DYAMOND Visualization

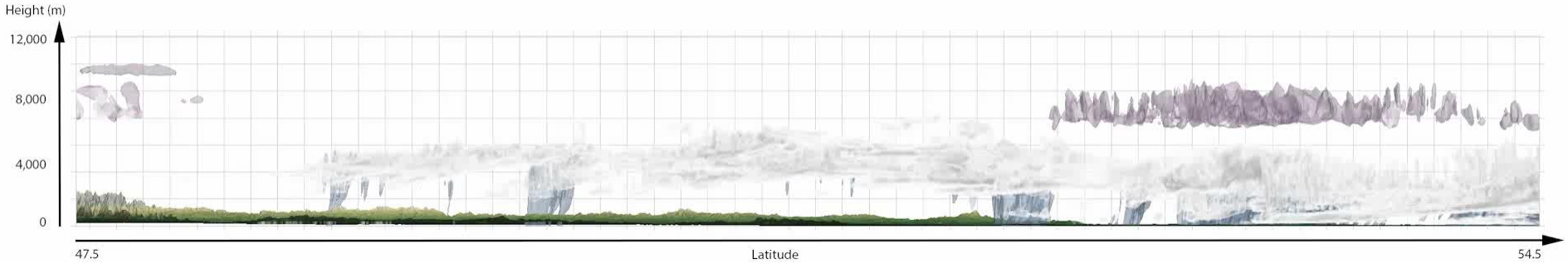
Requirements for High Resolution Climate Simulations

Niklas Röber
Deutsches Klimarechenzentrum GmbH (DKRZ)
Hamburg, Germany

Data Visualization

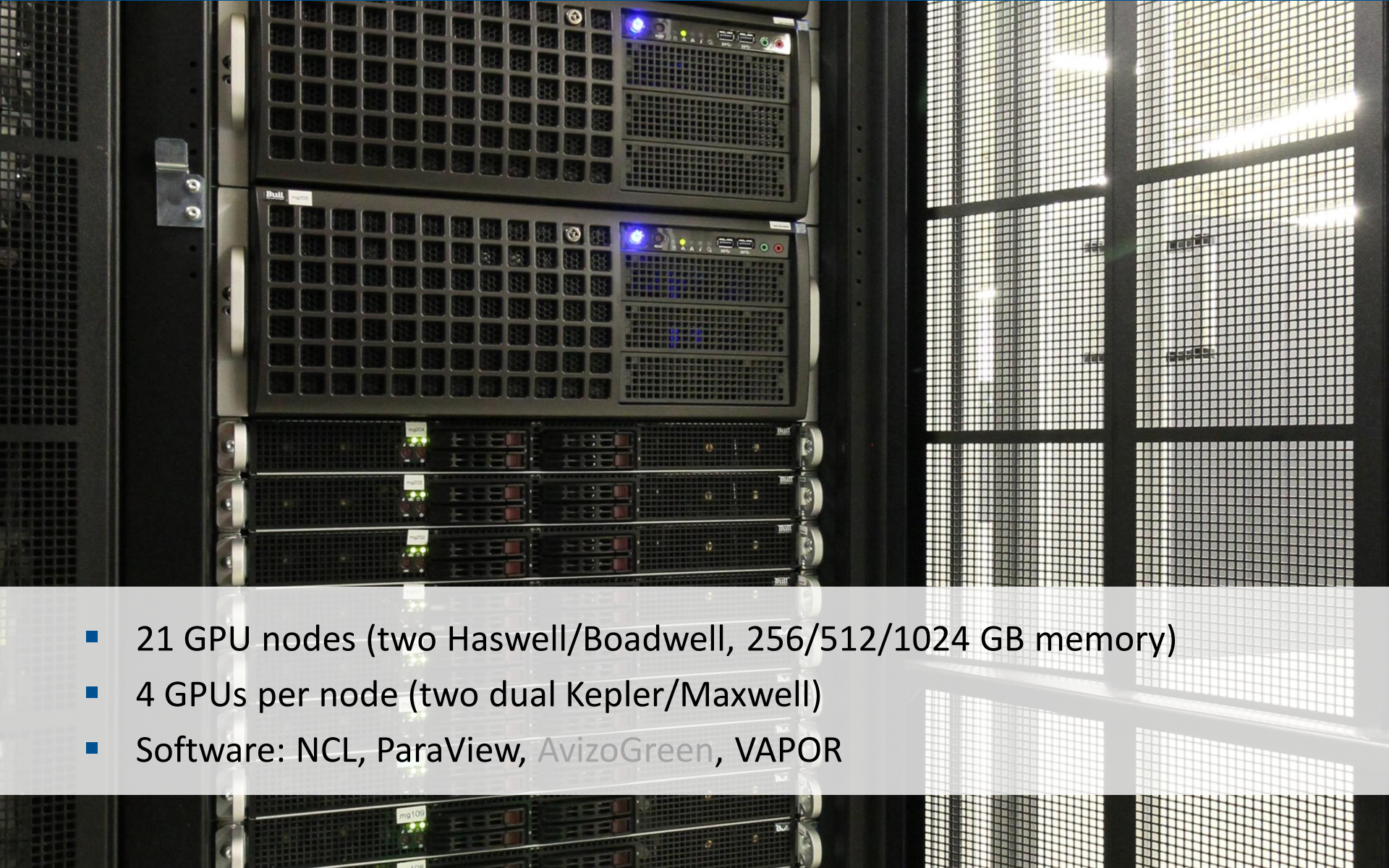
See, understand, learn, communicate …

- Confirmatory visualization (verification)
- Interactive visualization (exploration)
- Animations and images (communication)
- But also: model debugging and performance analysis

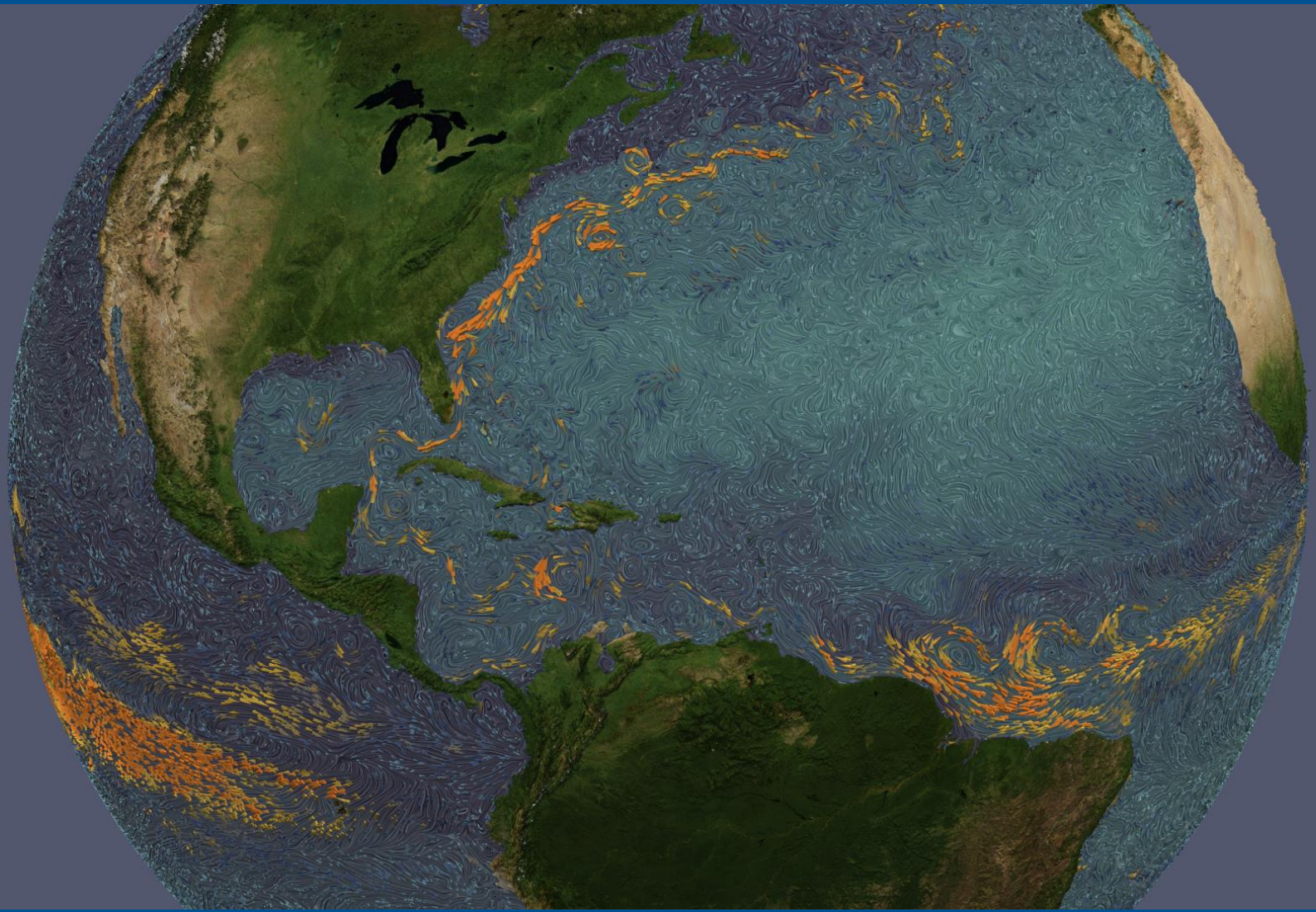


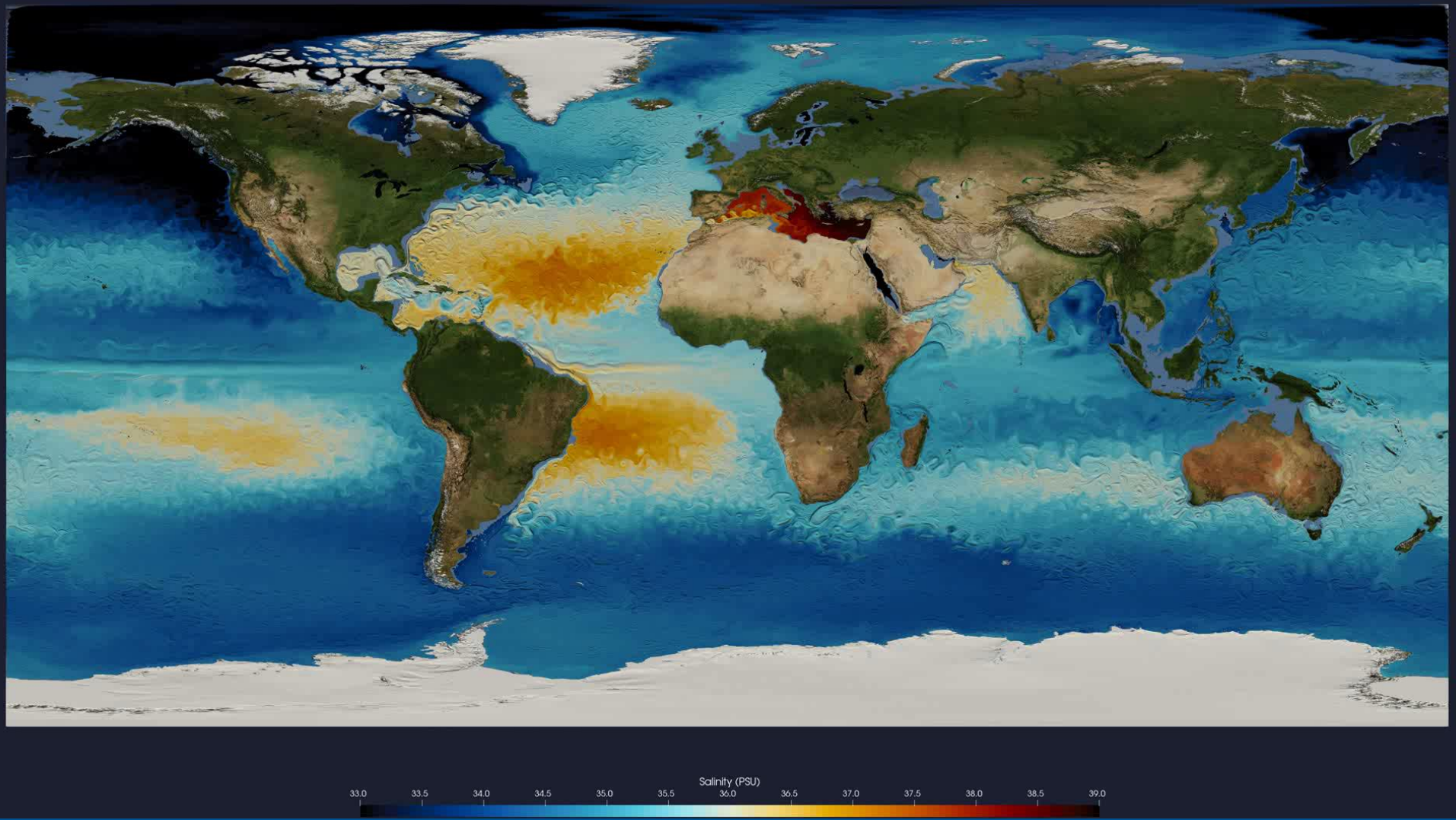
“Numbers have an important story to tell. They rely on you to give them a clear and convincing voice.” – Stephen Few

