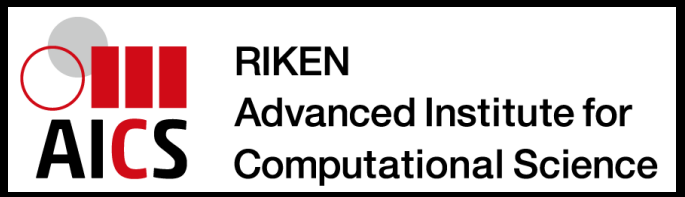
大規模データの可視化・分析環境の構築と展開

Kenji ONO

Advanced Visualization Research Team

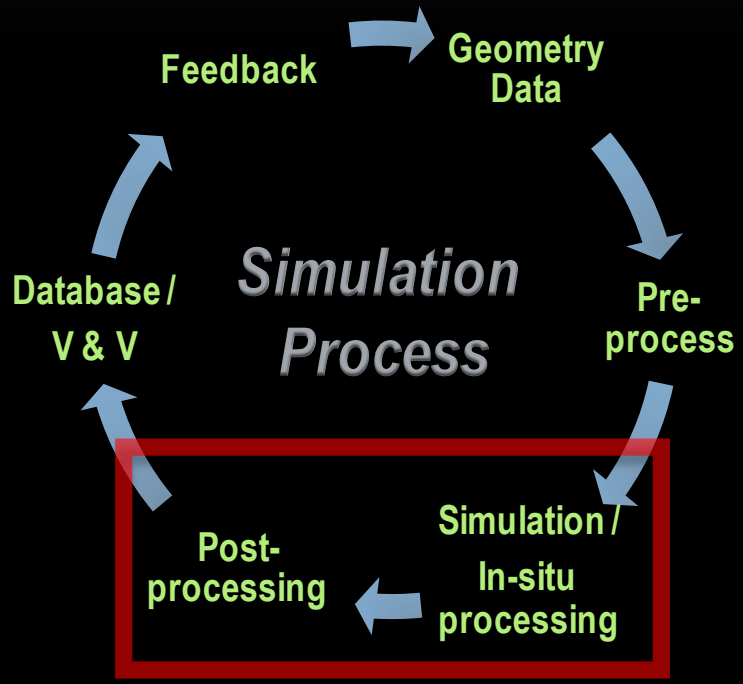
Advanced Institute for Computational Science, RIKEN



ISSUES AND DIRECTION

- Effective use of huge computational resources
  - Capacity computing
    - Exploration of design space
    - Uncertainty quantification
    - Parameter sweep
  - Capability computing
    - Ultra-short turnaround simulation

Many trials
Organization of prjects
Handling of large-scale data
Automation
…



ISSUES OF VISUALIZATION FOR LARGE-SCALE DATASET

- Large-scale parallel simulation
  - Numerous, distributed files

  ⇒ Scalable algorithm

  File handling
- We can't move / copy data files

  ⇒ Remote visualization
- Trials and errors
  - To find best parameters for vis.

  ⇒ Interactivity
- Various vis. scenarios

  ⇒ Customizable
- Various computer environments

  ⇒ Multi-platform
- Long term support
  - Less dependency for arch.

