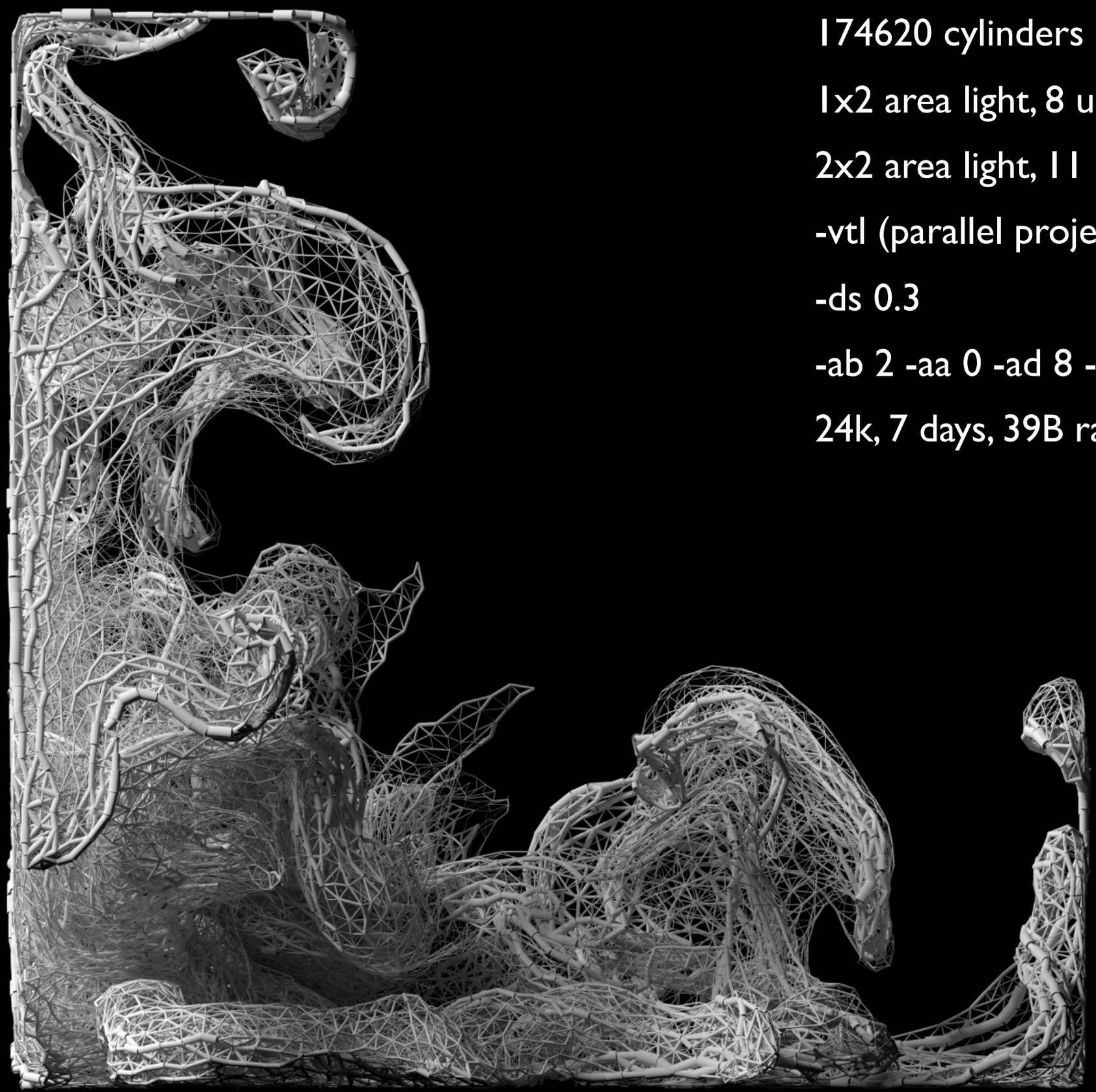


in no easy steps.

Digital elevation models

Rendering

View type



174620 cylinders (plastic .6 .6 .6 .0 .0)

1x2 area light, 8 units away

2x2 area light, 11 units away

-vtl (parallel projection)

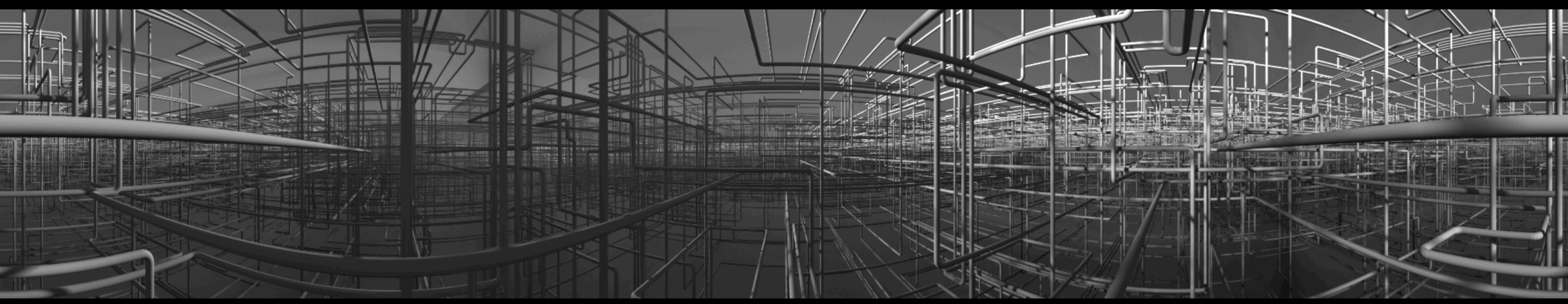
-ds 0.3

-ab 2 -aa 0 -ad 8 -as 0

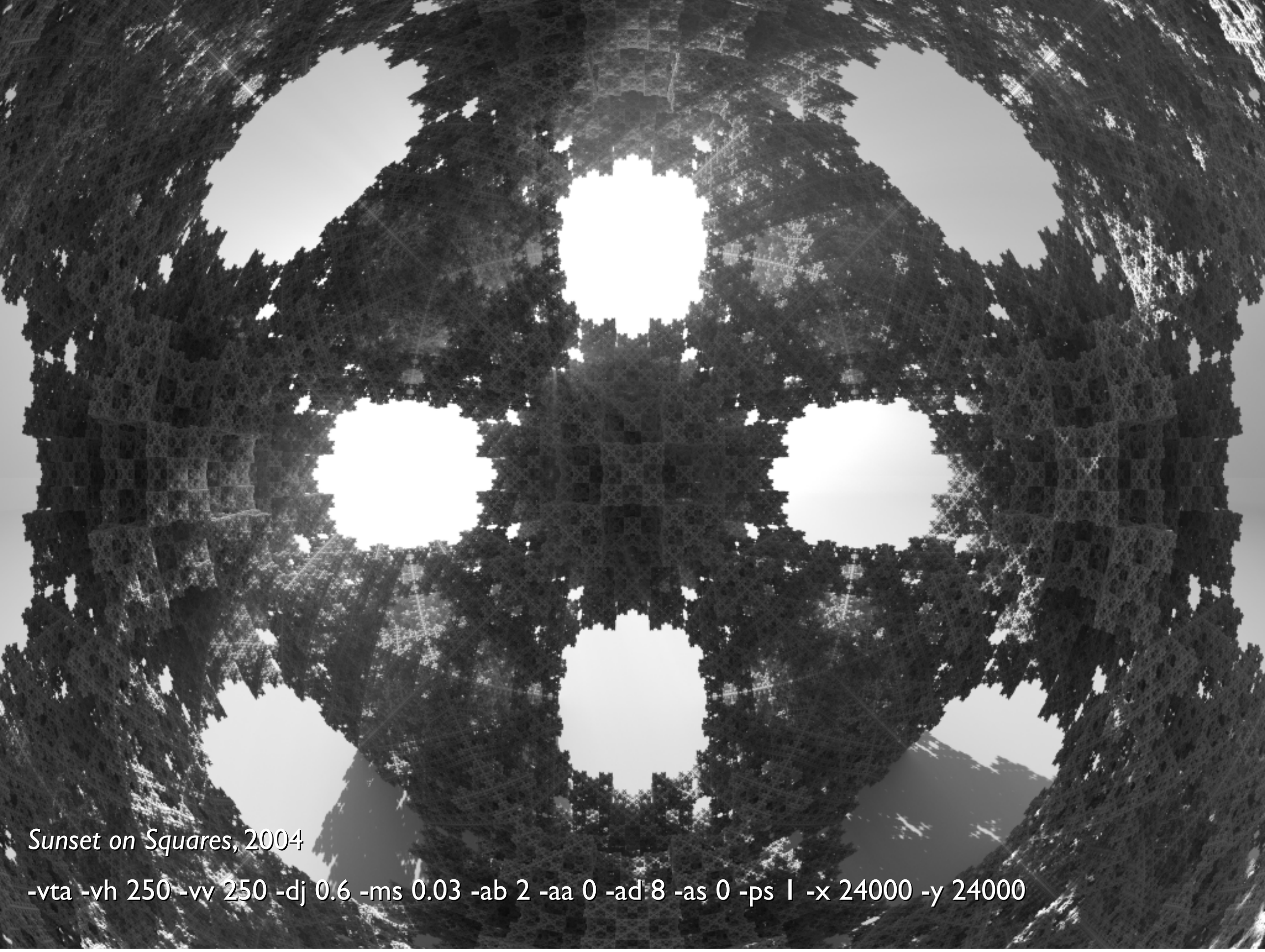
24k, 7 days, 39B rays

Mesh #3 Iso, 2000

this is “Mesh #3 Iso” an isometric rendering (-vtl)
first recognition (SIGGRAPH 2002)



Refinery #53, 2002 -vtc -vh 360 -vv 60 -ms 0.01 -ab 2 -ar 256 -as 64 -ds 0.01 -dj 1.0

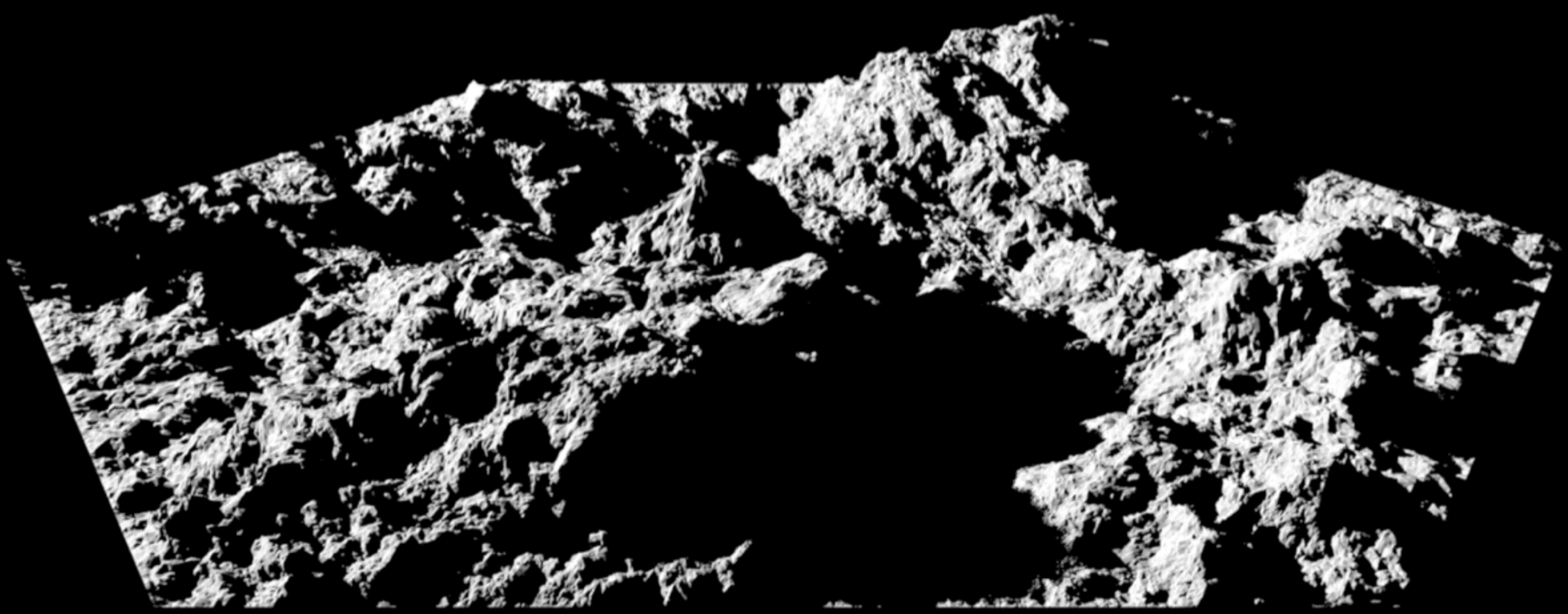


Angular fisheye, very wide angle

2.5 million cubes, all algorithmically placed with 2kB of files

Ugh, the ambient calculation

I don’t have to convince you that the ambient calculation is what makes Radiance output realistic.
It is intrinsically different than “ambient occlusion”, which merely checks occlusion with the sky dome (approximating a CIE overcast day).
Just to give a quick example...