“Visualization gives you answers to questions you didn’t know you had.” – Ben Shneiderman



- 21 GPU nodes (two Haswell/Boadwell, 256/512/1024 GB memory)
- 4 GPUs per node (two dual Kepler/Maxwell)
- Software: NCL, ParaView, AvizoGreen, VAPOR


3.9e+01
38.5
38
37.5
37
36.5
36
35.5
35
34.5
3.4e+01
os
1.0e+00
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0e+00
mag


Salinity (PSU)
33.0
33.5
34.0
34.5
35.0
35.5
36.0
36.5
37.0
37.5
38.0
38.5
39.0

ICON Ocean Model

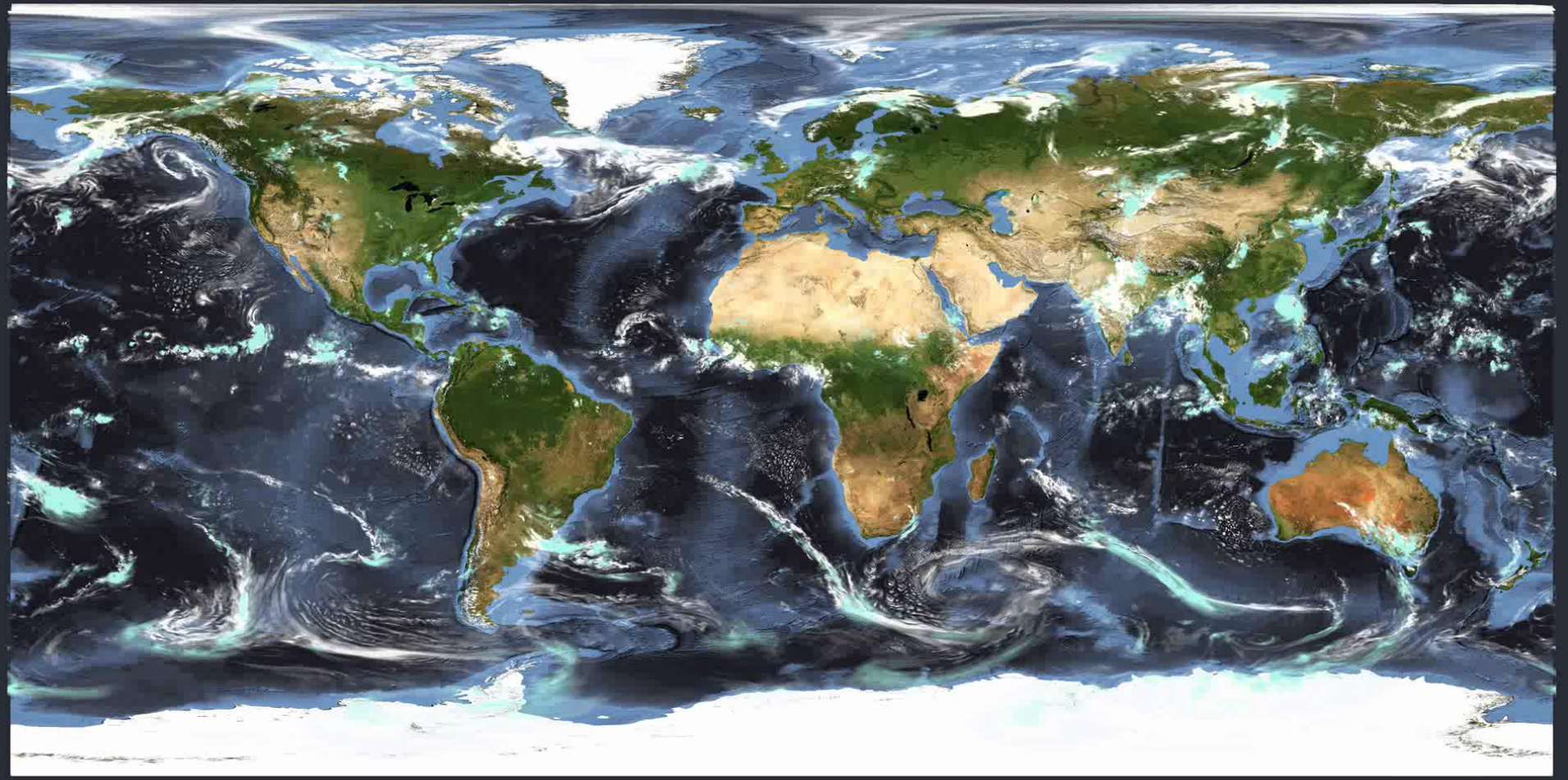
(5km/1h resolution @ 72m depth)

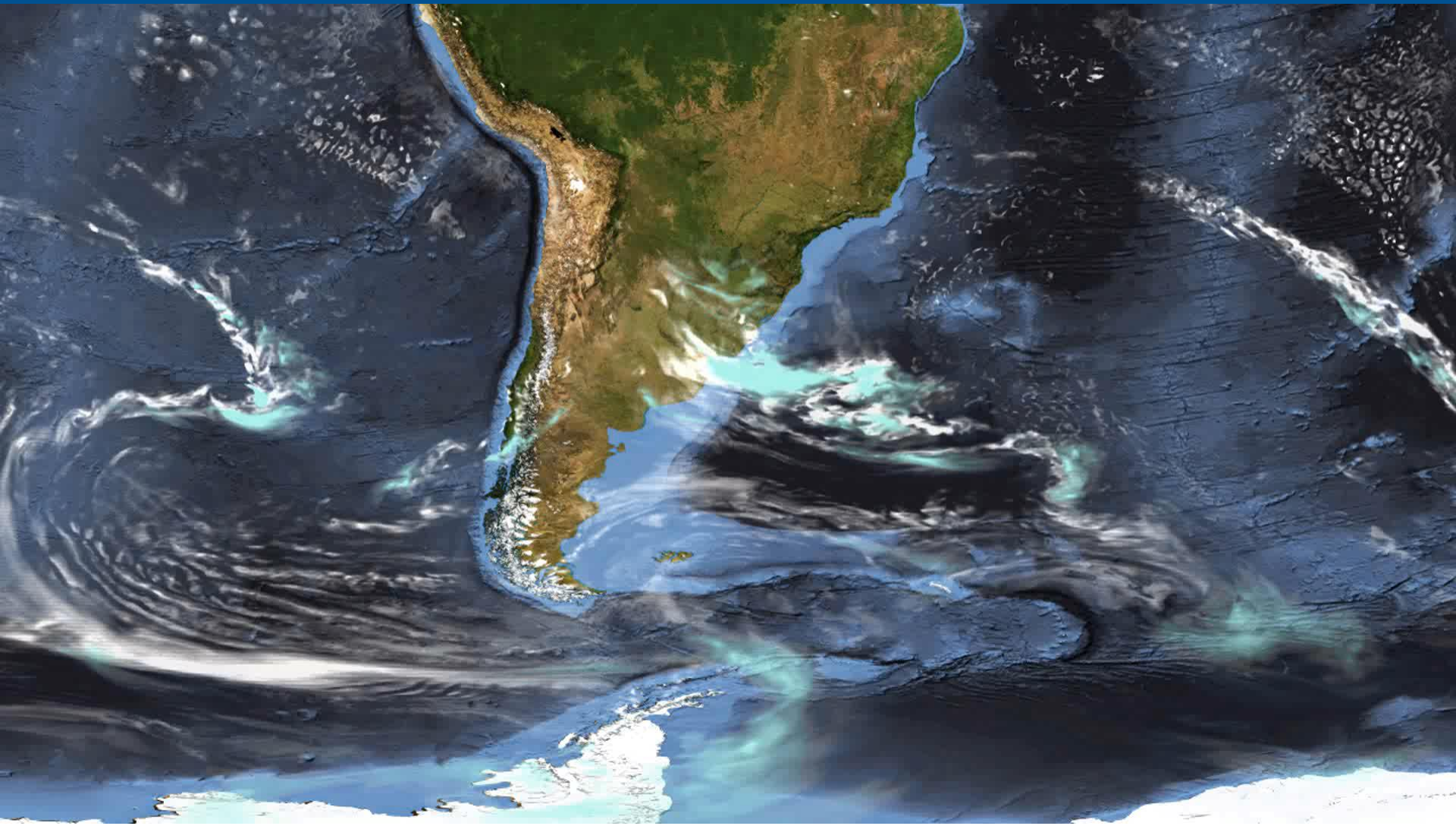
Data: Helmuth Haak, Peter Korn

Visualization: Niklas Röber

Max-Planck-Institut für Meteorologie

DKRZ DEUTSCHES KLIMARECHENZENTRUM

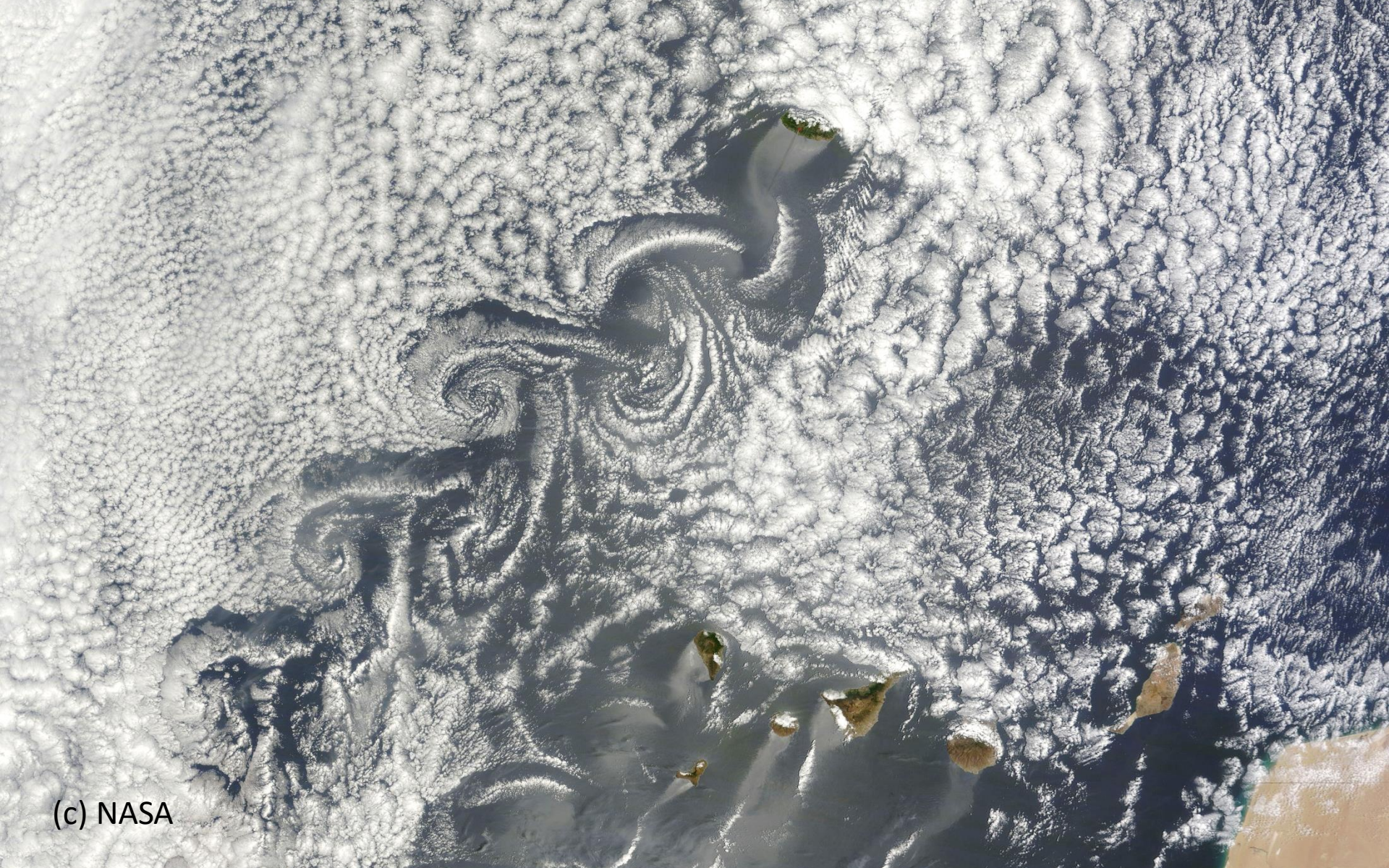

ICON DYAMOND R2B10 2.5km Resolution
01.08.2016 at 00:00
Vertically Integrated Cloud Water
0.005 0.01 0.02 0.05 0.1 0.2 0.5 1 2 5 10
Vertically Integrated Cloud Ice
0 0.5 1 1.5 2 2.5 3