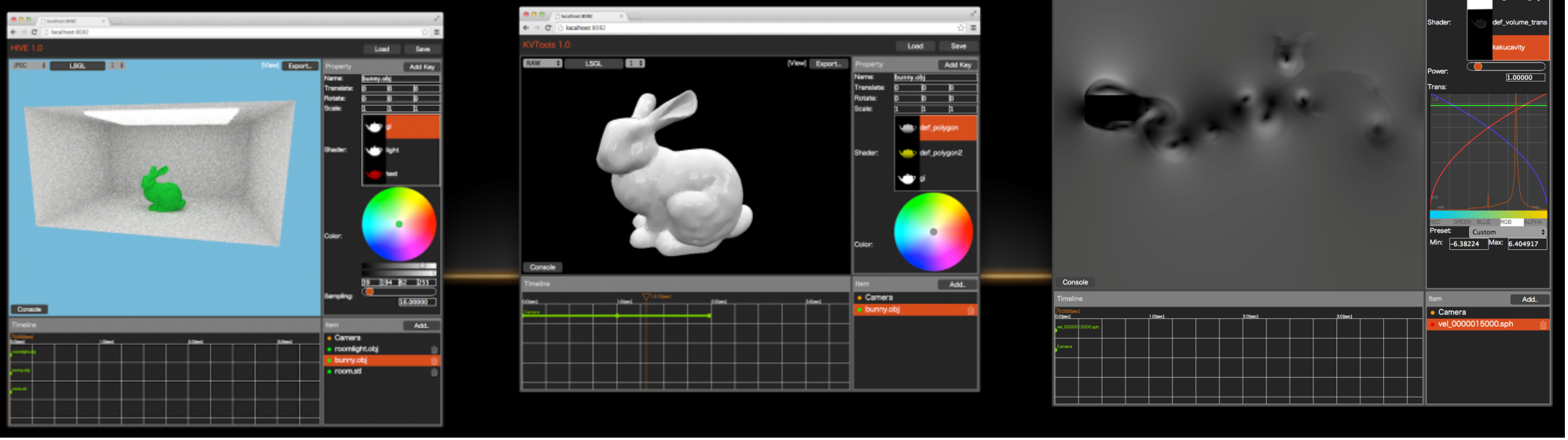
  ⇒ Use recent technology

ISSUES OF SOFTWARE DEVELOPMENT

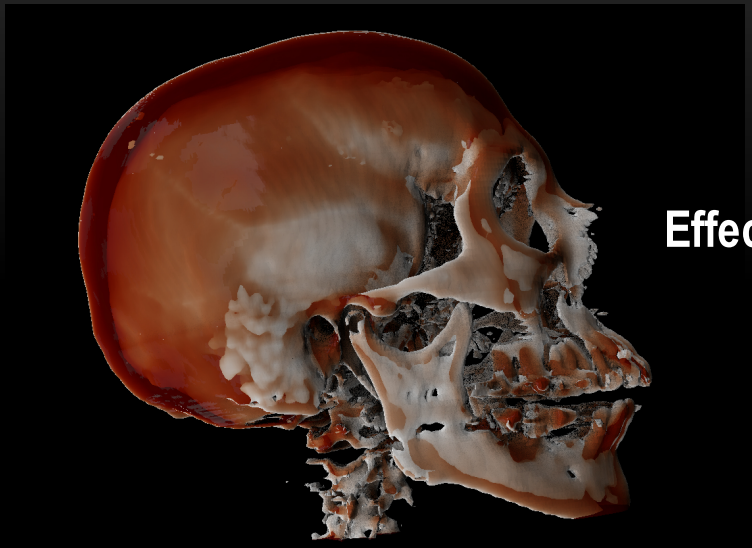
- Selection of parallelism
  - Sort-first / -middle / -last
- Selection of rendering API
  - OpenGL or C or others
- Support architectures
  - Intel, AMD, FX/K, ARM, mobile,…
- Software life cycle and maintenance for new architectures
  - FX/K, coming Exa. machines