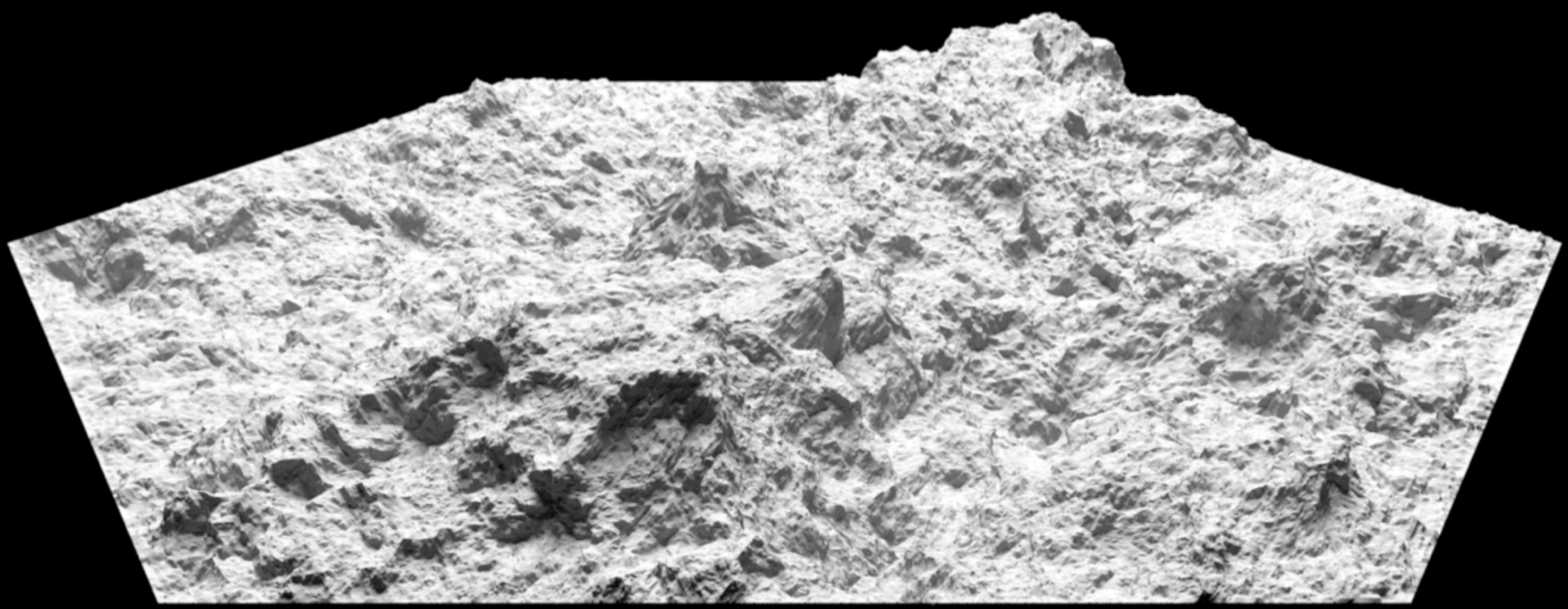
538k triangles (plastic .2 .2 .2 0 0); small ring light; -ds 0.1 -ab 0; 4k x 2k

Here’s the “before” image

538k triangles (plastic .2 .2 .2 0 0); small ring light; -ds 0.1 -ab 3; 4k x 2k

And this is “after,” with the ambient calculation turned on.

And what if we increased the size of the light source from 5% to 100% of the scene...



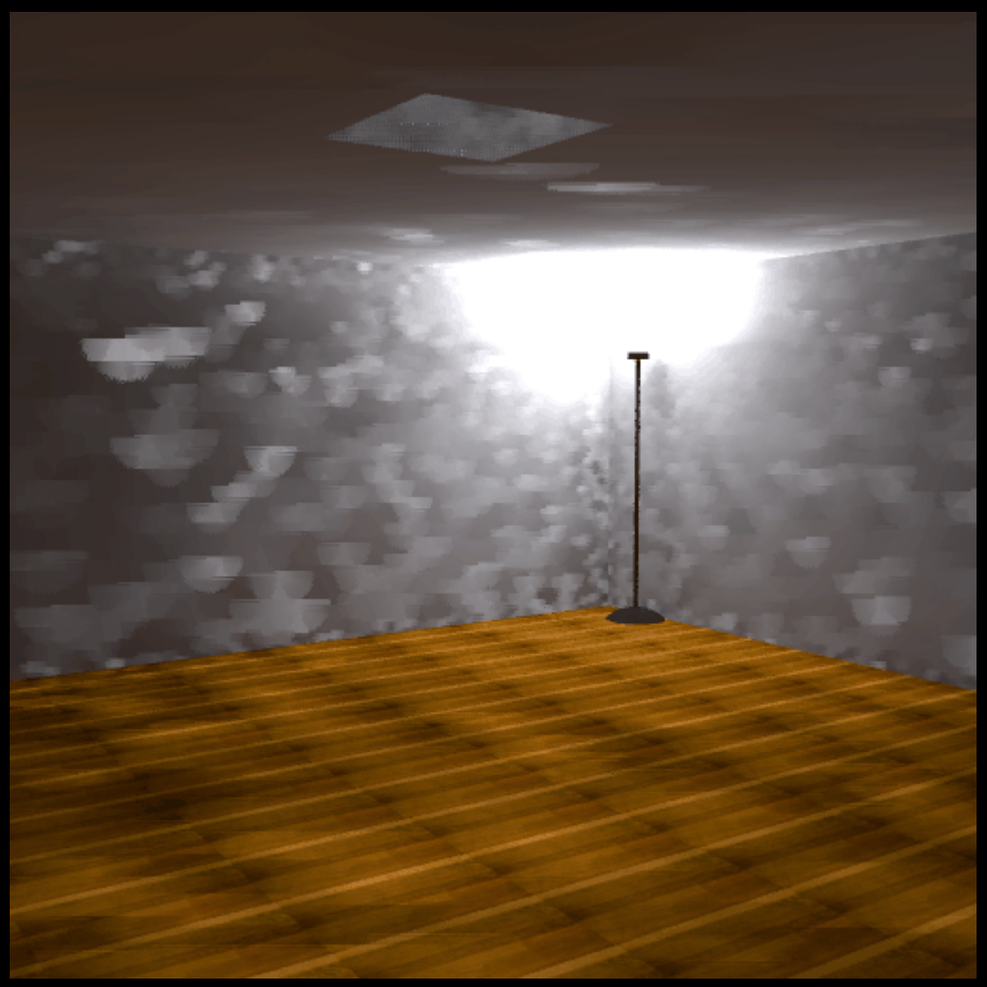
538k triangles (plastic .2 .2 .2 0 0); large ring light; -ds 0.1 -ab 3; 4k x 2k

...the character changes again.

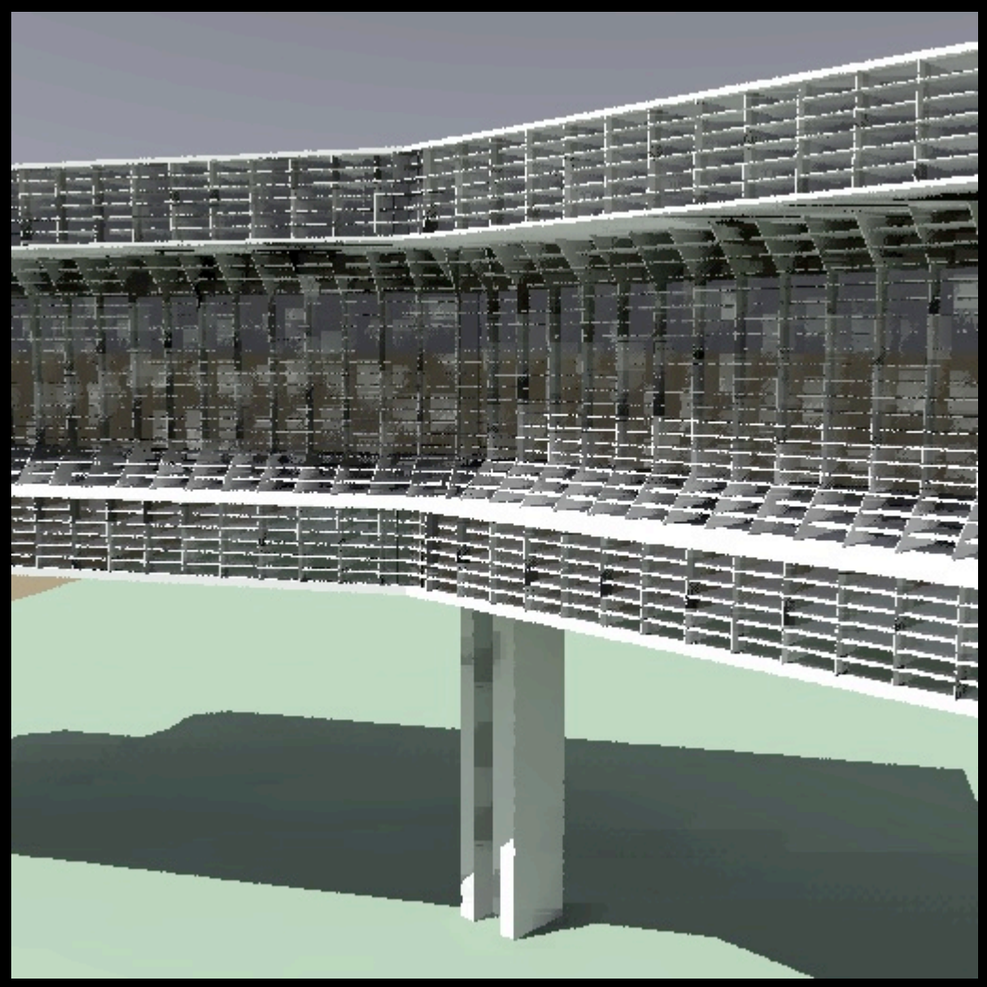
Here is where -ds turns on: if the size of the light divided by its distance is greater than the -ds threshold, that light source is subdivided (as a square) and tested again.