Coupled Cloud Resolving Earth System Model - DYAMOND++

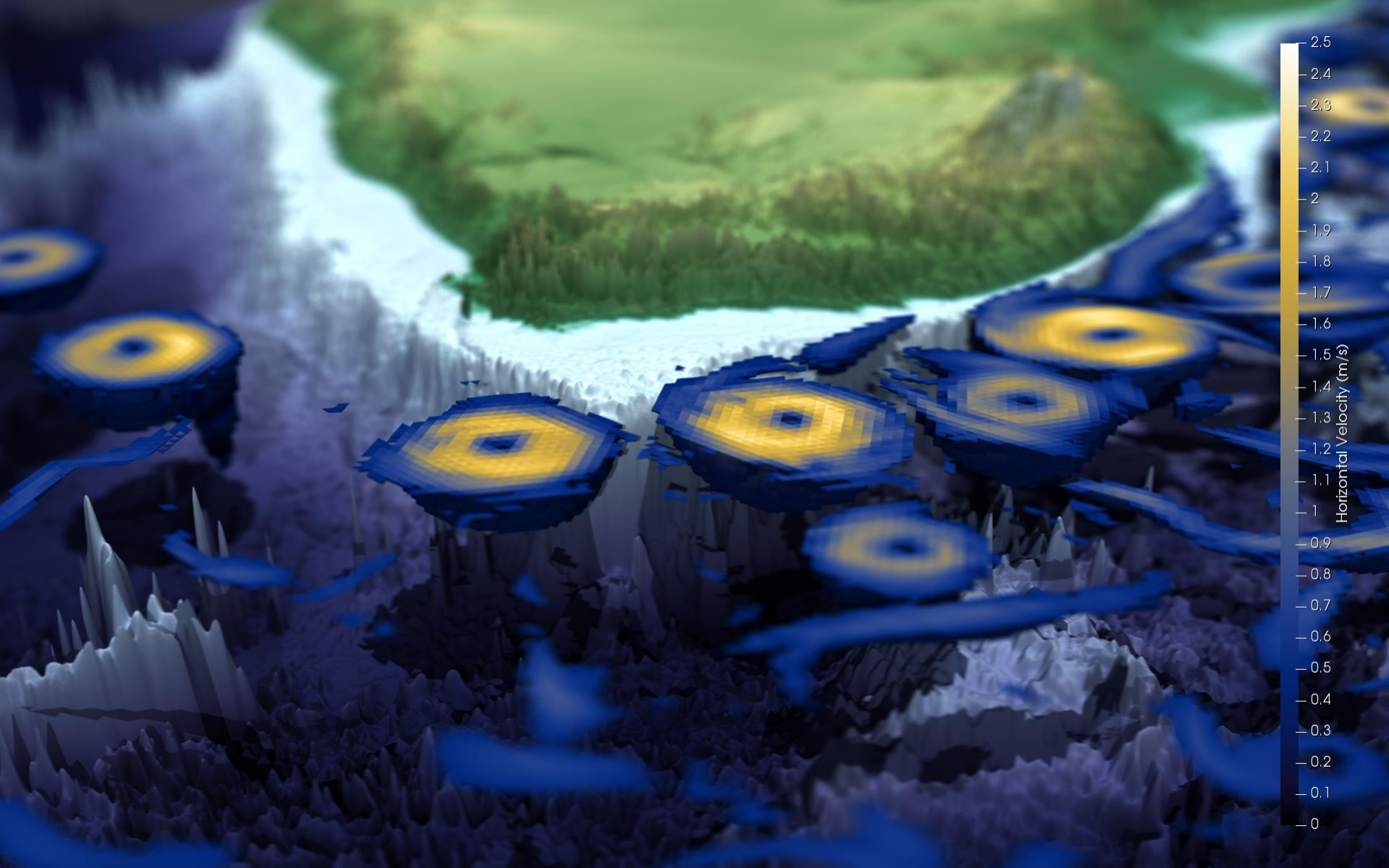
Horizontal Resolution Atmosphere & Ocean 5km, 15min Interval

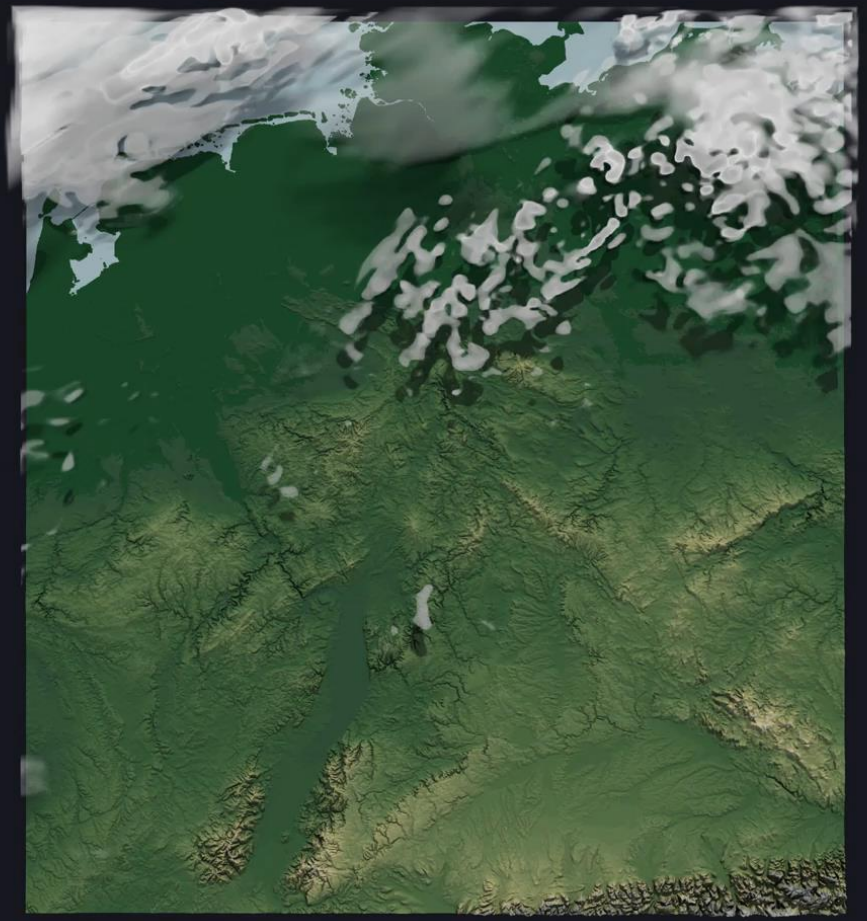
September 4, 2013


(c) NASA

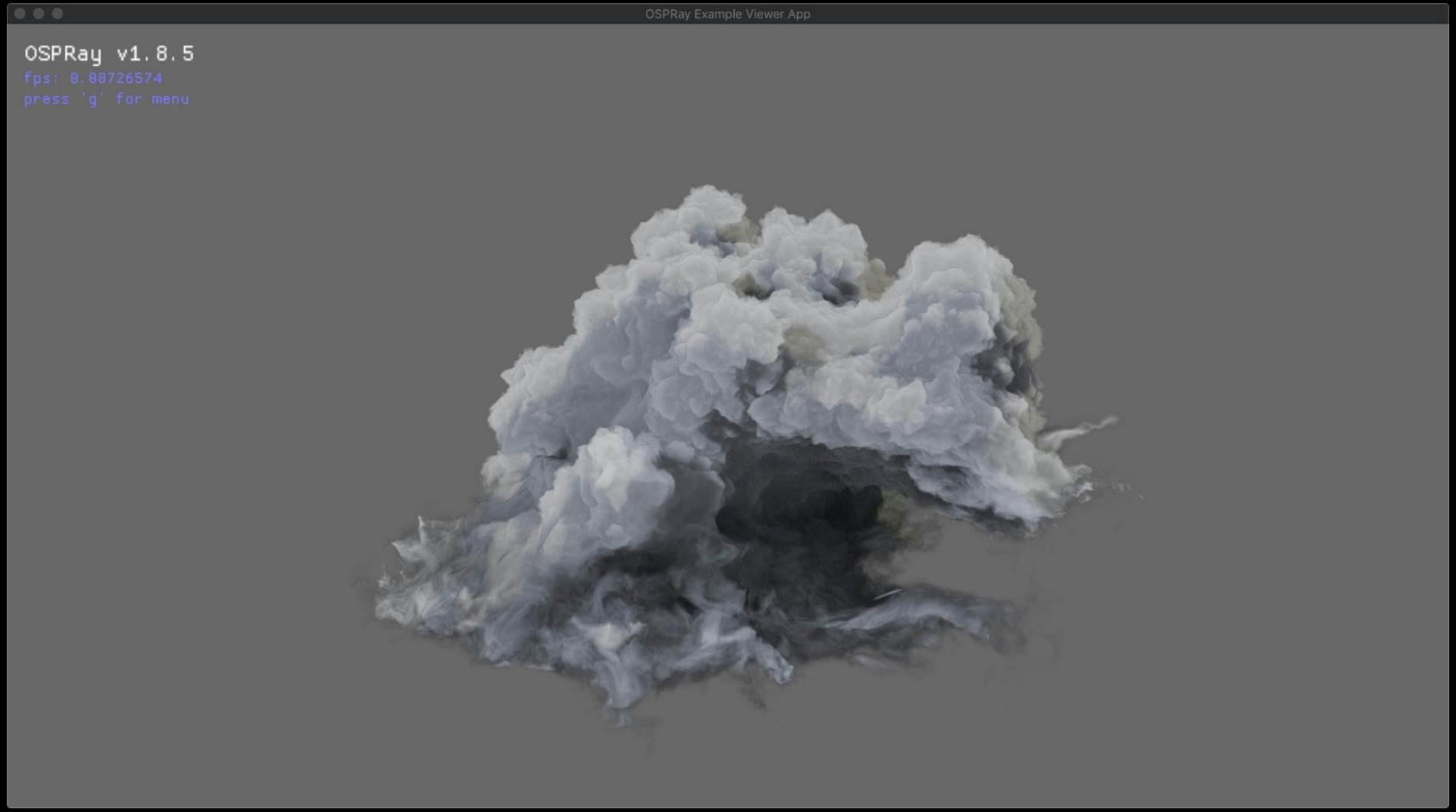
Canary Islands

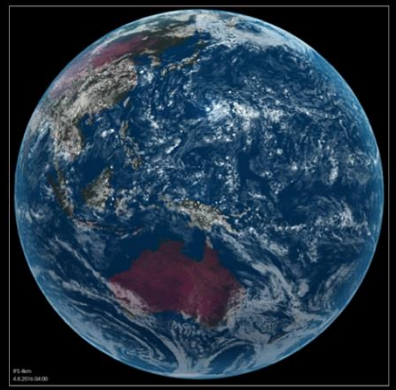
DYAMOND R2B10 - 2D Wind Visualization
(3 Minute Output - 10m Height)


Horizontal Velocity (m/s)
2.5
2.4
2.3
2.2
2.1
2
1.9
1.8
1.7
1.6
1.5
1.4
1.3
1.2
1.1
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0