HIVE

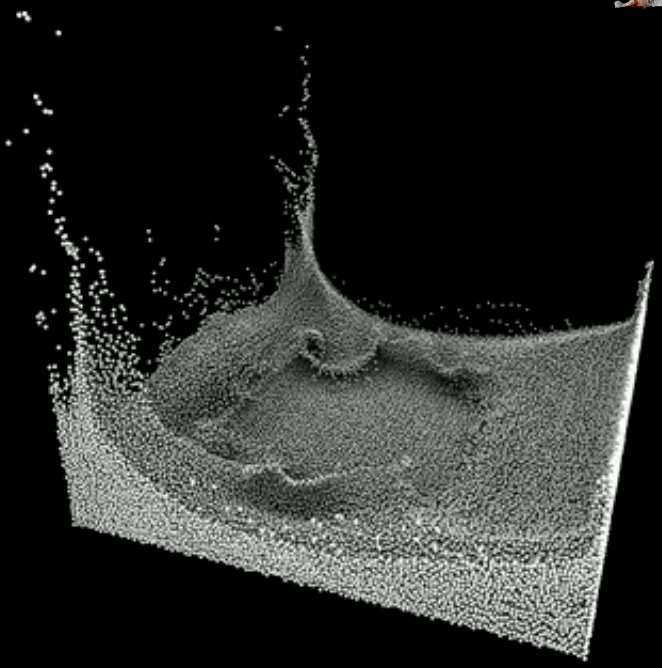
- HIVE offers multiple visualization scenarios to users
  - { Parallel, Remote, Interactive } visualization
- Web-based architecture => Ubiquitous
  - Interactive exploration of visual parameters and layout
- Multi-platform
  - K-Computer, Intel Cluster, PC,...
  - Linux, Windows, Mac