In this case,

Common ambient problems



Parameters problem - “splotches”
on open expanses of geometry



Geometry problem - “splotches”
creep out from under walls, floors

On the left, we learn that our final rendering is going to take a lot longer than we had anticipated. We need to play w
the ambient options in rpict.

On the right, our primary problem is that we have used a single polygon for the back wall of each of the little cells.
Polygons shouldn’t meet in a “T”, they should meet in an “L”

Fixing splotches - I

- Can light even get there?
  Increase -ab
  2 to 5 are good values
- Is the range of light levels large?
  Increase -ad and -as
  Try -ad 1024 -as 512

Fixing splotches - 2

- Are the splotches too large?
  Increase -ar
  -ar 512 for starters
- Do you need a bigger hammer?
  Drop -aa closer to zero
  Try -aa 0.05

-aa 0

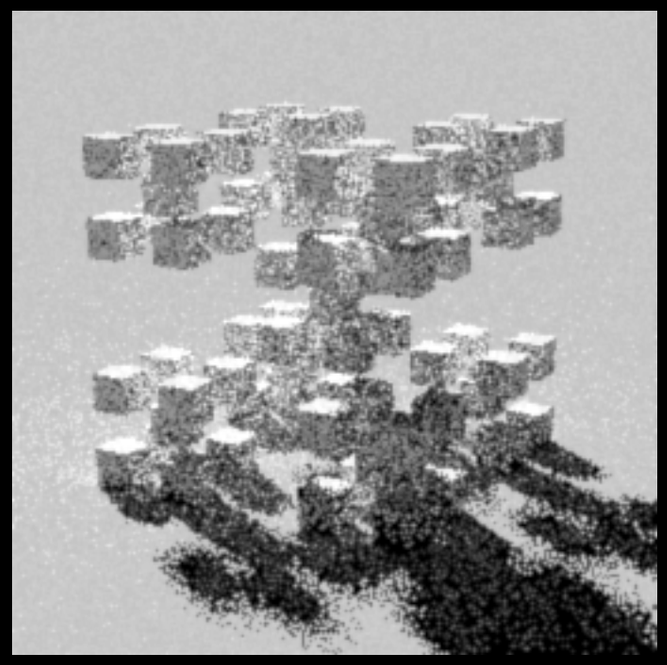
Turning off ambient caching

Why turn off caching?

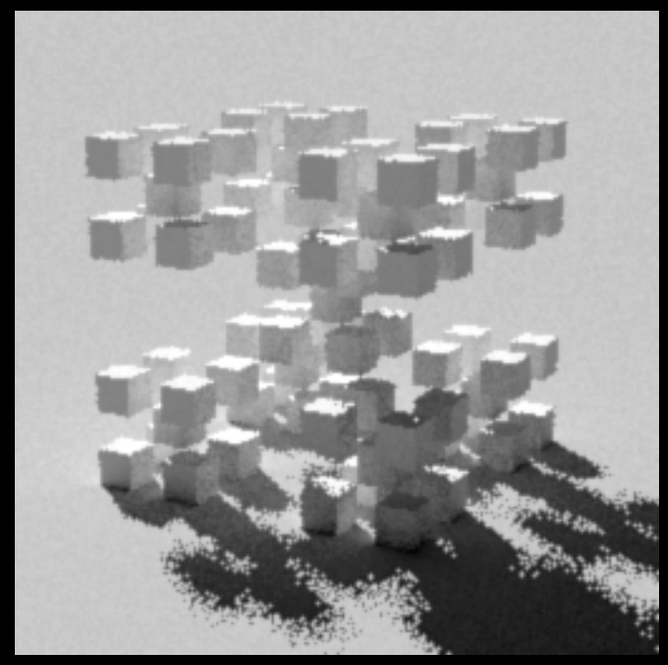
- Not enough resources for ambient calc.
- Geometry too detailed
- Geometry doesn’t meet at edges
- Pixel noise is better than splotch noise
- Need to render in parallel

How to “-aa 0”

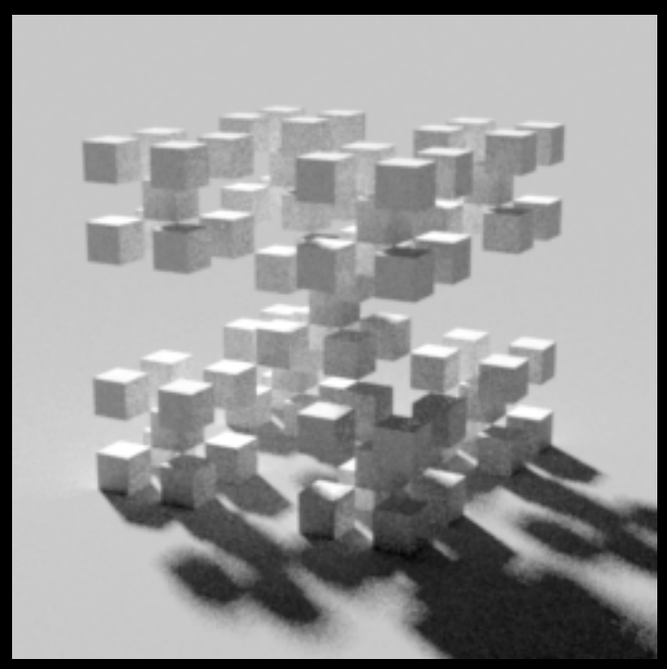
- Keep ambient bounces low “-ab 2”
- Keep ambient divisions low “-ad 16”
- Turn off ambient supersampling “-as 0”
- Turn off pixel subsampling “-ps 1”
- Turn on uncorrelated sampling “-u+”
- Render large and reduce later (oversample)



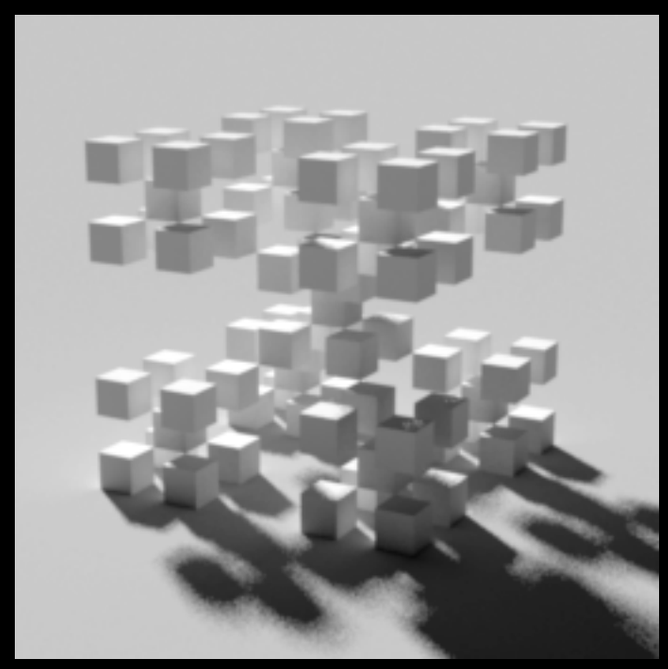
1x sampling, -ad 4



1x sampling, -ad 64



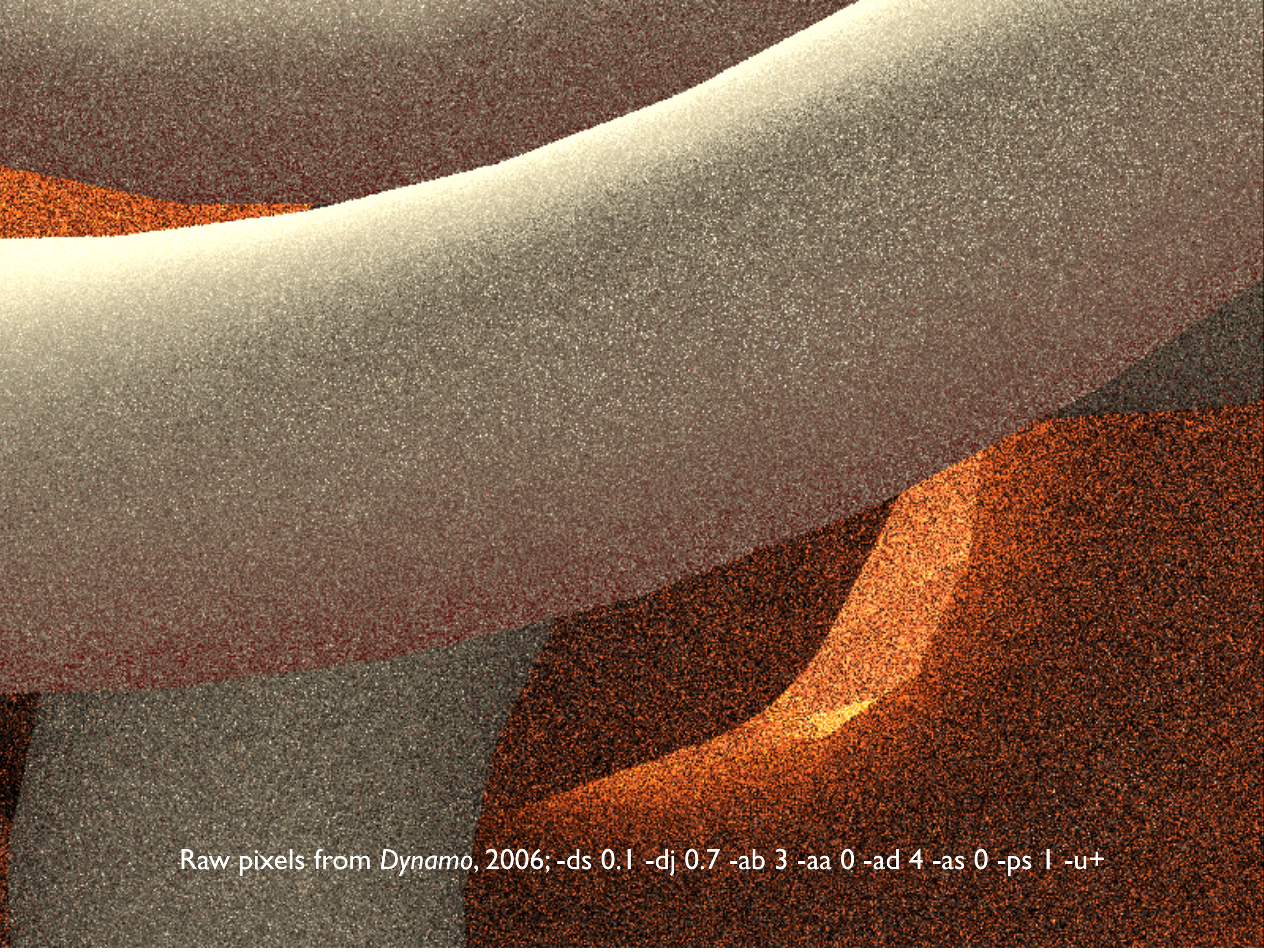
6x sampling, -ad 4



6x sampling, -ad 64

All: -aa 0 -ab 2 -as 0 -ps 1 -u+

Top row: 2 seconds, 3 min 47 sec
Bottom row: 1 min 20 sec, 2 hrs 16 min



These are pixels from a 24k rendering

Even with area sources and no sunlight, lots of noise

Always use -u+

If sunlight, must replace with many small suns to make penumbras less spotty

Even with 4x oversampling, sunlight shadows spotty

Have to oversample a lot, and render big

Why use caching?