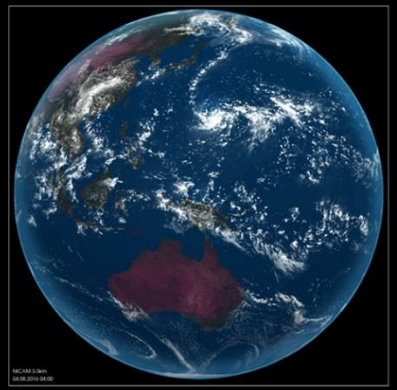


26.04.2013 00:00


OSPRay Example Viewer App
OSPRay v1.8.5
fps: 0.00726574
press 'g' for menu

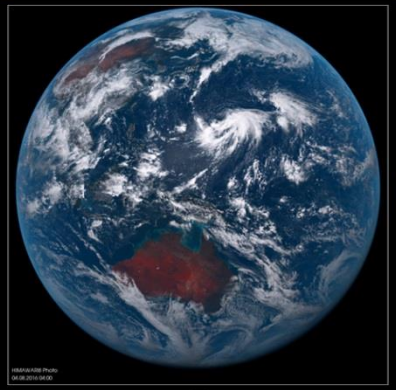

IFS 4km
4.8.2016 04:00


IFS 9km
4.8.2016 04:00

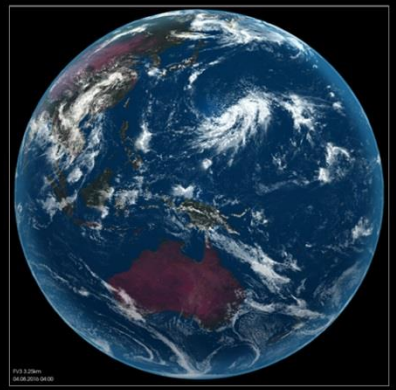

NICAM 3.5km
04.08.2016 04:00


UM 4km
04.08.2016 04:00

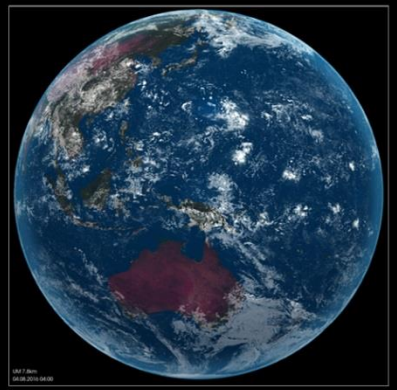

ARPEGE-NH 2.5km
04.08.2016 04:00

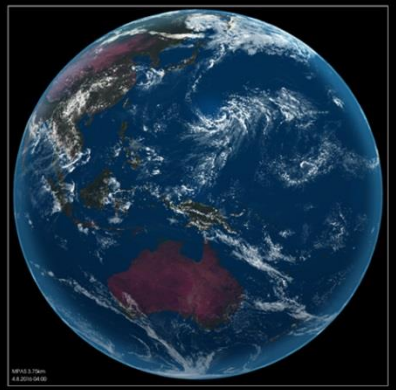

HIMAWARI8 Photo
04.08.2016 04:00

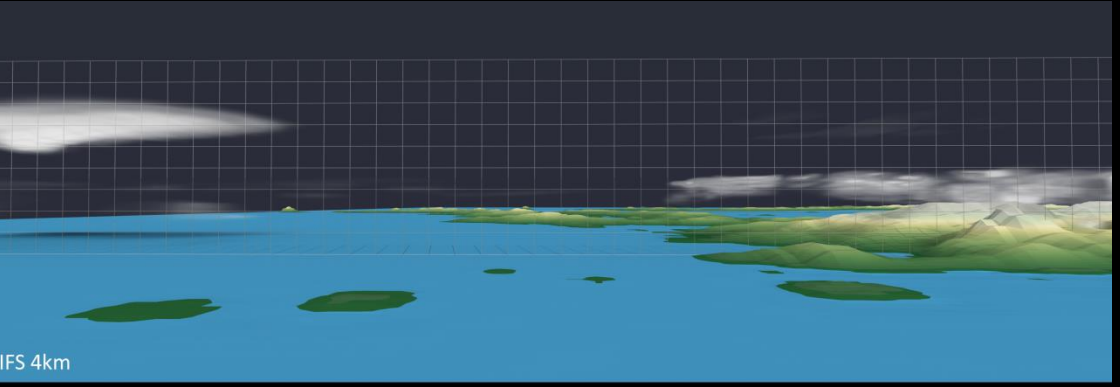

ICON 2.5km
04.08.2016 04:00

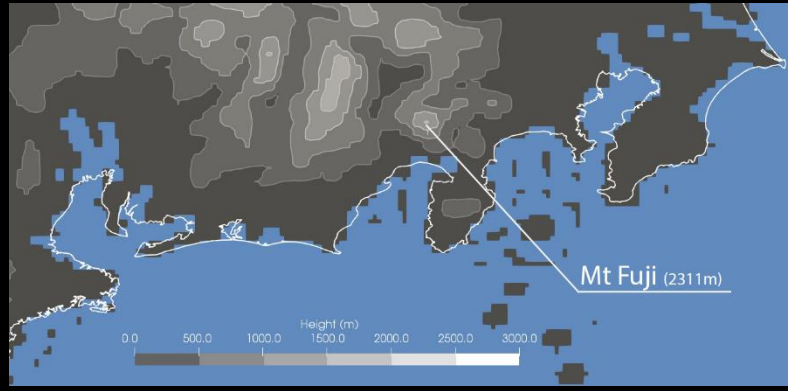

FV3 3.25km
04.08.2016 04:00

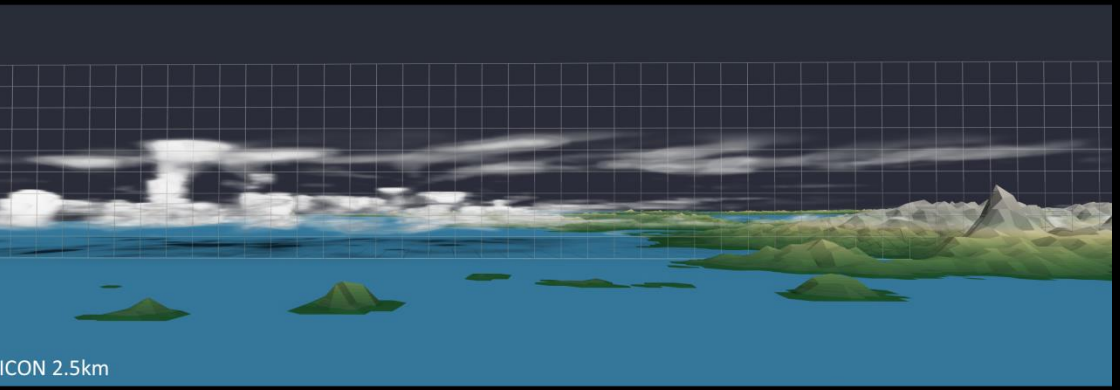

GEOS 3km
04.08.2016 04:00

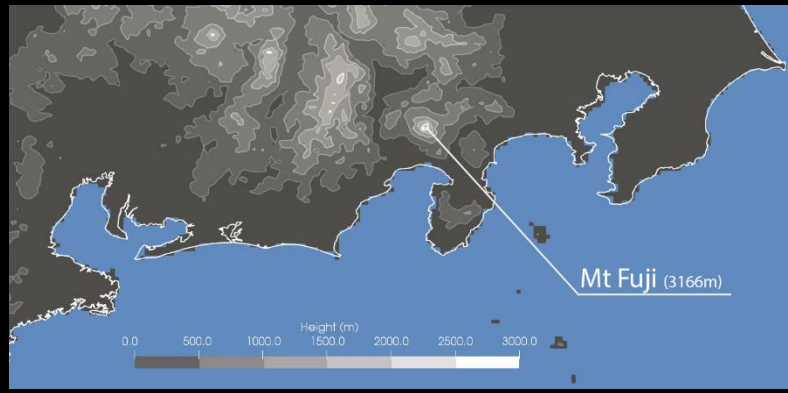

UM 7.8km
04.08.2016 04:00

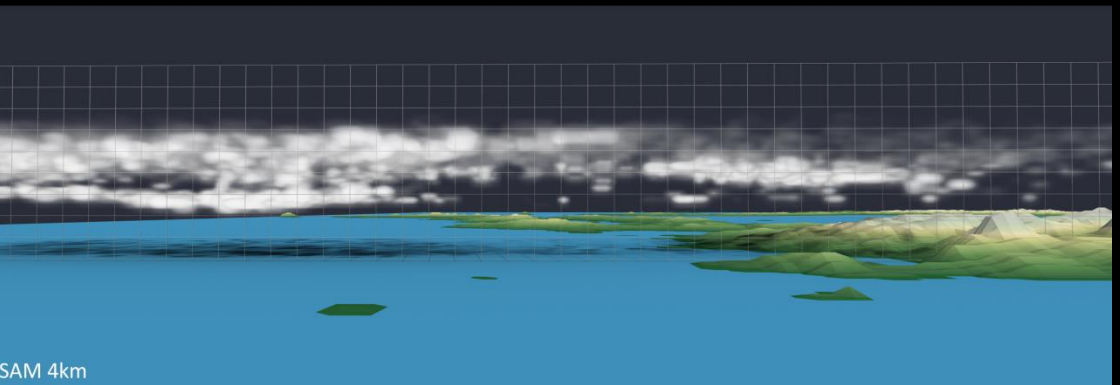

MPAS 3.75km
4.8.2016 04:00

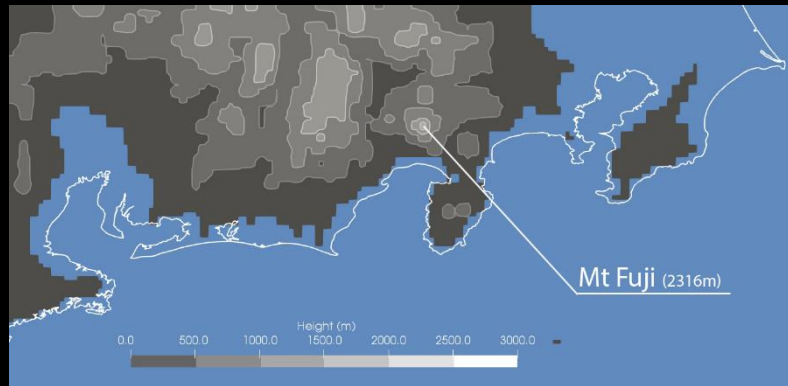

IFS 4km

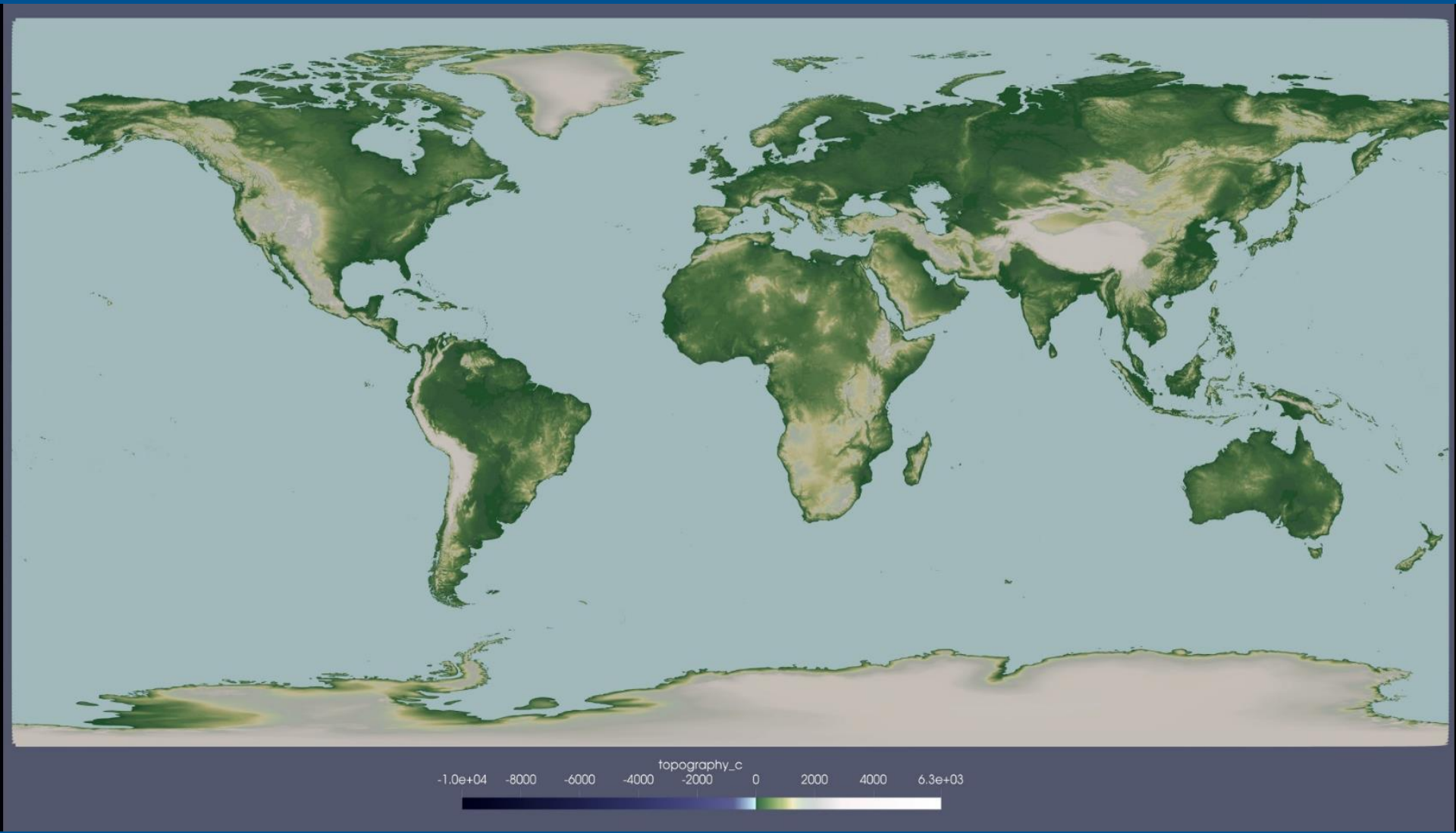

Mt Fuji (2311m)
Height (m)
0.0
500.0
1000.0
1500.0
2000.0
2500.0
3000.0


ICON 2.5km

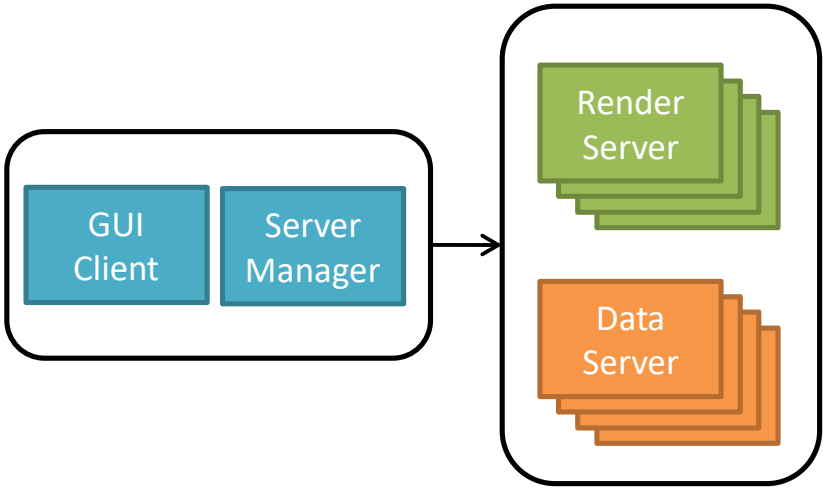

Mt Fuji (3166m)
Height (m)
0.0