EXAMPLES

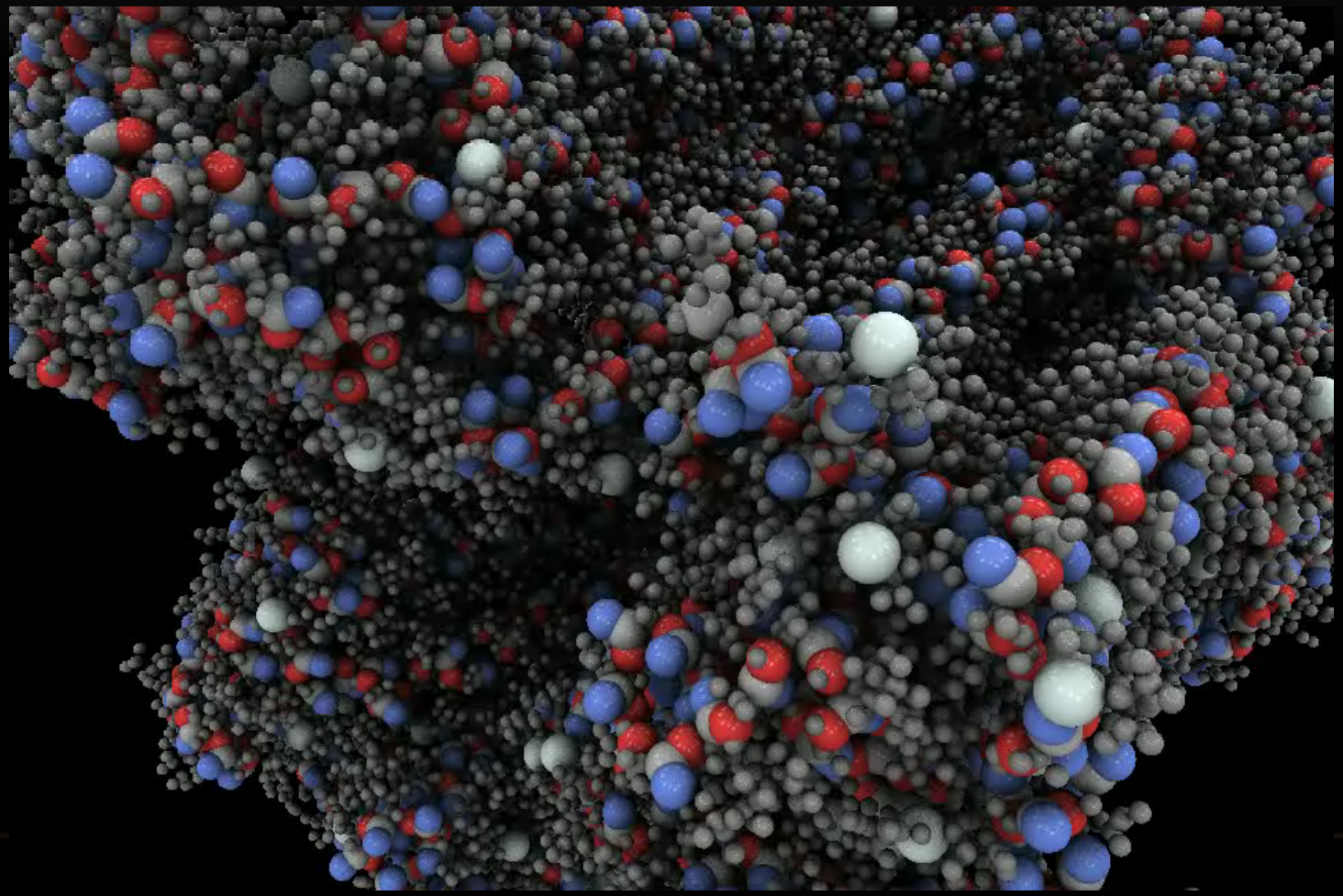


Effect of sub-surface scattering

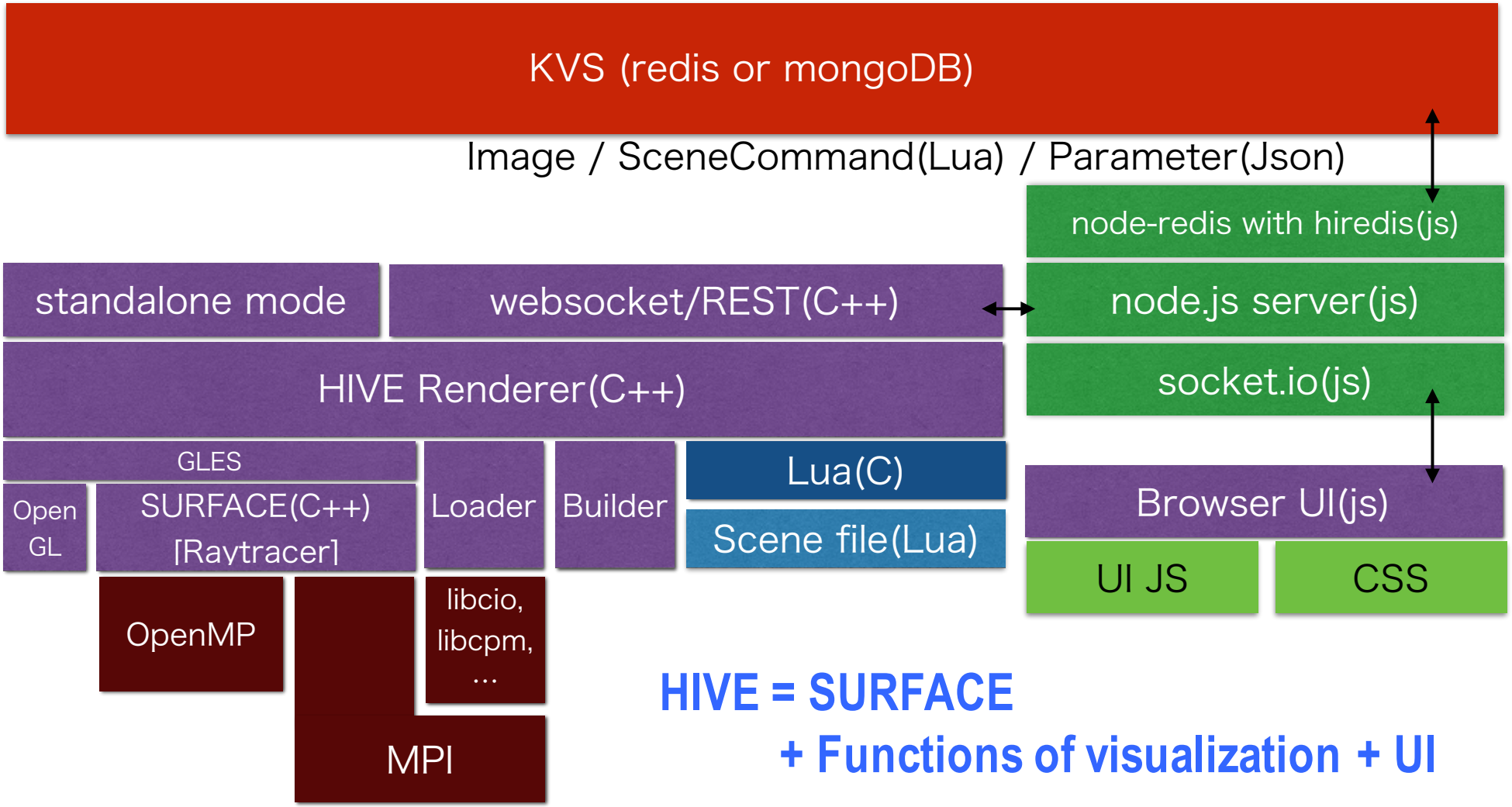


Resolution of 4096 x 4096 pixels, Asian Dragon

HIGH RES. RENDERING IMAGE

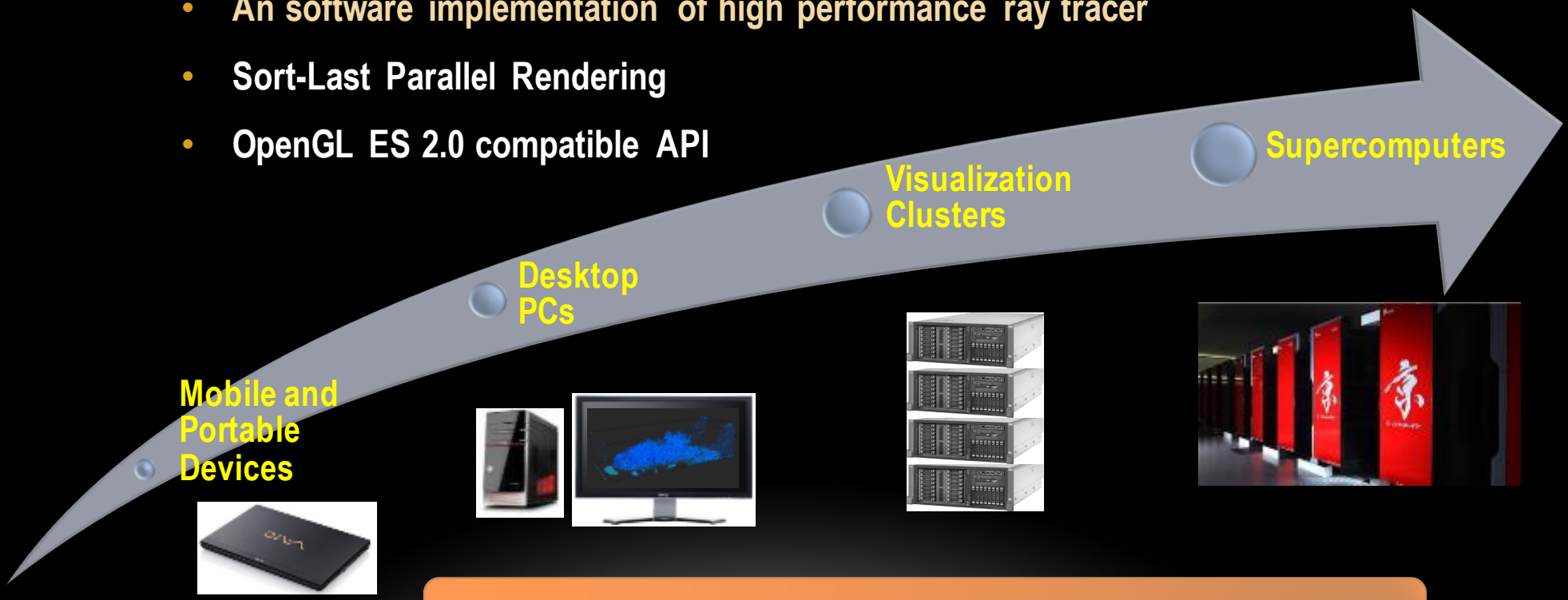


SOFTWARE STACK OF HIVE



SURFACE

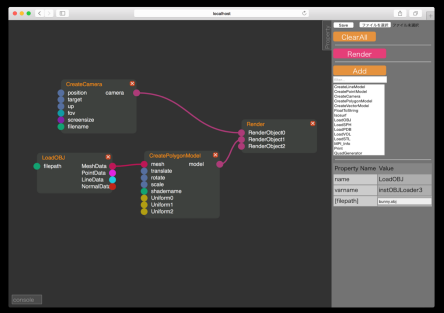
- Scalable and Ubiquitous Rendering Framework for Advanced Computing Environment
  - An software implementation of high performance ray tracer
  - Sort-Last Parallel Rendering
  - OpenGL ES 2.0 compatible API

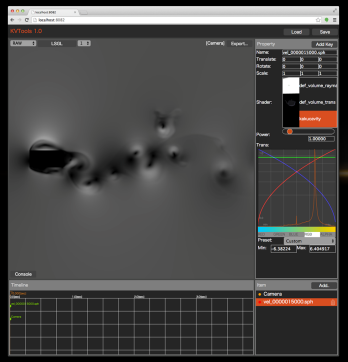


HIVE APPLICATION

HIVE composes of 3 components

```
# Rendering by command line

$ cd $HIVE/hrender/test
$ mpirun -np 1
../../build/bin/hrender
render_obj.scn
```





hrender

SURFACE

standalone renderer

raytracing library

Scene Node Editor

web interface node editor

HIVE UI

web interface animation editor

HRENDER

- Wrapper app. of SURFACE and data I/O
- Command line module

Web-based HIVE UI

Scene file (parameter)