- Making many renderings of same scene
- Simpler geometry, well-built
- Need lots of ambient bounces
- Need smooth results
- Have lots of RAM

The ambient pyramid

- Use “-af ambfile” option
- “pre-render” at 32x32
- render again at 512x512
- master render at >4k x 4k

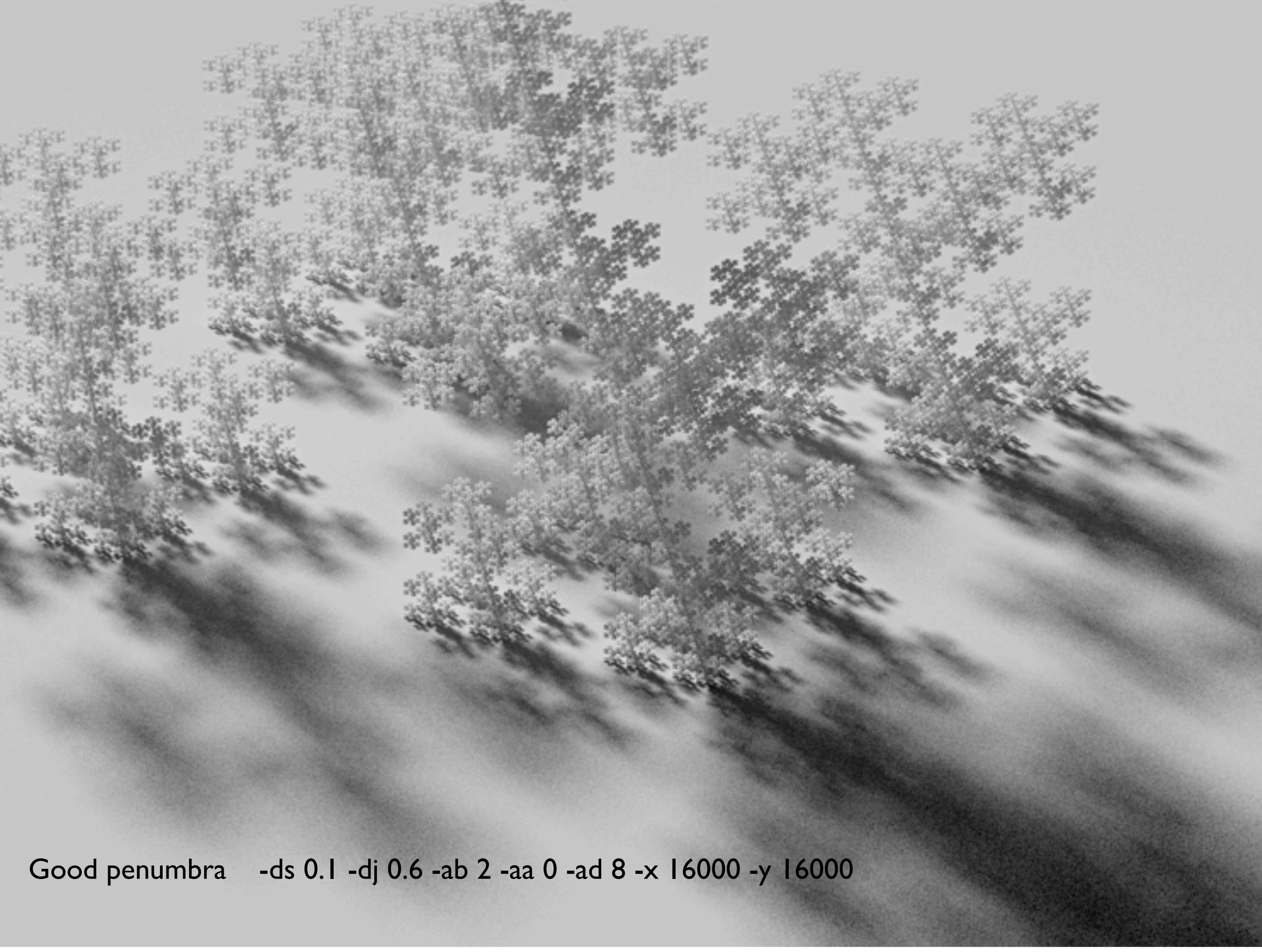
If you are performing one large render and want to save time, render the scene in stages.

Penumbras

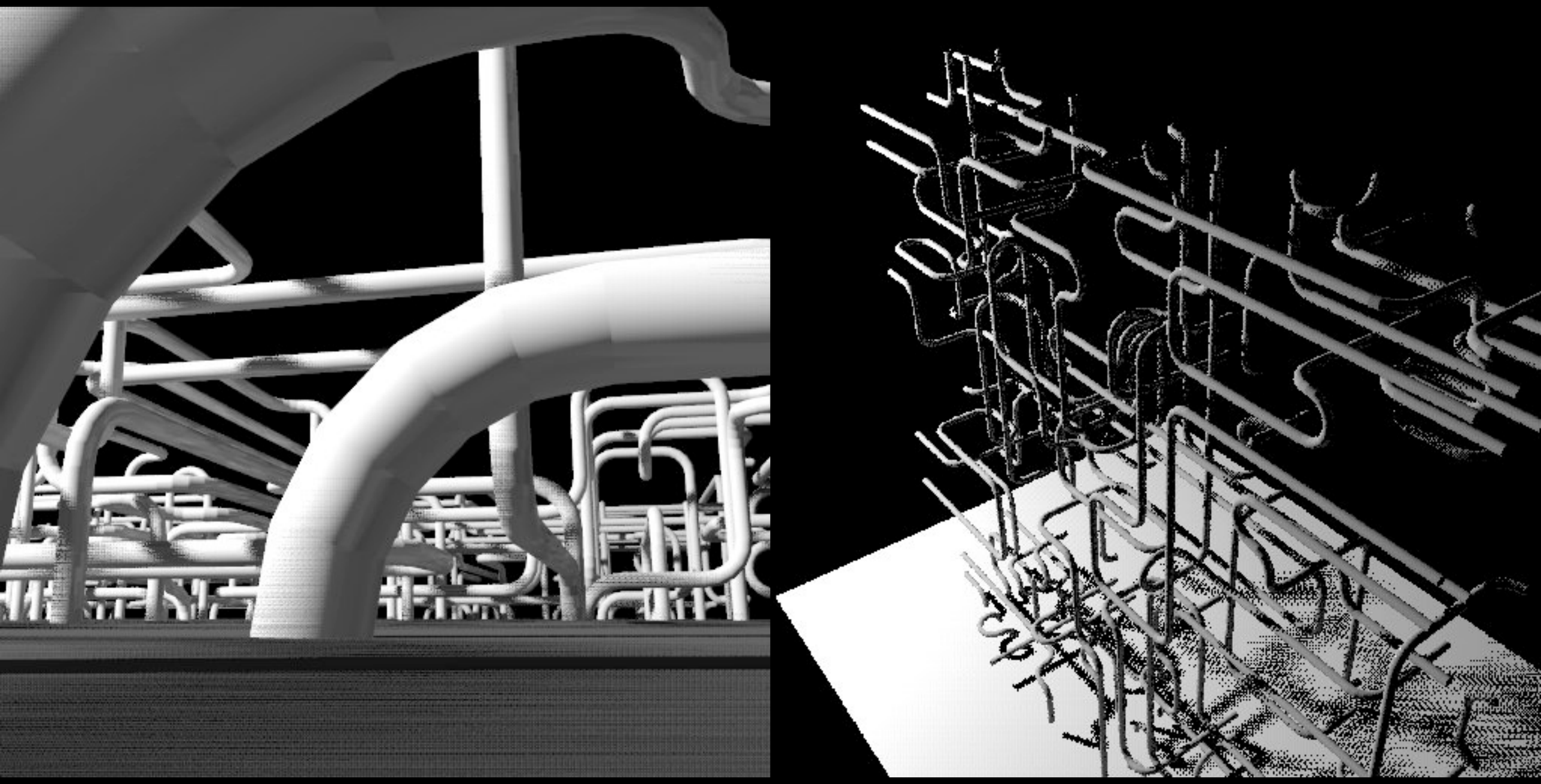
We’re already seen that penumbras happen when we use area light sources.

We know that they are spotty unless we oversample (render at multiple of final res)

But there’s another rpict trick to make them look better: monte-carlo sampling -u+



Good penumbra -ds 0.1 -dj 0.6 -ab 2 -aa 0 -ad 8 -x 16000 -y 16000



Rendering with bad penumbras, fix it with “-u+”

Bad penumbras.
Fix with oversampling, or by turning on Monte-Carlo sampling

These are specular reflections off a “metal” object.
Not a penumbra, but another problem that -u+ will solve.

This is caused by even sampling of the light source.
Two solutions to this:
Increase direct jitter to something like “-dj 0.9”
Manually subdivide your light source into