500.0
1000.0
1500.0
2000.0
2500.0
3000.0

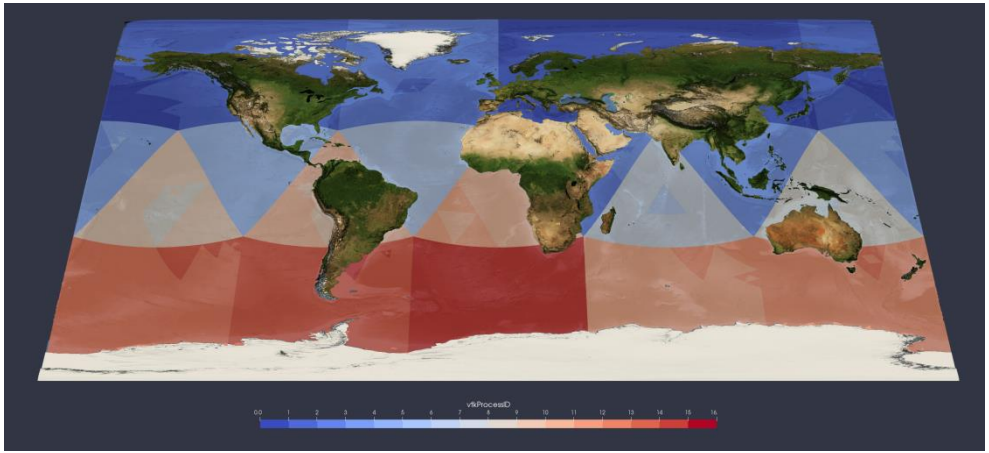

SAM 4km

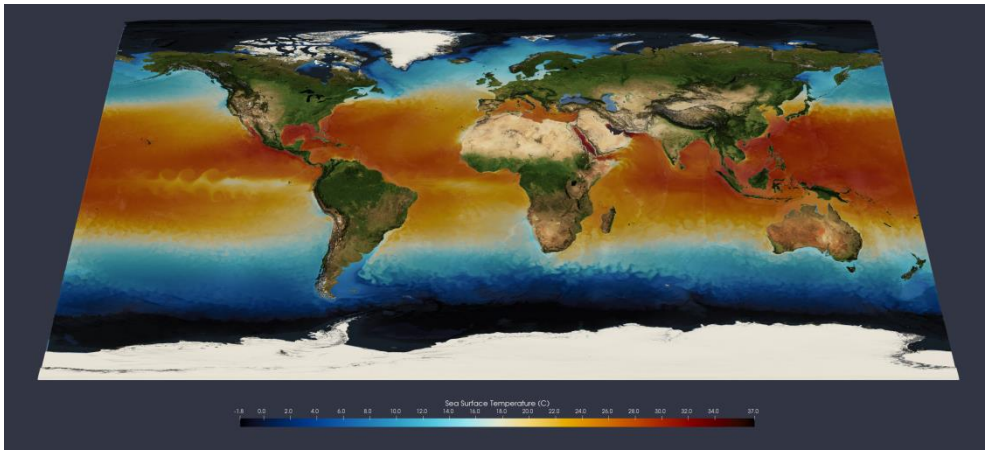

Mt Fuji (2316m)
Height (m)
0.0
500.0
1000.0
1500.0
2000.0
2500.0
3000.0


topography_c
-1.0e+04
-8000
-6000
-4000
-2000
0
2000
4000
6.3e+03

Parallel Processing and Visualization



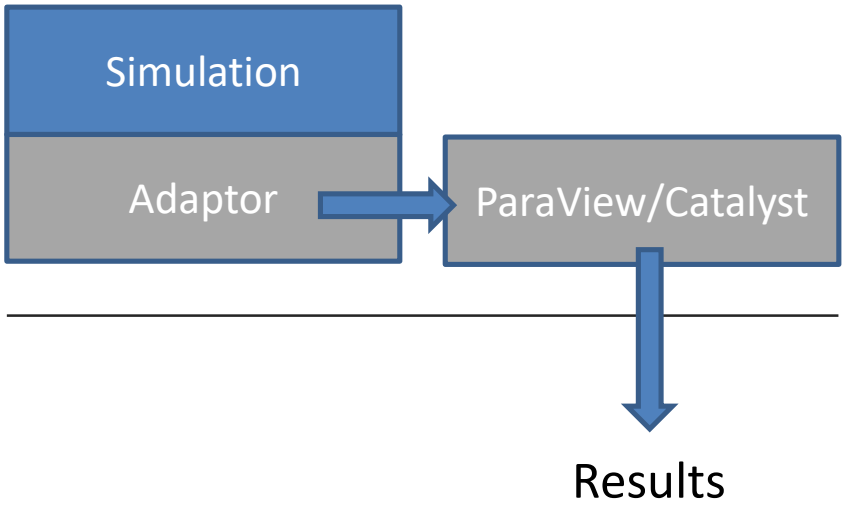




Large Data

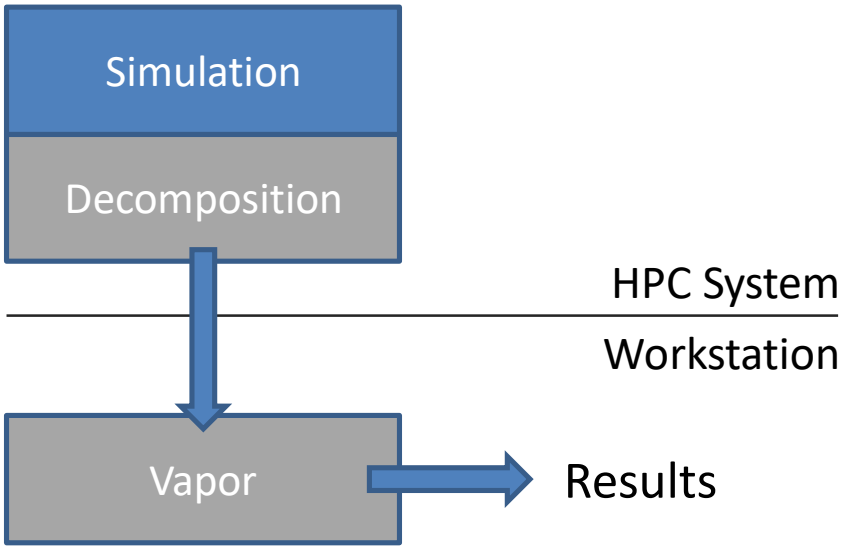
in-situ Visualization

(ParaView/Catalyst)



in-situ Compression

(VAPOR)

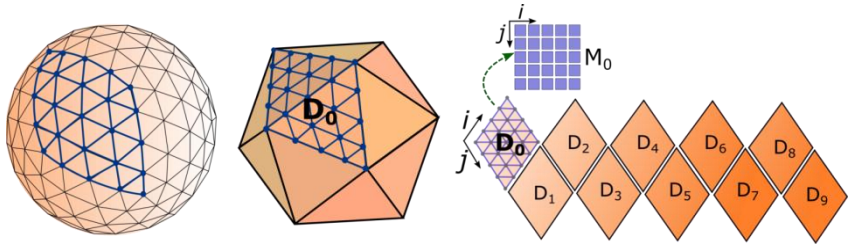


Discrete Hex Wavelet Transform for ICON/MPAS

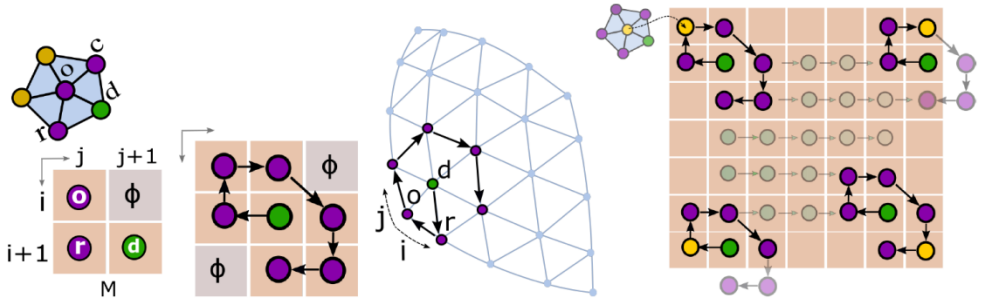
esiwace CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

Decompose sphere into 10 diamonds

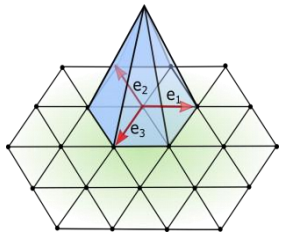
- Diamond vertices at original base icosahedron vertices
- Each diamond has regular topology



Map centroids of quad, triangles, and hexagons cells into a hexagonal mesh with explicit connectivity



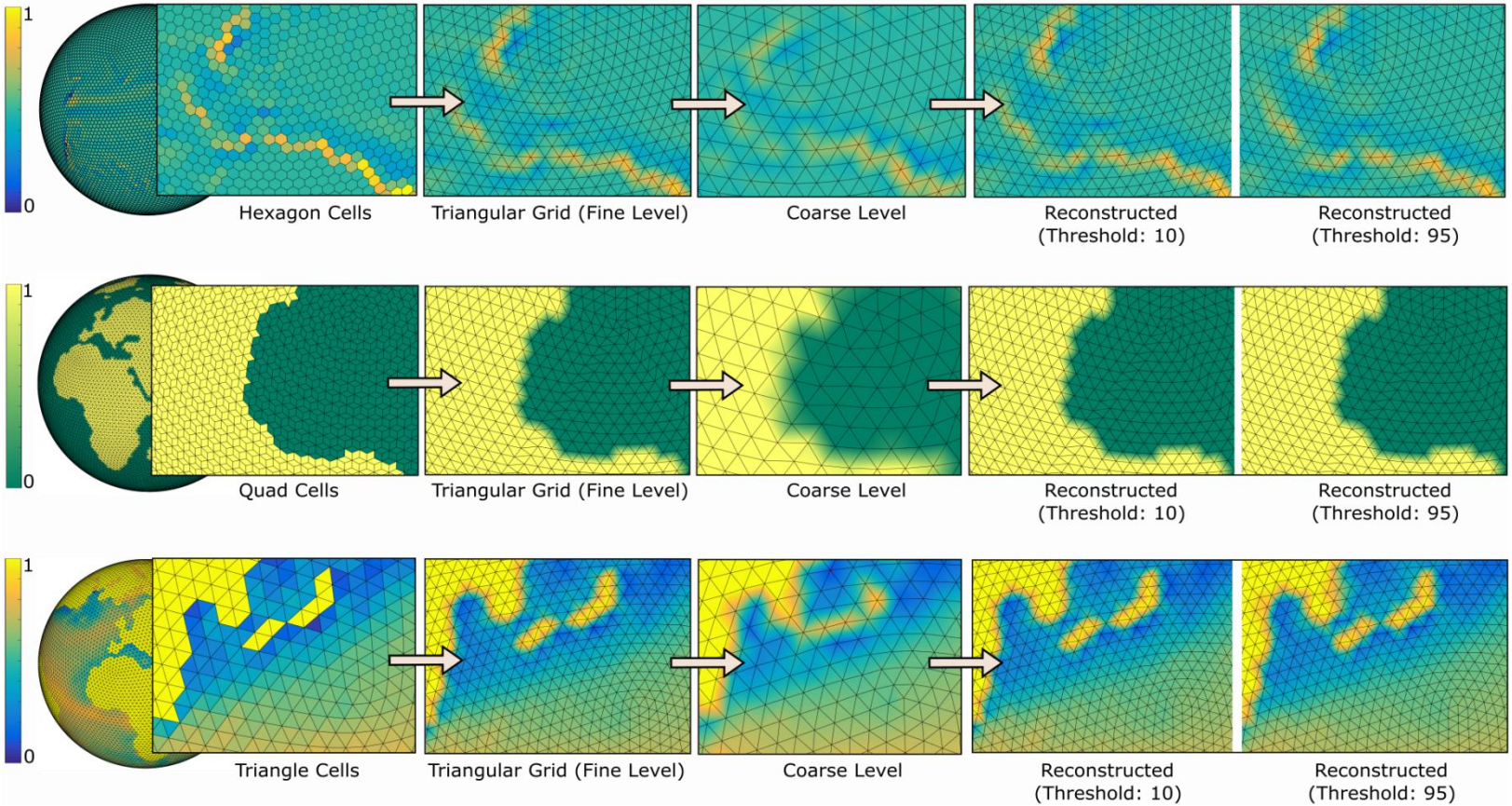
Apply discrete wavelet transform to each regular hexagonal mesh (diamond)



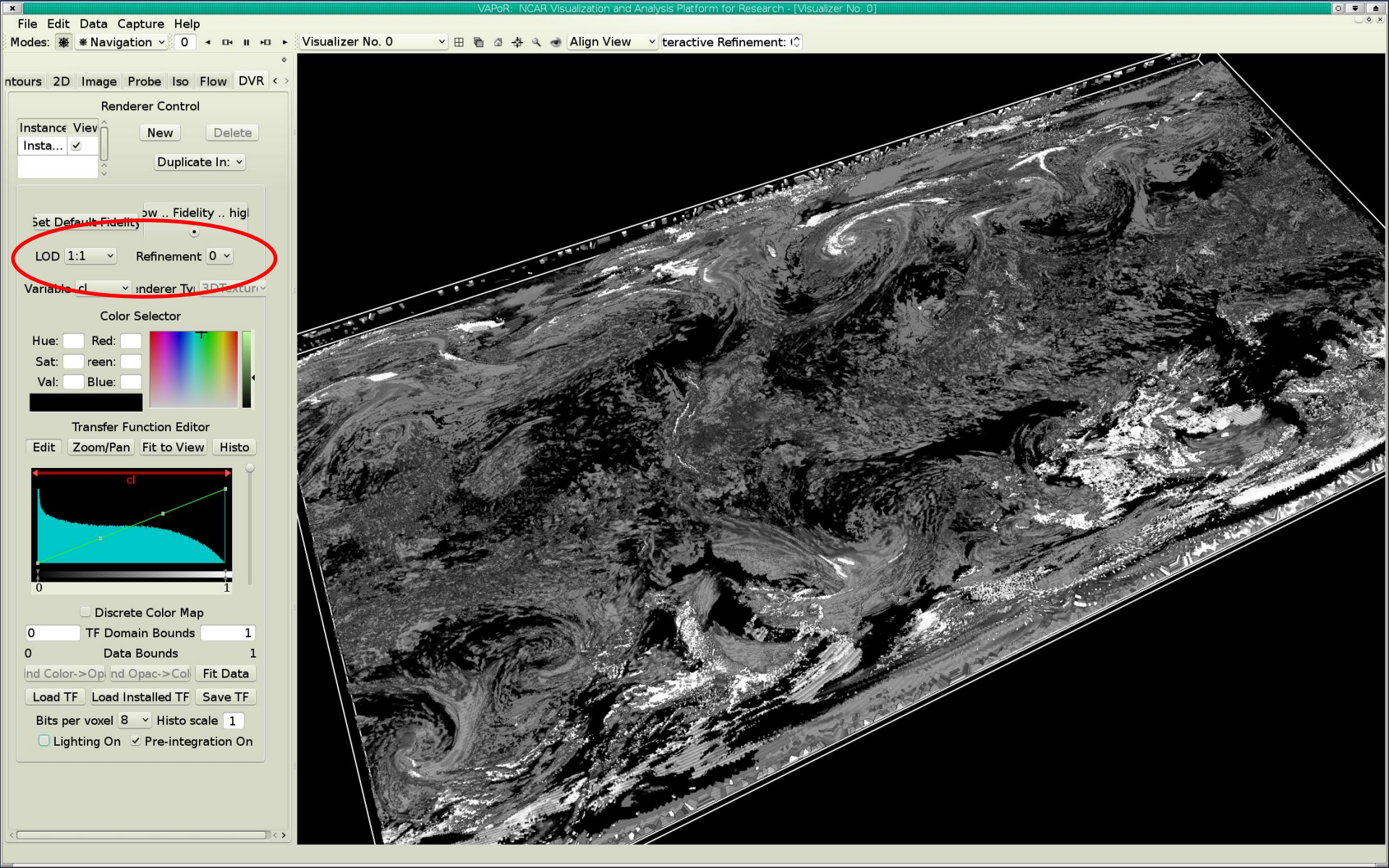
$$f(\mathbf{x}) = \sum_{\mathbf{k}} \overbrace{F[\mathbf{k}]}^{\text{data}} \overbrace{\varphi(\mathbf{x}-\mathbf{L}\mathbf{k})}^{\text{box spline}}$$

[1] Jubair et.al. *"Icosahedral Maps for a Multiresolution Representation of Earth Data"*, VMV 2016

Multiresolution Presentation

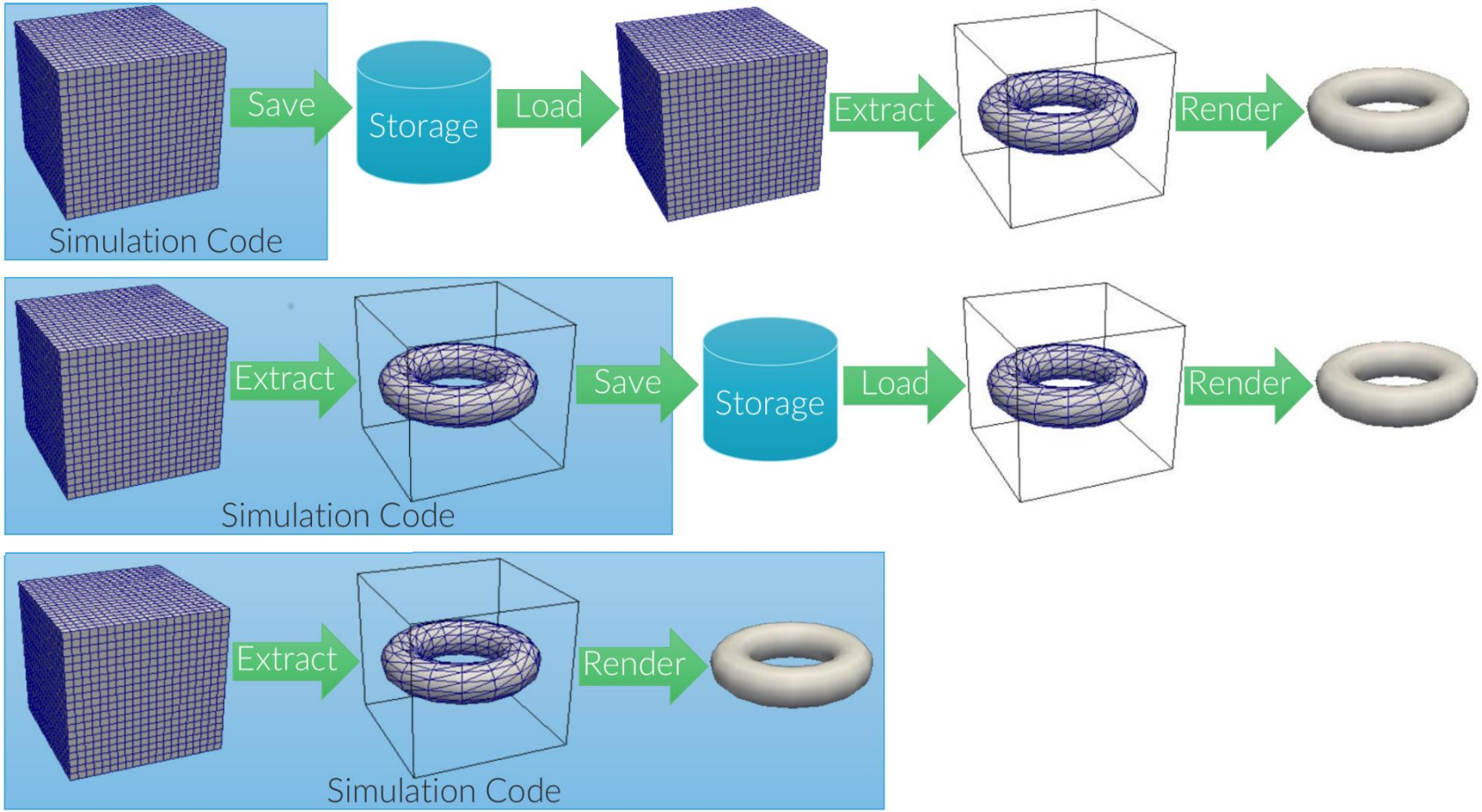


[1] Jubair et.al. *"Icosahedral Maps for a Multiresolution Representation of Earth Data"*, VMV 2016

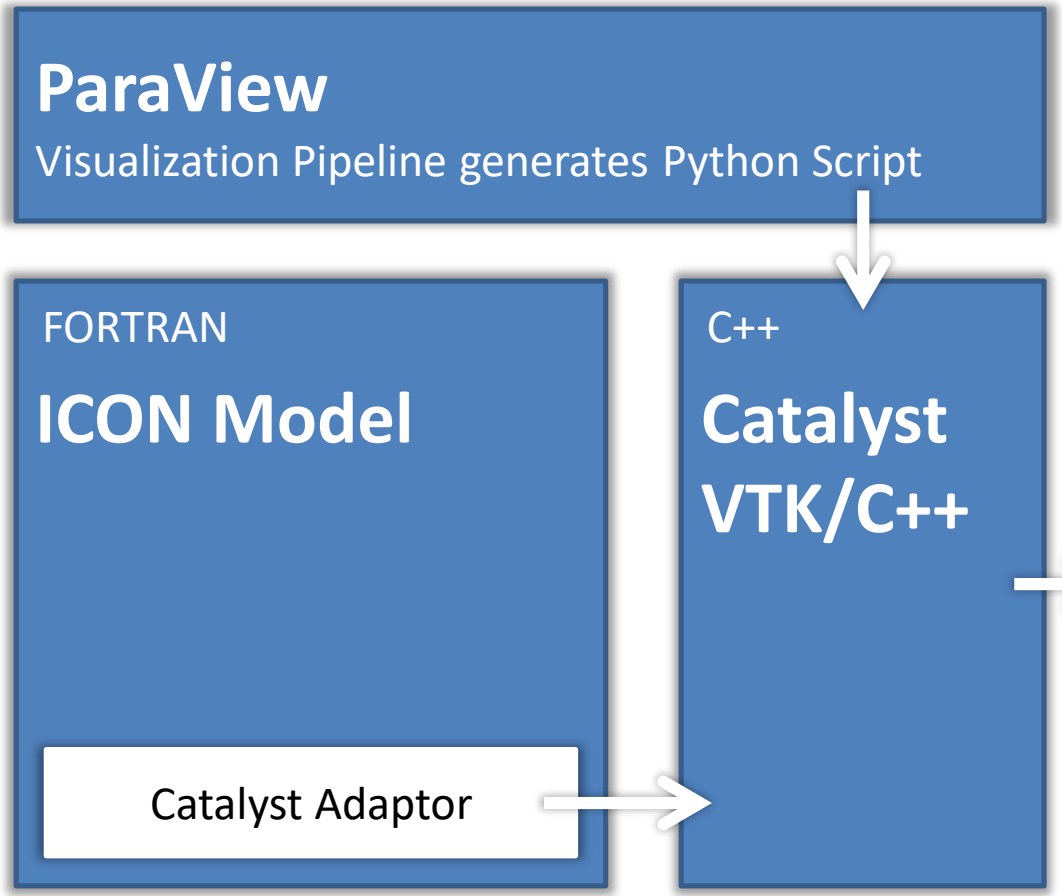

VAPoR: NCAR Visualization and Analysis Platform for Research - [Visualizer No. 0]
File Edit Data Capture Help
Modes: Navigation 0
Visualizer No. 0
Align View
teractive Refinement:
ntours 2D Image Probe Iso Flow DVR
Renderer Control
Instance View
Insta...
New
Delete
Duplicate In:
Set Default Fidelity
ow .. Fidelity .. high
LOD 1:1
Refinement 0
Variable cl
enderer Ty 3DTexture
Color Selector
Hue:
Red:
Sat:
reen:
Val:
Blue:
Transfer Function Editor
Edit
Zoom/Pan
Fit to View
Histo
cl
0
1
Discrete Color Map
0
TF Domain Bounds
1
0
Data Bounds
1
nd Color->Op
nd Opac->Col
Fit Data
Load TF
Load Installed TF
Save TF
Bits per voxel 8
Histo scale 1
Lighting On
Pre-integration On