Depth-of-field blur

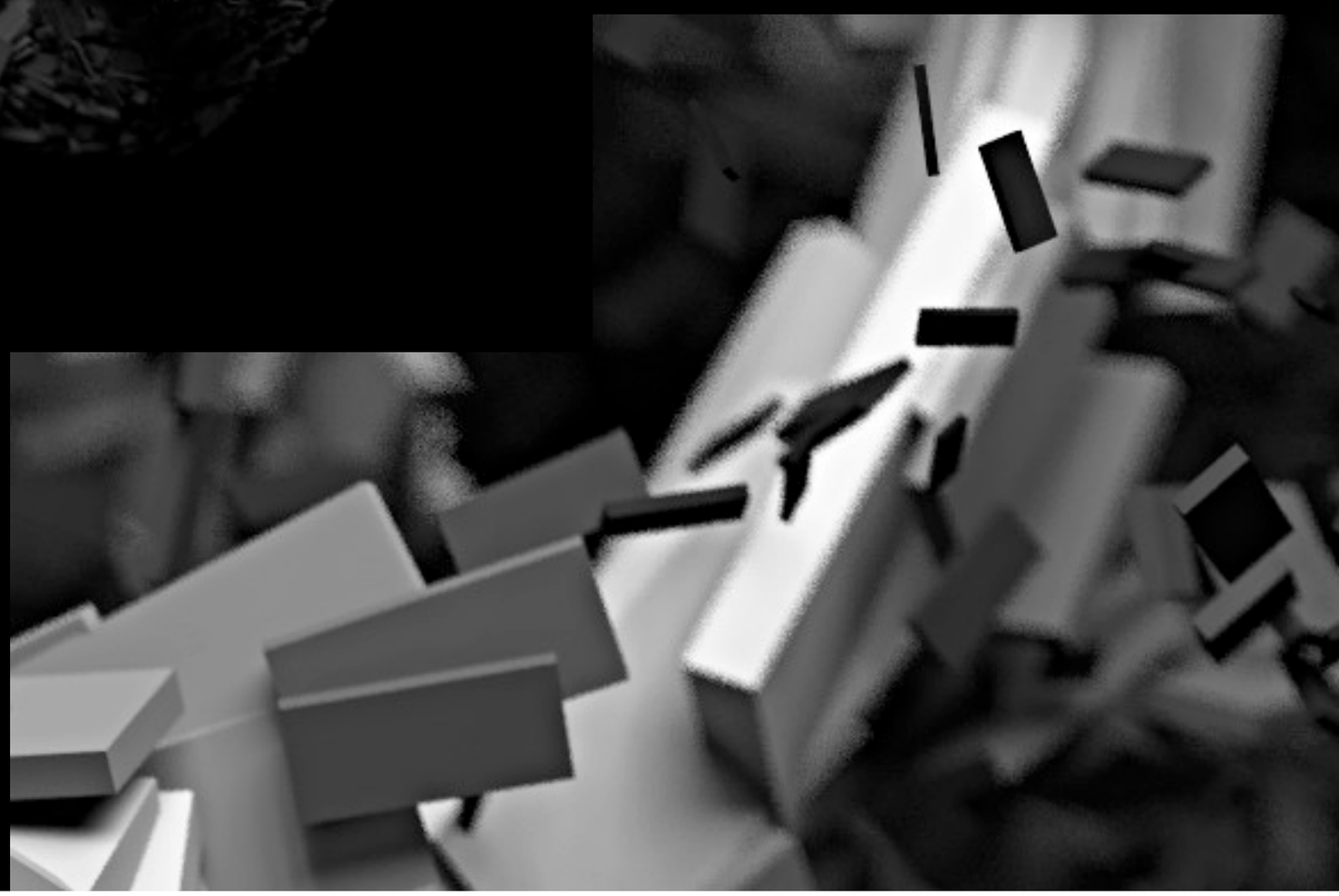
Tangled Pieces, 2010

-vtv -ab 3 -aa 0 -ad 32 -as 0 -ps 1 -u+

-dv- -dj 0.6 -ds 0.1 -dt 0.01 -lw 1.e-3

-pd 0.1

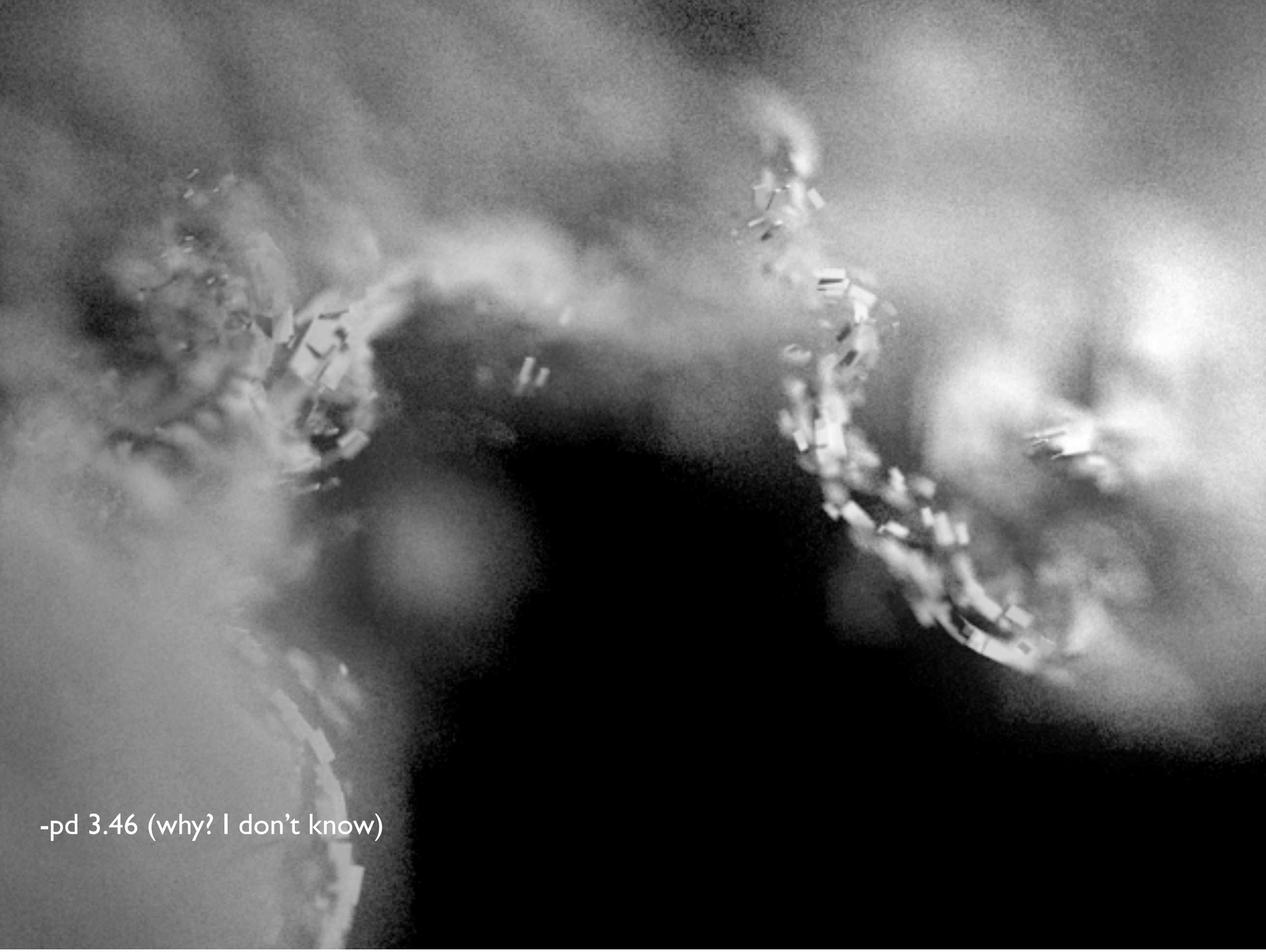
-x 20000 -y 16000



View direction vector also gives focus distance!

Note -pd 0.1, that’s the aperture

The -vd length here is 2.27, so this is like f/22 ?

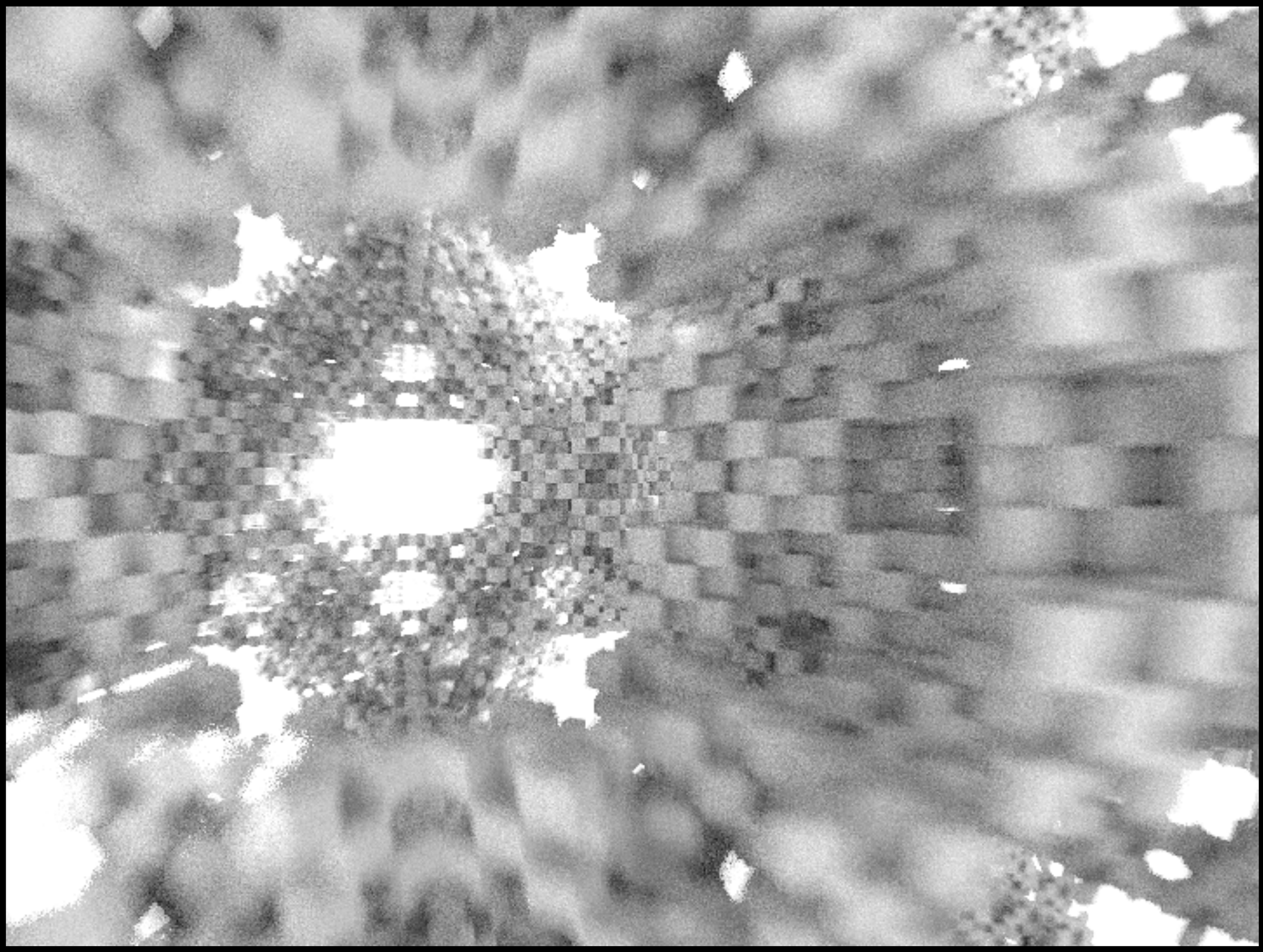


Distance (-vd) was also 3.46, so this is like a macro shot at f/1.0

Motion blur

rpict supports motion blur for animated sequences, but I’ve never actually used it

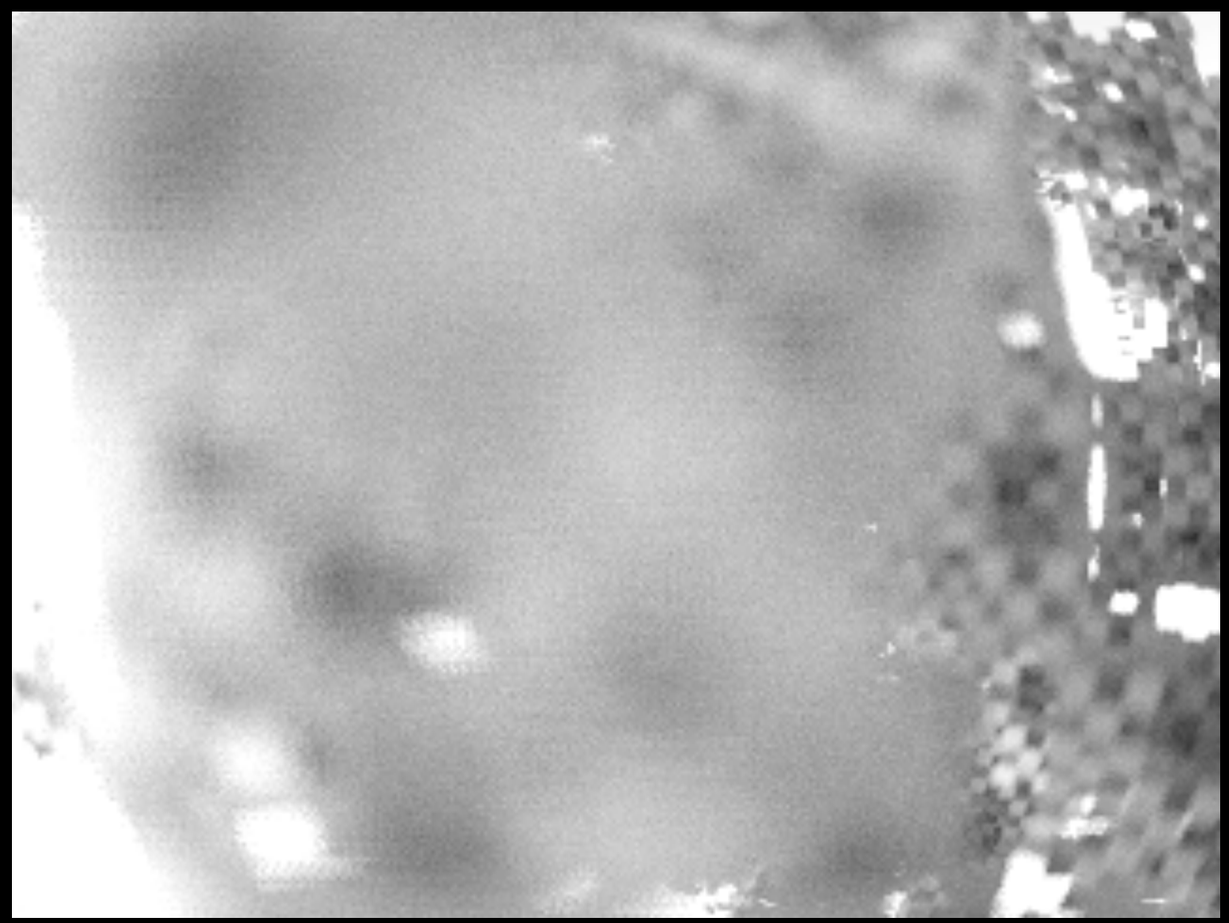
I wrote my own...



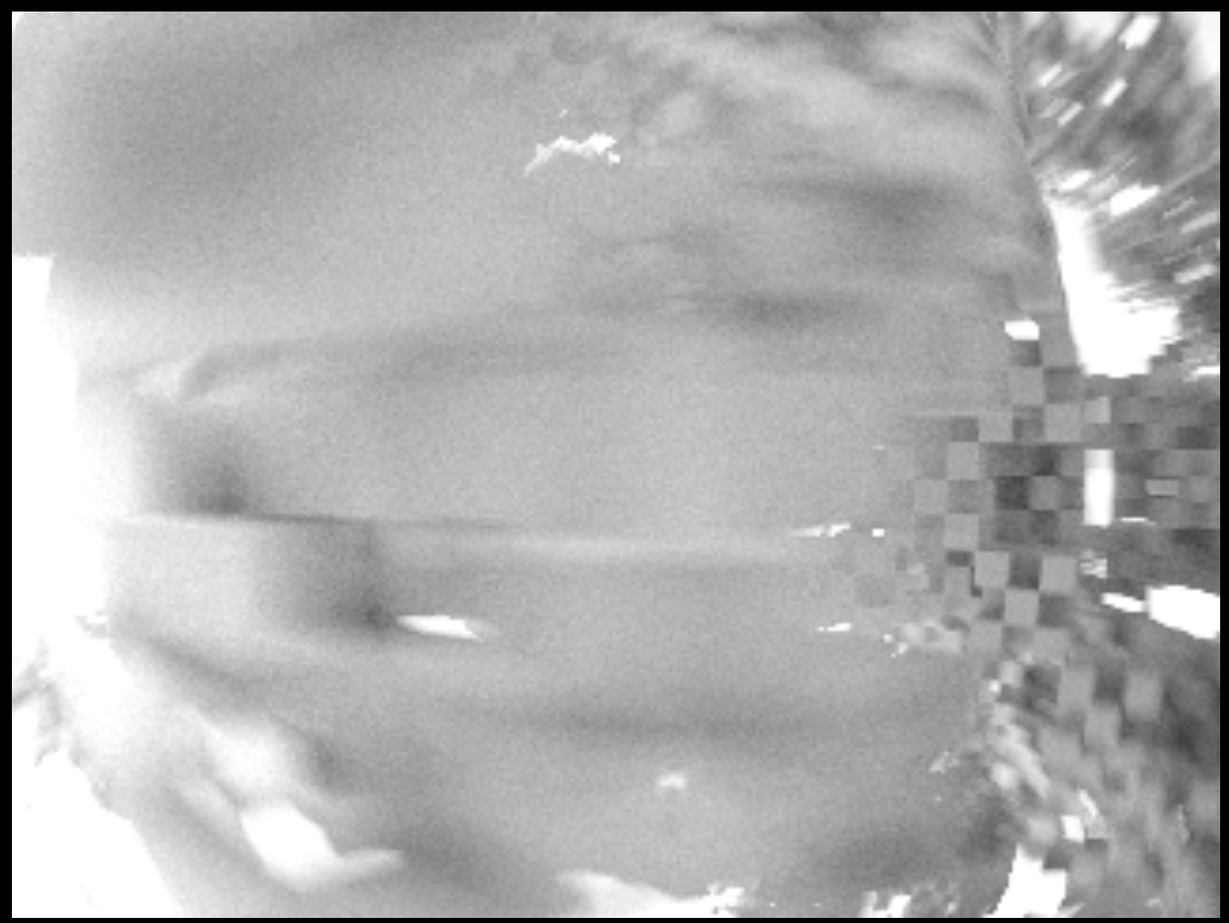
Motion blur sample; *not* using -pm

It works by rendering a number of low-quality images per final output image

then merging them together with pcomb



depth-of-field blur of one frame



motion blur of same frame

So, you can use the pmblur command, -pm in rpict, or roll your own motion-blur script.

I’d recommend doing it the easy way: using rpict.

Post-processing

Post-processing

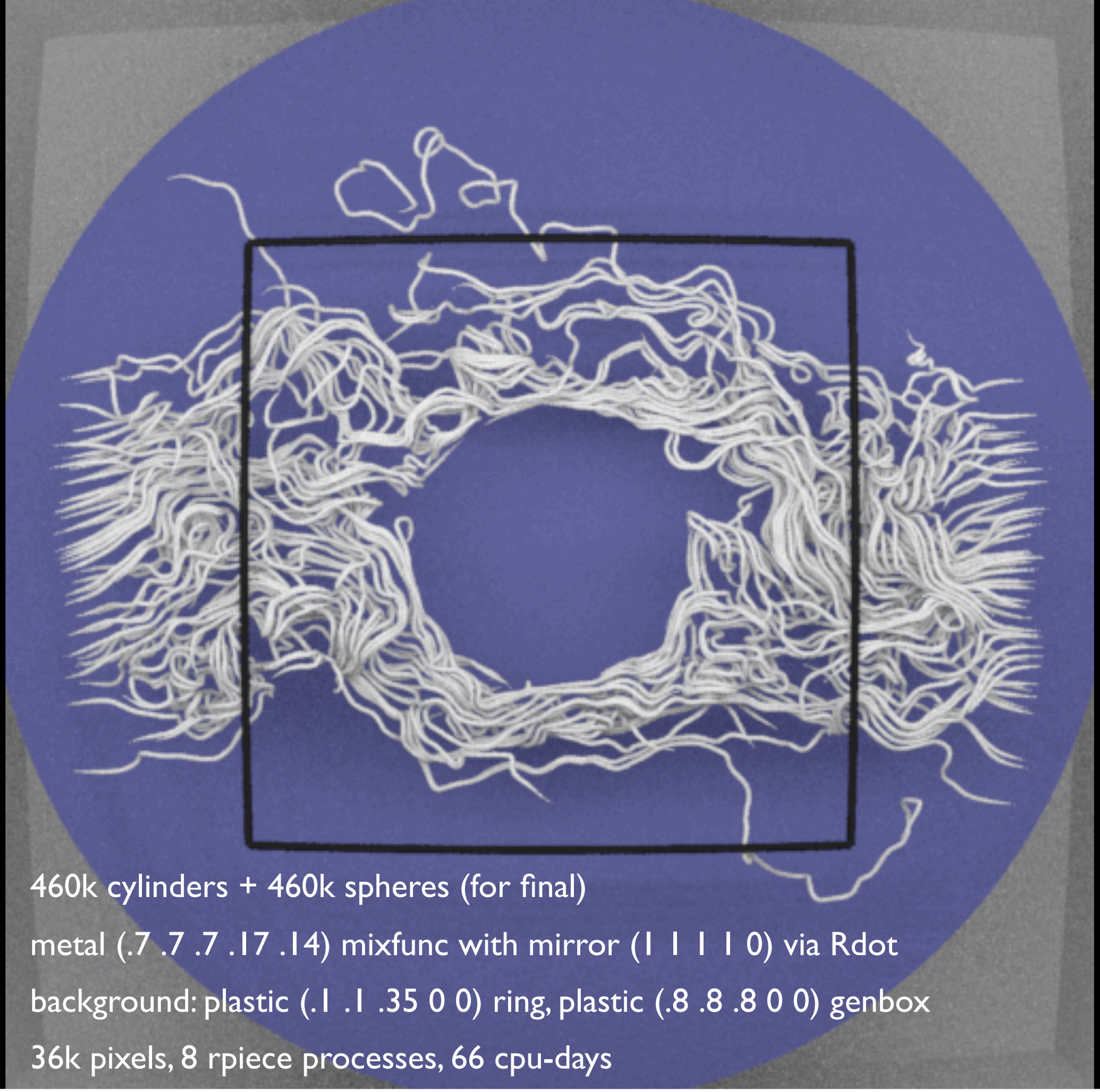
- Start with “pfilt -1 -e” to get exposure
- Drop to 8-bit with “ra_ppm | pnmtopng”
- Resizing in LDR is often cleaner
- Do any clean-up in Photoshop/Gimp