From Post to In-Situ Visualization/Processing

esiwace
CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE



ICON and Catalyst Adaptor



- Rendered images
- Cinema database
- Data reduction
- Feature det./tracking (cloud classification)
- Live visualization
- Compression

Advantages

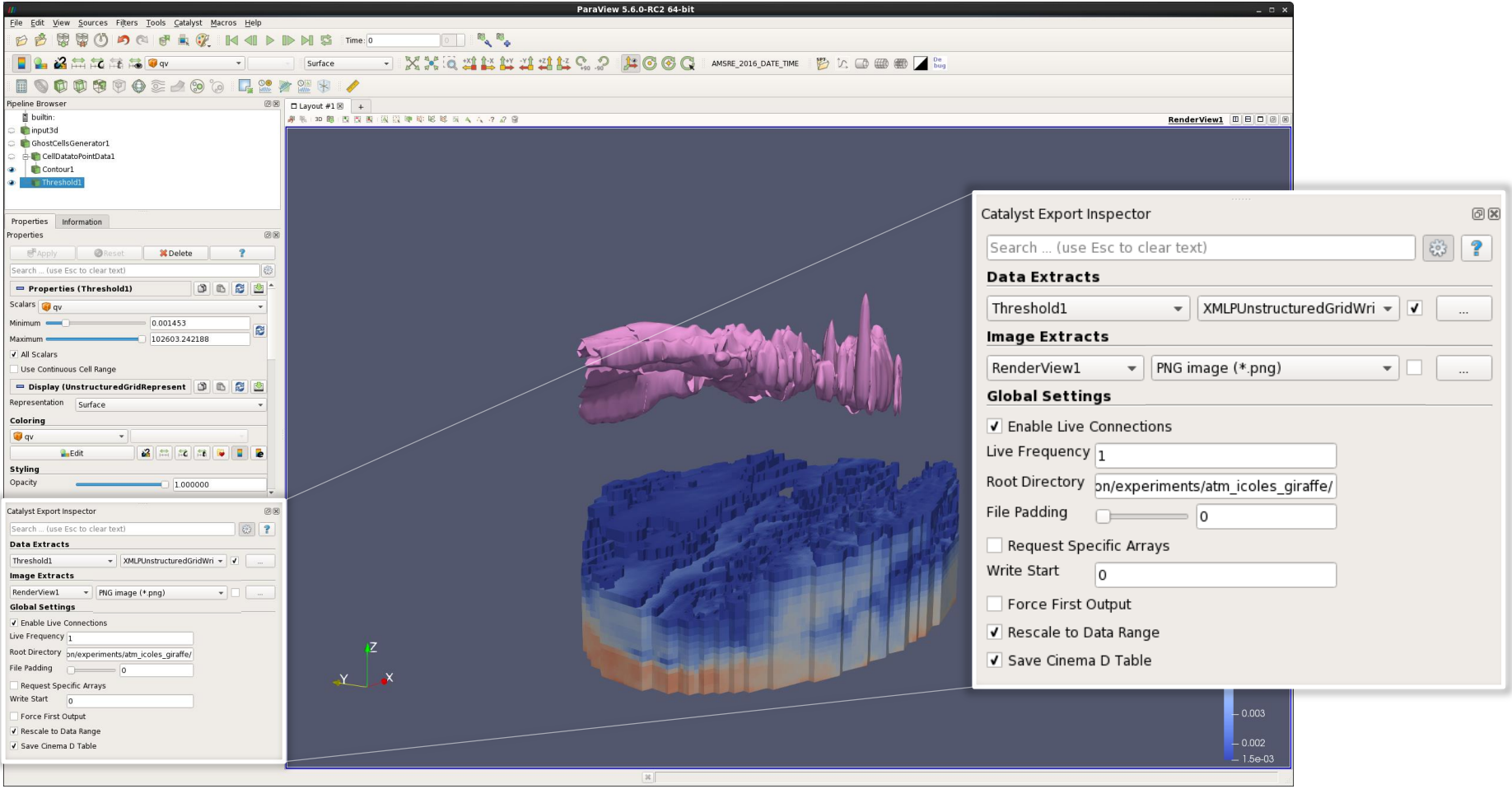
- Much less I/O

  -> Faster / less disk space
- Preview of data
- Allows to run extremely large simulations
- Time to knowledge much shorter

Drawbacks

- Additional resources required
- A priori knowledge needed
- Need to run sim/vis again for new visualizations
- Complexity increases

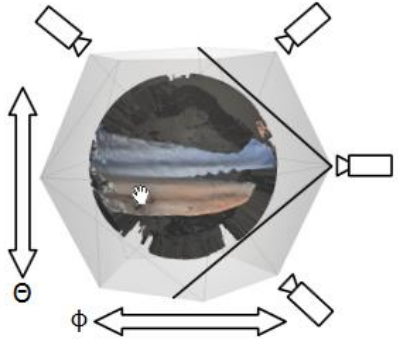
Generating a Catalyst Script

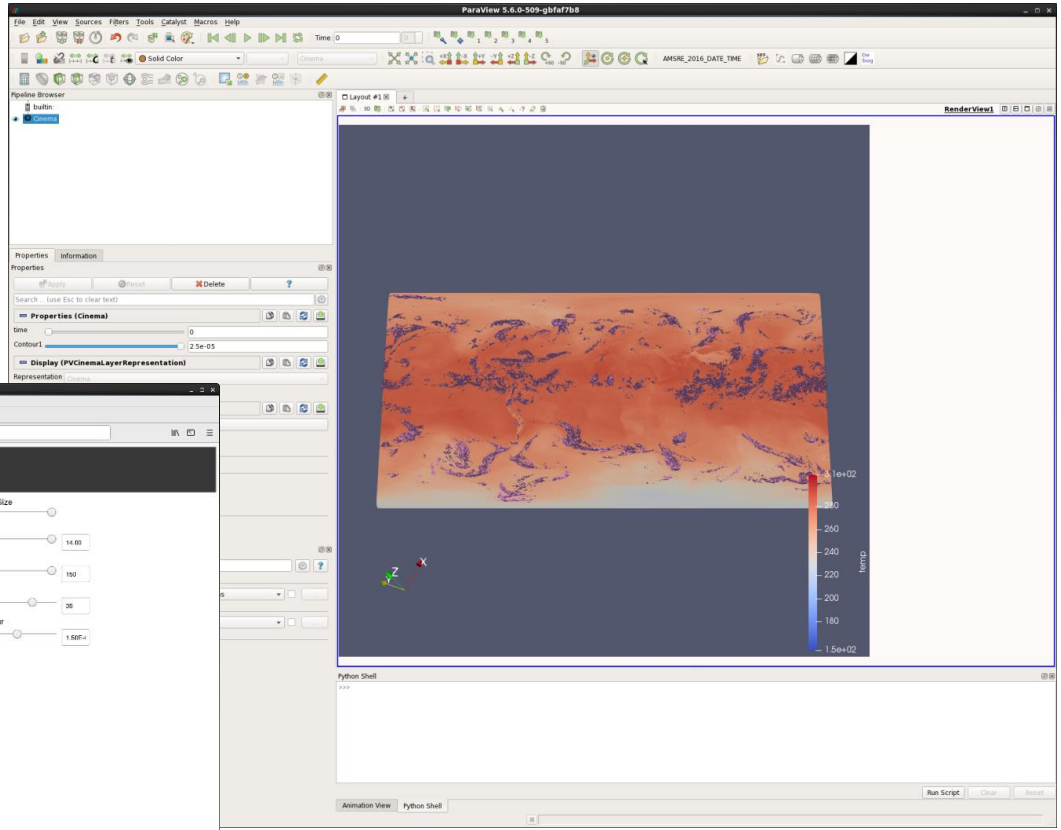


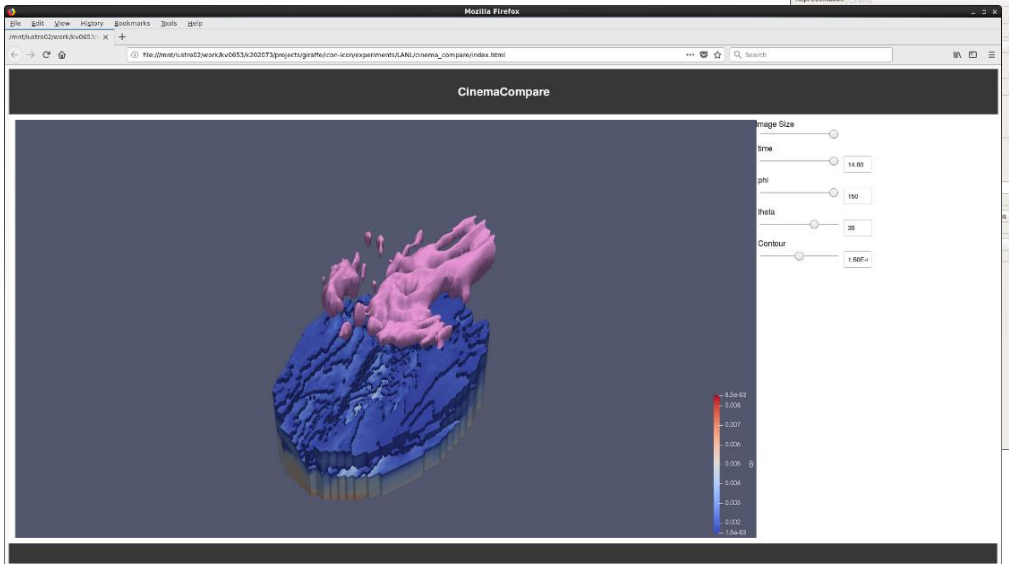
Catalyst Script


```
183
184  class CoProcessor(coprocessing.CoProcessor):
185    def CreatePipeline(self, datadescription):
186      self.Pipeline = _CreatePipeline(self, datadescription)
187
188  coprocessor = CoProcessor()
189  # these are the frequencies at which the coprocessor updates.
190  freqs = {'input3d': [1, 1, 1, 1, 1, 1]}
191  coprocessor.SetUpdateFrequencies(freqs)
192  if requestSpecificArrays:
193    arrays = [['pressure', 1], ['qc', 1], ['qi', 1], ['qr', 1], ['qv', 1], ['temp', 1]]
194    coprocessor.SetRequestedArrays('input3d', arrays)
195  coprocessor.SetInitialOutputOptions(timeStepToStartOutputAt,forceOutputAtFirstCall)
196
197  if rootDirectory:
198      coprocessor.SetRootDirectory(rootDirectory)
199
200  if make_cinema_table:
201      coprocessor.EnableCinemaDTable()
202
203  return coprocessor
204
205
206 #--------------------------------------------------------------
207 # Global variable that will hold the pipeline for each timestep
208 # Creating the CoProcessor object, doesn't actually create the ParaView pipeline.
209 # It will be automatically setup when coprocessor.UpdateProducers() is called the
210 # first time.
211 coprocessor = CreateCoProcessor()
212
213 #--------------------------------------------------------------
214 # Enable Live-Visualizaton with ParaView and the update frequency
215 coprocessor.EnableLiveVisualization(True, 1)
216
217 # ---------------------- Data Selection method ----------------------
218
219 def RequestDataDescription(datadescription):
220     "Callback to populate the request for current timestep"
221     global coprocessor
222
223     # setup requests for all inputs based on the requirements of the
224     # pipeline.
225     coprocessor.LoadRequestedData(datadescription)
226
227 # ------------------------ Processing method ----------------------
228
229 def DoCoProcessing(datadescription):
230     "Callback to do co-processing for current timestep"
231     global coprocessor
232
233     # Update the coprocessor by providing it the newly generated simulation data.
234     # If the pipeline hasn't been setup yet, this will setup the pipeline.
235     coprocessor.UpdateProducers(datadescription)
```