Oversampling

Oversampling test: 1x, 2x, 4x

pfilt -r 0.75 for 2x and 4x

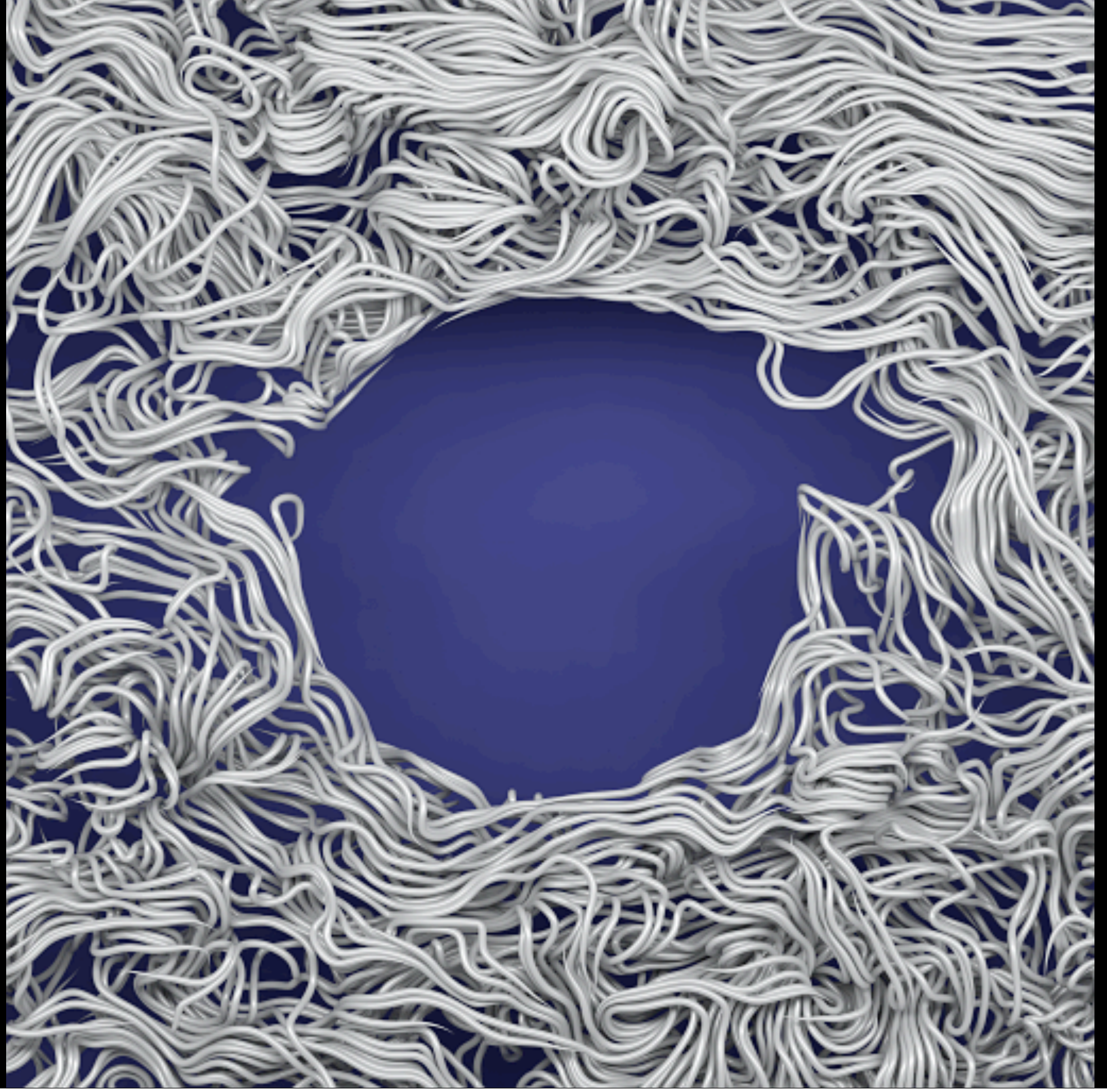
Putting it all together



We can trace streamlines through the vector field

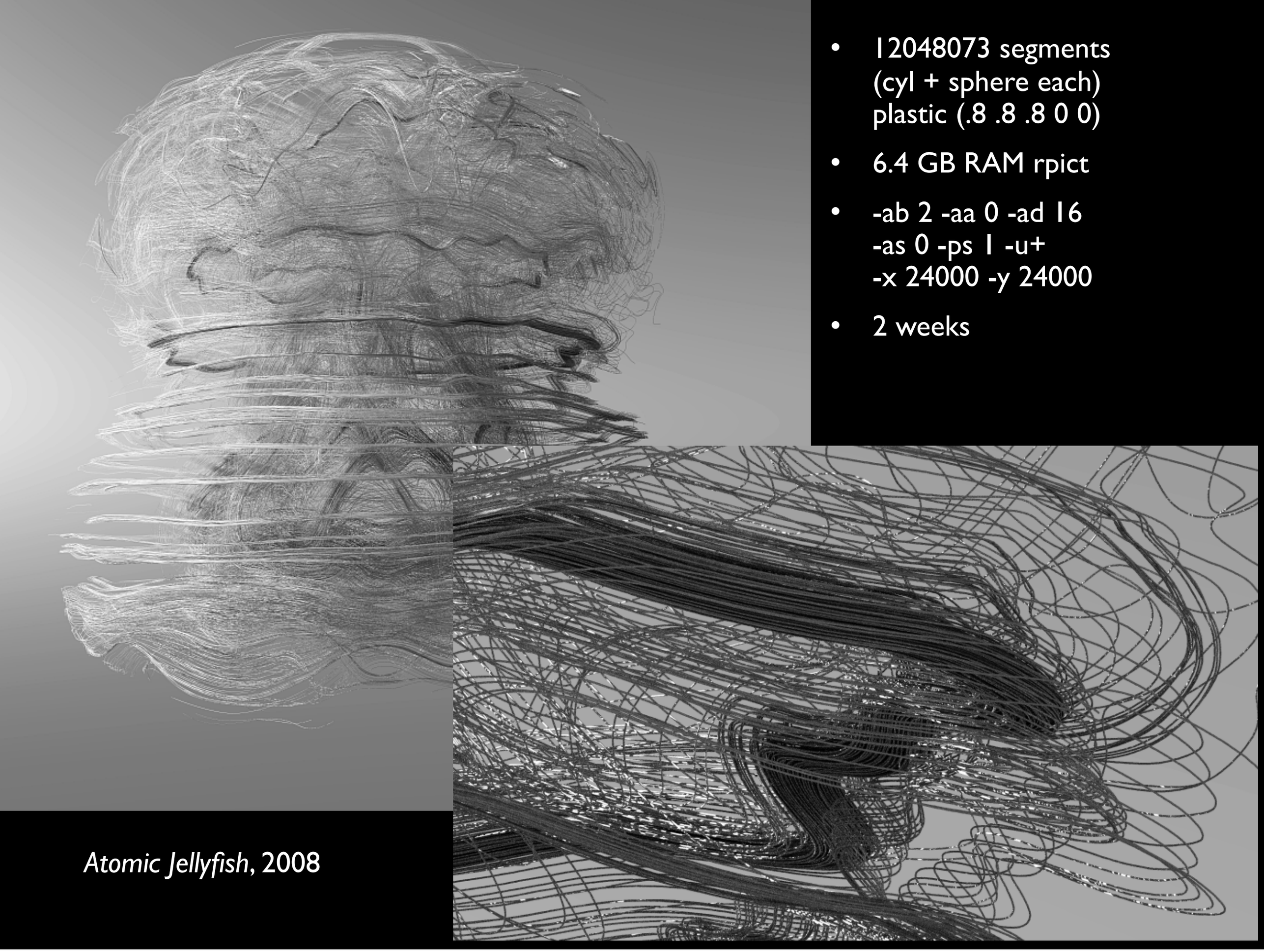
This is a test image that I made in the production of...

Extruded Simplices B
2005



This piece, which I call “Extruded Simplices B”

Just streamlines drawn through vortex field and confined inside a regular volume



- 12048073 segments
  (cyl + sphere each)
  plastic (.8 .8 .8 0 0)
- 6.4 GB RAM rpict
- -ab 2 -aa 0 -ad 16
  -as 0 -ps 1 -u+
  -x 24000 -y 24000
- 2 weeks

Atomic Jellyfish, 2008

Dumping space curve data to Radiance.

Note that there are 24 million primitives in this pic

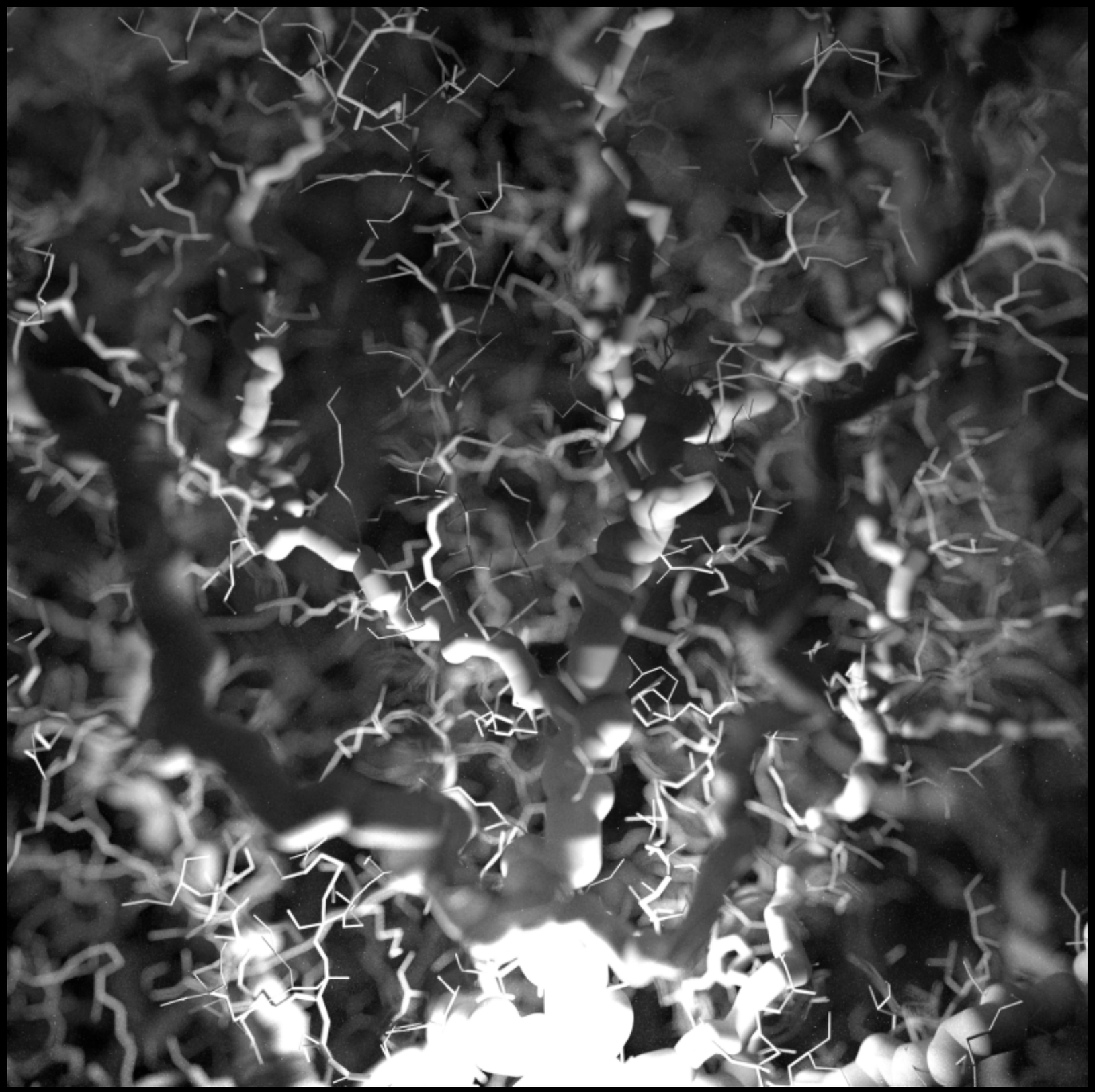
Twigs #23, 2004

464k cylinders,
464k spheres,
plastic (.5 .5 .5)

One ring light

Old-school depth-of-field blur

28.8k px, 4 days

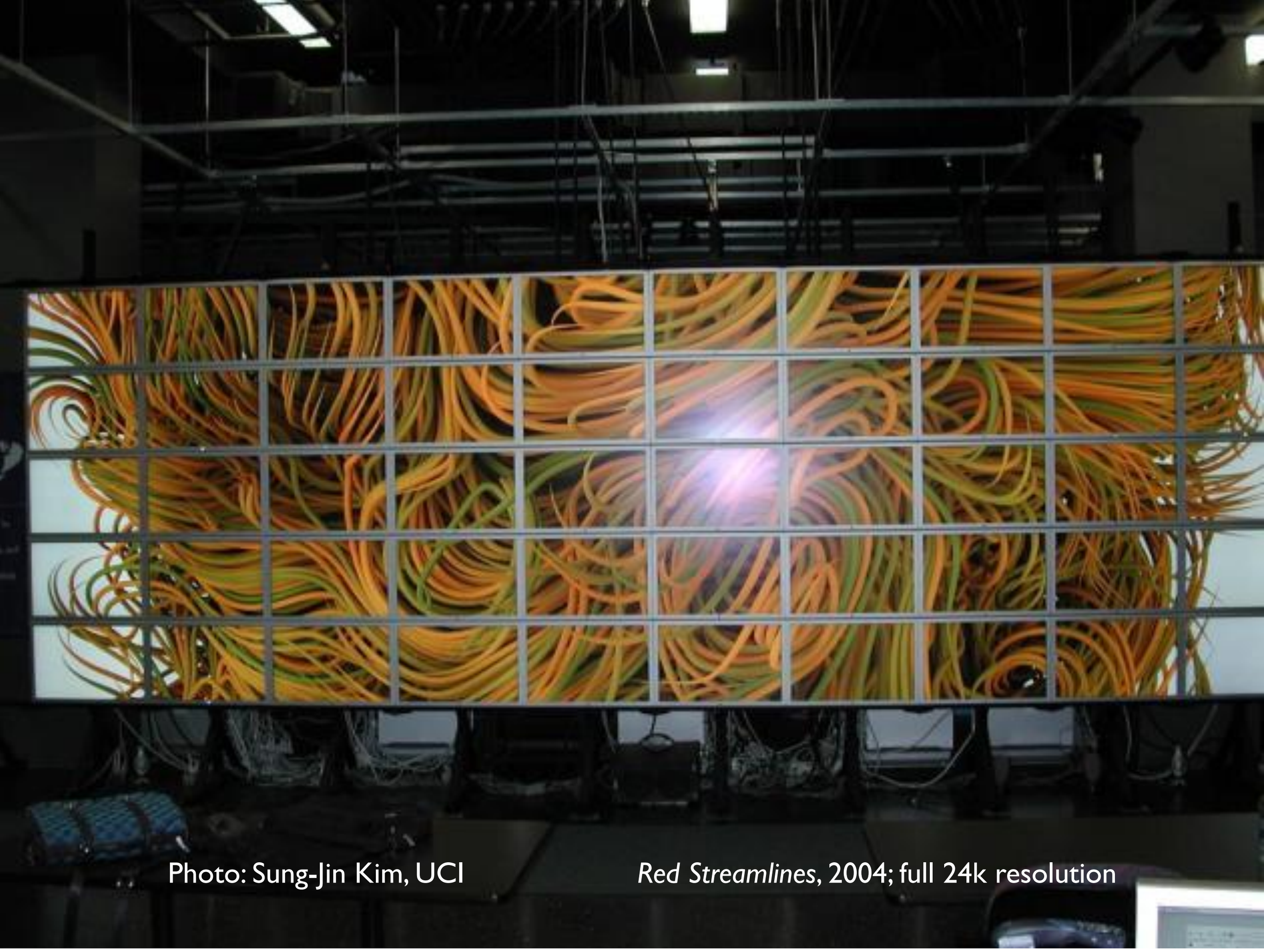


- vwrays -fd -vf vf2 -x 28800 -y 28800 | rcalc -id6 -e 'a:5.0;d:100.0' -e `vwright i < vf2` -e 'theta=2*PI*rand(2*recno-1);r=0.5*a*sqrt(rand(2*recno))' -e 'r1=r*cos(theta);r2=r*sin(theta)' -e 'dx=r1*ihx+r2*ivx;dy=r1*ihy+r2*ivy;dz=r1*ihz+r2*ivz' -e '$1=ipx+dx;$2=ipy+dy;$3=ipz+dz' -e '$4=$4-dx/d;$5=$5-dy/d;$6=$6-dz/d' -od | rtrace -fdc -x 28800 -y 28800 -ab 2 -aa 0 -ad 4 -as 0 -dj 0.7 image11.oct > img31.pic

This piece took only 4 days to render at 29k

I did not think of that command line on my own. Thanks Greg!

This was before pdfblur or -pd



HiPerWall at U-California Irvine

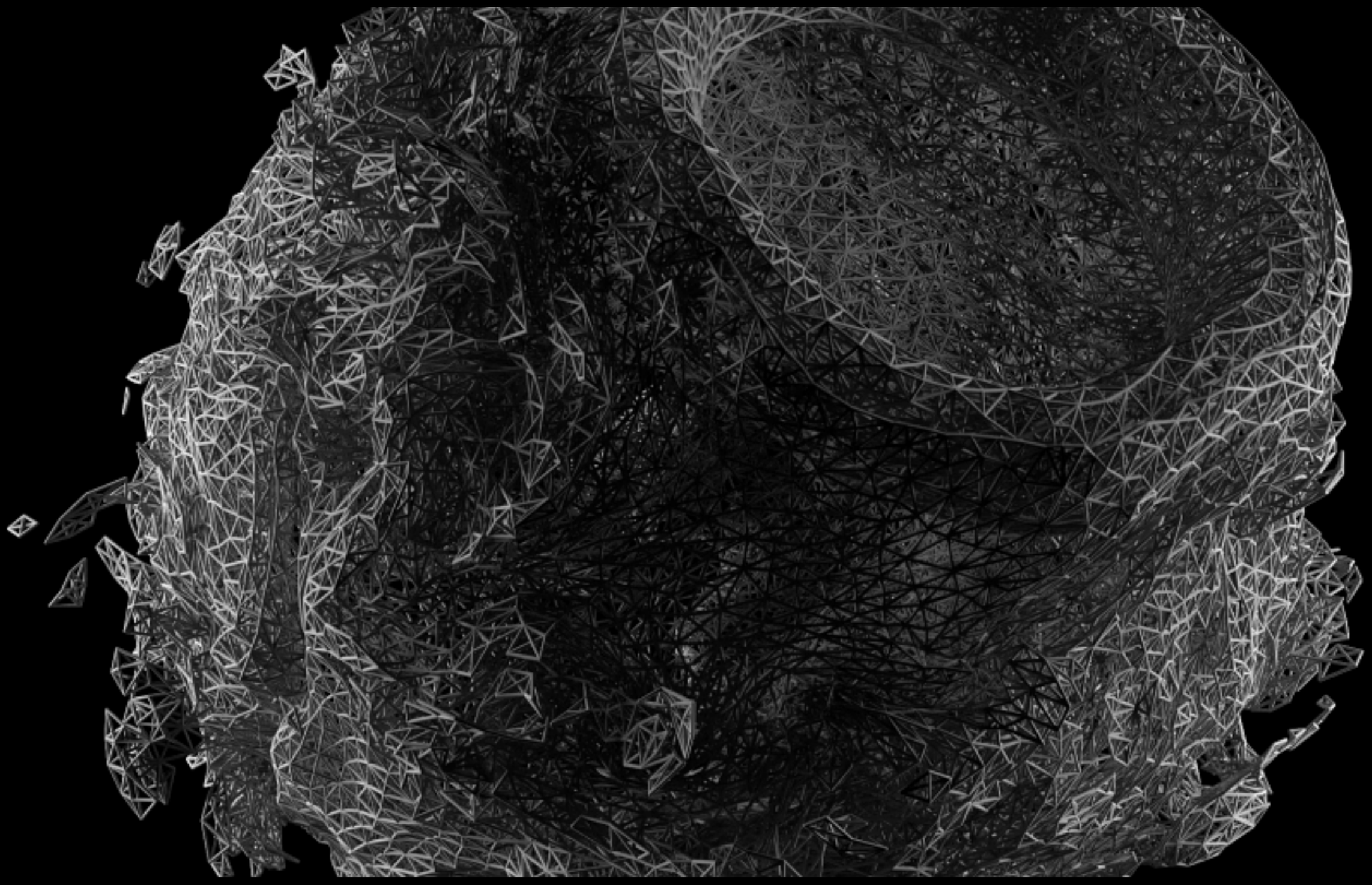
Project run by Prof. Stephen Jenks
50 apple monitors at 2560x1600 each

24k original
“People can't believe it is rendered – they think it looks so real.”

Perpetuity?, 2008 -ab 4 -aa 0 -ad 16 -as 8, 55k x 35k, 58B rays, 9 CPU days

Open original .pic in Cinepaint
Find best exposure (subject whites peak)
save
use gamma correction to sqrt the data
blur to get lightmap image
divide original by lightmap to get detail map
squash both to 8-bits
print

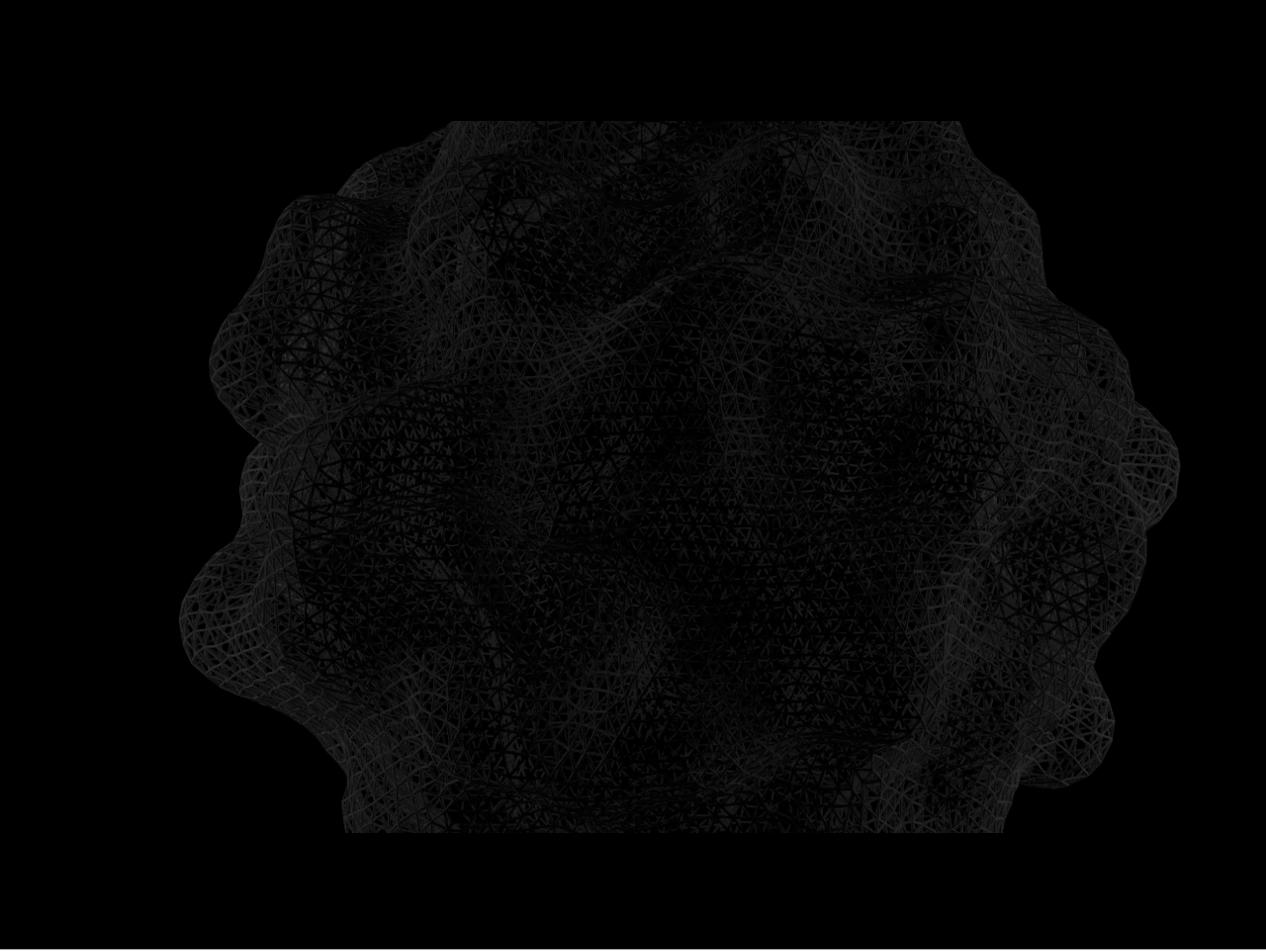
Smoke Water Fire, 2007



50k-700k cylinders (plastic .1 .1 .1 0 0), 2 cylinder lights, motion blur via frame averaging

7250 frames, -ab 2 -aa 0 -ad 16 -as 0 -ps 1 -dj 0.7 -ds 0.2 -u+ -x 3840 -y 2160

Still image from *Smoke Water Fire*, at SIGGRAPH 2008, ASPECT 19



This is the first major animation that I made with Radiance, in 2007/8

7300 frames at 3840x2160

Took 2 months to render

Thank you

Mark J. Stock
mstock@umich.edu
http://markjstock.com/

Harvard Graduate School of Design
Apr 7, 2011

So that's what I do for fun. Any questions?

My art is at http://markjstock.com/
Other fun stuff at http://markjstock.org/radiance/
Follow me on Twitter @markjstock
More at http://www.youtube.com/view_play_list?p=5226906B9F7500CA