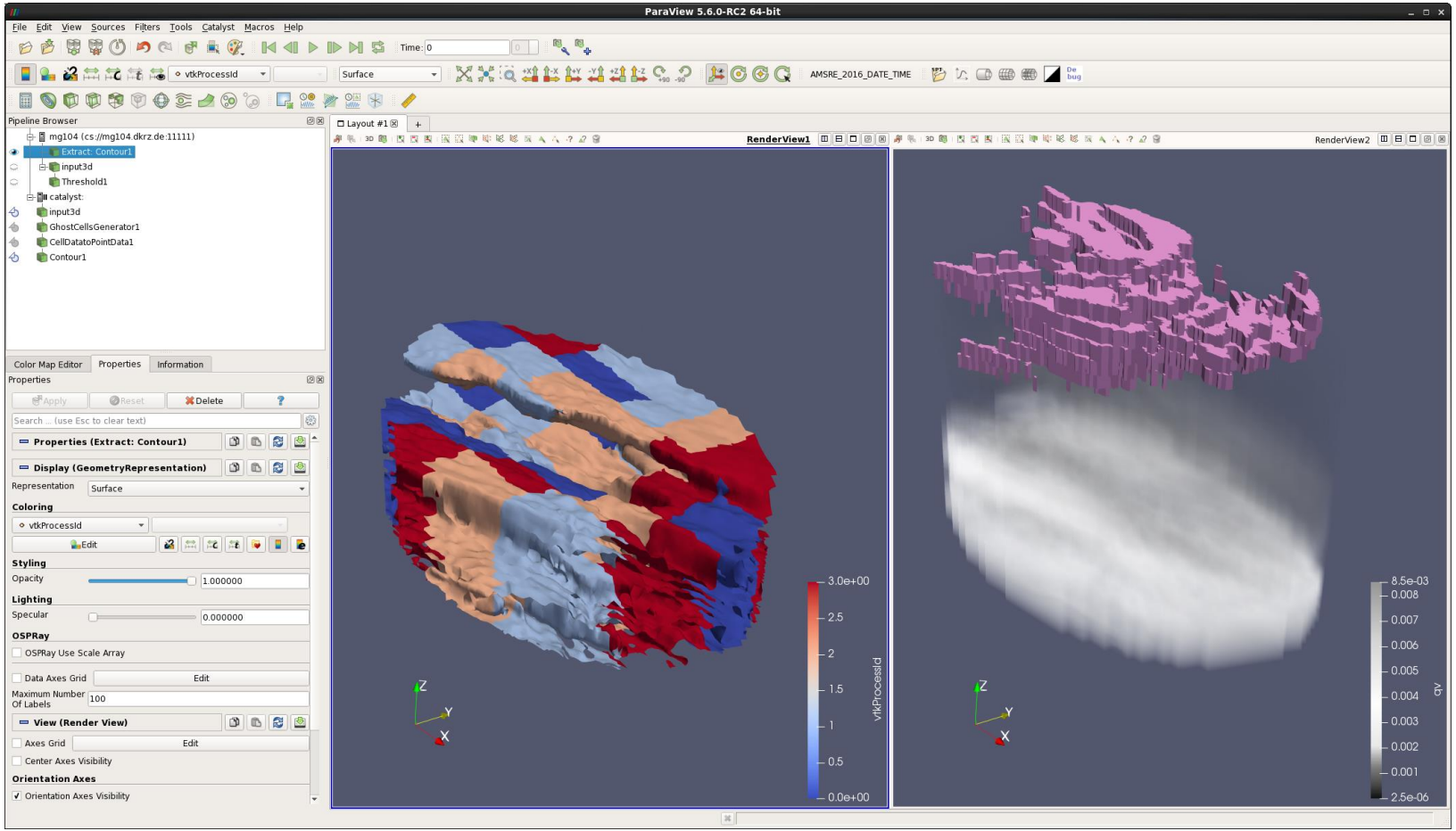
CINEMA Image Database

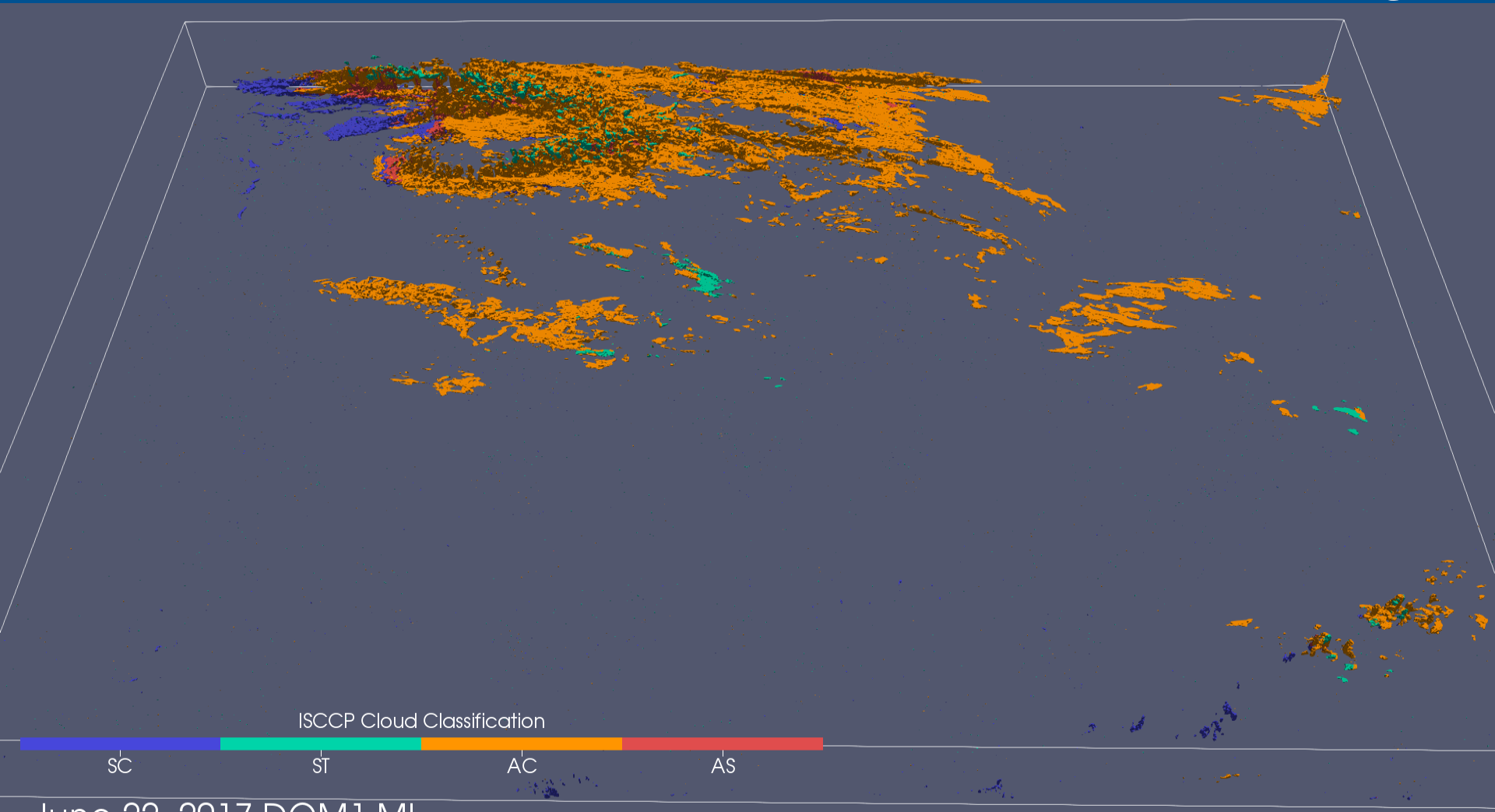






Live Visualization, Steering?





June 22, 2017 DOM1 ML

Implementation

- Started refactoring other in-situ code -> too complex
- Started fresh -> few hundred lines in FORTRAN and C++ with minimal changes to ICON
- Zero Copy Arrays FORTRAN -> C++
- Tightly coupled (in line), even number of sim/vis processes
- Not everything works yet (CINEMA)

Timings R2B10 – 2.5km global / 540 nodes

esiwace CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

name	# calls	t_min	min r	t_avg	t_max	max r	total min (s)	total min r	total max (s)	total max rank
total	4305	06m48s	[659]	**06m48s**	06m48s	[3919]	**408.010**	[659]	408.027	[3919]
L wrt_output	8610	0.00778s	[1756]	11.8735s	23.7090s	[3239]	23.707	[1469]	23.779	[2838]
L integrate_nh	344400	3.9458s	[3476]	4.3407s	15.2784s	[256]	347.016	[3476]	347.308	[0]
L nh_solve	1722000	0.29028s	[0]	0.45025s	1.1131s	[216]	156.504	[2048]	190.621	[47]
L nh_hdiff	344400	0.09548s	[1368]	0.13672s	0.44944s	[420]	8.426	[2111]	15.589	[1852]
L physics	344400	0.53099s	[418]	0.94401s	12.2728s	[2598]	57.132	[421]	101.765	[2831]
....										
L insitu_set_var	344400	0.01999s	[7]	**0.02430s**	0.05760s	[221]	**1.663**	[7]	2.037	[126]
L insitu_do_work	340095	0.00014s	[1095]	**0.06853s**	0.72341s	[0]	**5.312**	[2392]	10.067	[0]
L insitu_do_work1st	4305	1.5387s	[2239]	**1.6174s**	2.0325s	[0]	**1.539**	[2239]	2.033	[0]
....										
model_init	12915	1.5042s	[1752]	**01m11s**	03m01s	[1672]	**214.388**	[1990]	215.458	[885]
L insitu_init	4305	4.9177s	[1990]	**6.1077s**	6.3881s	[4164]	**4.918**	[1990]	6.388	[4164]

Timings R2B10 – 2.5km global / 540 nodes

esiwace
CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

name	# calls	t_min	min r	t_avg	t_max	max r	total min (s)	total min r	total max (s)	total max rank
total	4305	06m48s	[6	**06m48s**	48s	[3	**408.010**	[659]	408.027	[3919]
L wrt_output	8610	0.00778s	[17		090s	[		[1469]	23.779	[2838]
L integrate_nh	344400	3.9458s	[34		784s			[3476]	347.308	[0]
L nh_solve	1722000	0.29028s			131s			[2048]	190.621	[47]
L nh_hdiff	344400	0.09548s	[13		944s			[2111]	15.589	[1852]
L physics	344400	0.53099s	[4	**0.02430s**	728s		**1.663**	[421]	101.765	[2831]
....										
L insitu_set_var	344400	0.01999s		**0.06853s**	760s		**5.312**	[7]	2.037	[126]
L insitu_do_work	340095	0.00014s	[10	**1.6174s**	341s		**1.539**	[2392]	10.067	[0]
L insitu_do_work1st	4305	1.5387s	[22		325s			[2239]	2.033	[0]
....										
model_init	12915	1.5042s	[17	**01m11s**	001s	[	**214.388**	[1990]	215.458	[885]
L insitu_init	4305	4.9177s	[19	**6.1077s**	381s		**4.918**	[1990]	6.388	[4164]

News

- New supercomputer „Mistral" at DKRZ delivers particularly detailed regional climate simulations for Germany
  Oct 05, 2015
- Preview: DKRZ at SC'15
  Sep 30, 2015
- Kick-off for ESiWACE and ESCAPE
  Sep 29, 2015
- Allocations 2016 - request resources

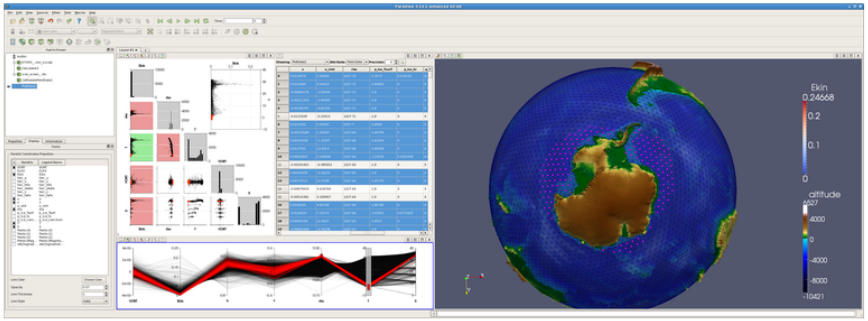
ParaView

Paraview is an open source visualization package that reads a variety of different data formats and lattices and implements the most common visualization techniques. More specifically, Paraview also reads netCDF files and supports different grids, so that it can be used to visualize climate and earth science data sets.

Paraview 4.1 is installed on all visualization nodes of Halo and can be started from the command line via 'paraview'. Older versions of Paraview can be started by appending the version number, such as 'paraview3.98'.

Paraview has come a long way and is used and developed by a very large community from a variety of different sciences. It is installed on DKRZ's Halo nodes since the end of 2012, and we have now prepared a little tutorial that will teach you how to use Paraview for the visualization of your own climate research data.

More general information on Paraview, along with some tutorial data can also be found online on the Paraview website.



The above example shows a complex visualization of an ICON ocean data set using Paraview. The viewport on the right displays the data, the selection made, as well as the Earth's topography. The three viewports on the left hand side are used to specify the selection, based on a scatterplot matrix and parallel coordinates. These techniques are especially well suited for an in-depth data analysis and exploration.

Paraview Tutorial

The final tutorial document will comprise 8 chapters and will be released at the end of the summer in 2014. Alongside, we will provide courses to teach Paraview in a hands-on setting. The first course will already start in December 2013.

Here is a glimpse of the content from the tutorial:

- Chapter 1 "Introduction and Overview" --- The first Chapter starts with an overview of Paraview and briefly explains the underlying visualization toolkit pipeline. The second part of this chapter concentrates on an introduction of the

What's next

- Develop in-situ vis workflows for end users
- Integration of in-situ with ICON Ocean (eg. eddy census) and other models (as part of ESiWACE2)
- In-Situ decomposition/(lossy) compression

- Pipeline automation (machine learning)

esiwace
CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

ESiWACE has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 675191

www.esiwace.eu