command line

Interactive
Batch Job

hrender

Data I/O

SURFACE

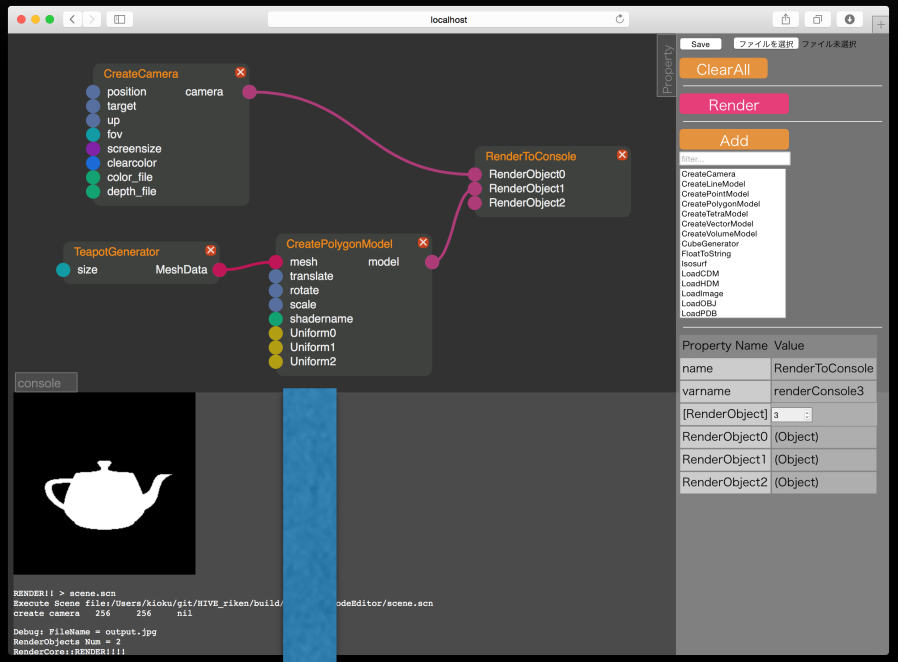
Parallel Processing
MPI
OpenMP

Data Loader
xDMlib

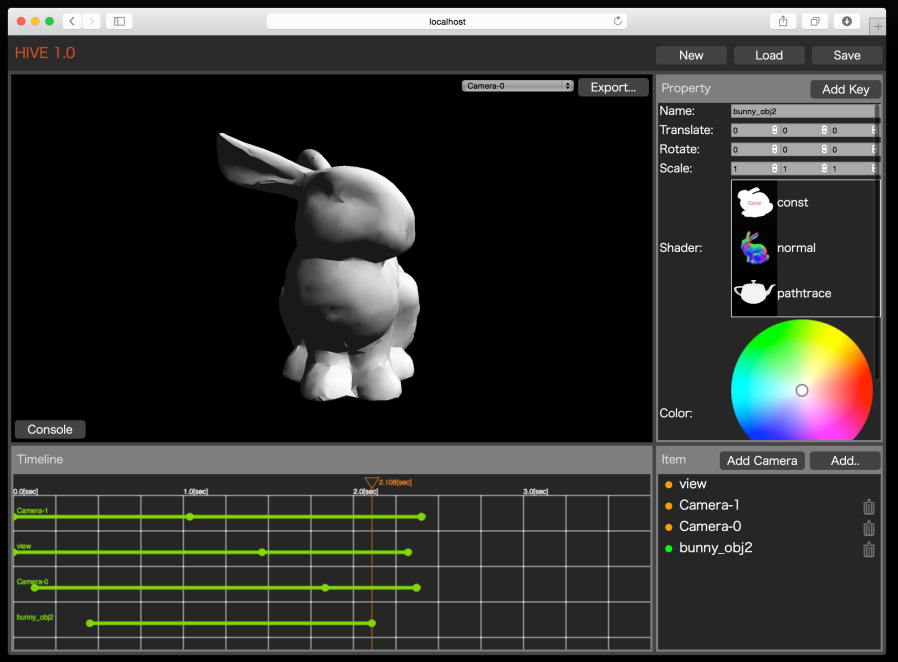
Data Exporter
Chowder

WEB-BASE APP. ON HRENDER

Scene Node Editor



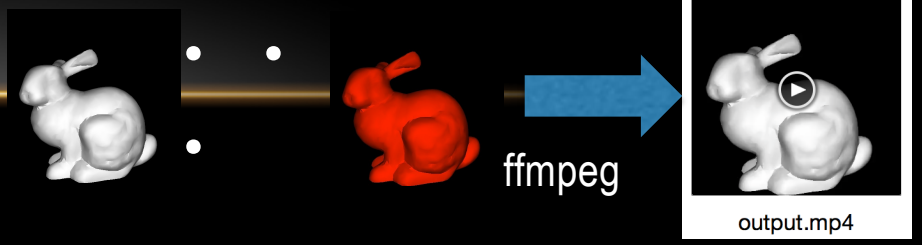
HIVE UI



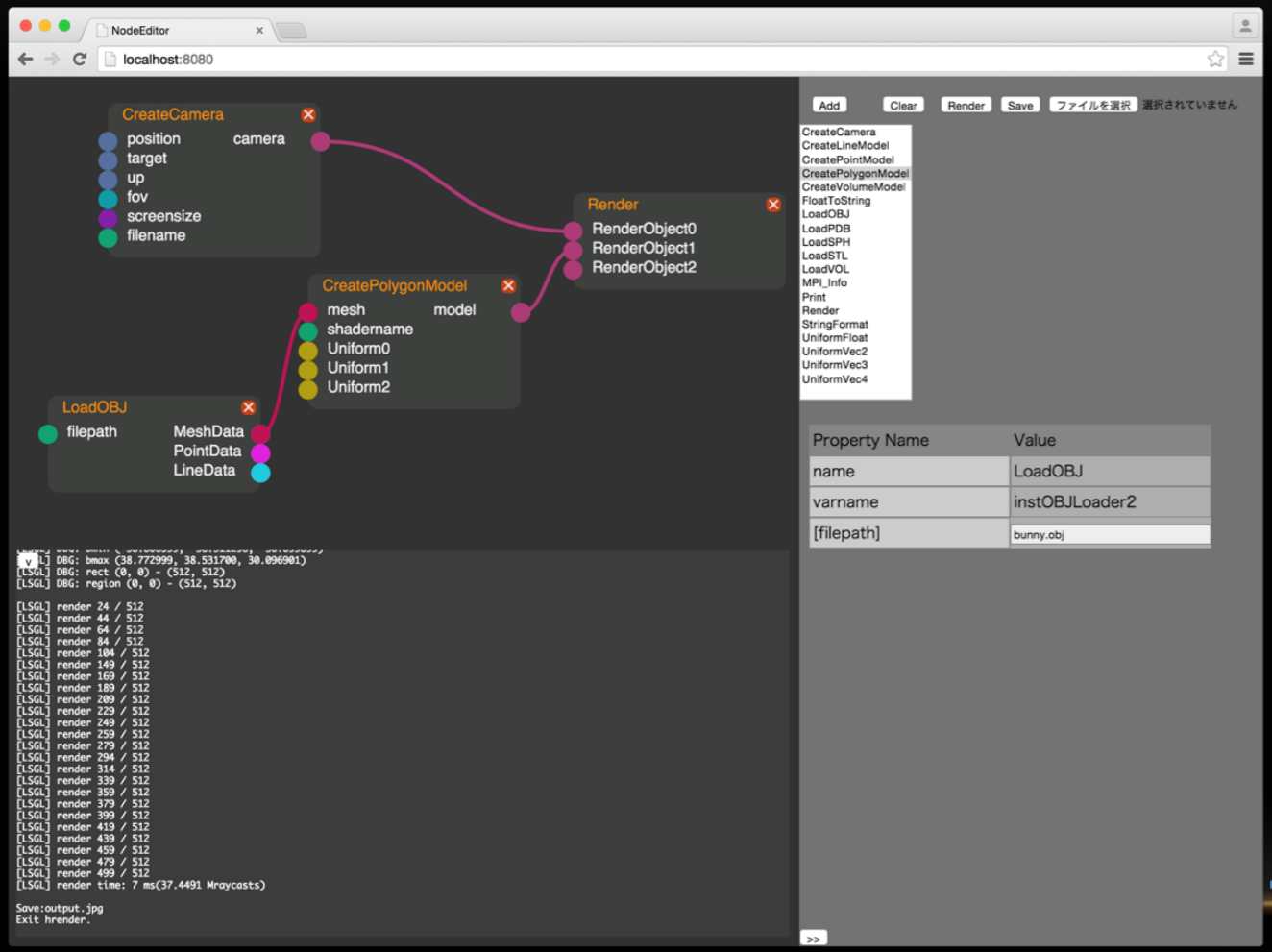
Tools for describing Scene file

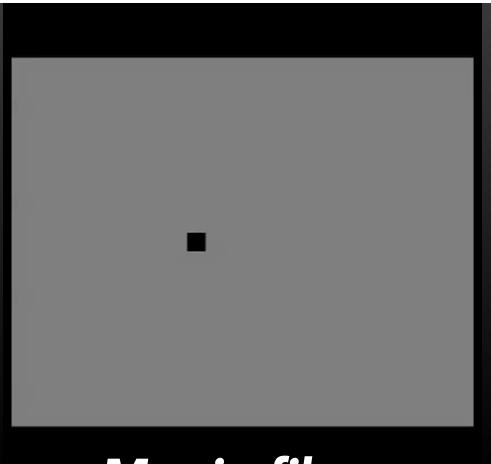
```
print('Render Obj')
local camera = Camera()
camera:SetScreenSize(1024, 1024)
camera:SetFilename('render_obj.jpg')

camera:LookAt(
        0.0,1.0,0.0,
        0.0,0.0,
        0.1.0,
        60
)

local obj = OBJLoader()
obj:Load('bunny.obj')

local model = PolygonModel()
local meshdata = obj:MeshData()
model:Create(meshdata)
model:SetShader('normal.frag')

local analy = PolygonAnalyzer()
analy:Execute(model)
print(analy:MinX())
print(analy:MaxX())
print(analy:MinY())
print(analy:MaxY())
print(analy:MinZ())
print(analy:MaxZ())


render {camera, model}
```

Tools for making movie

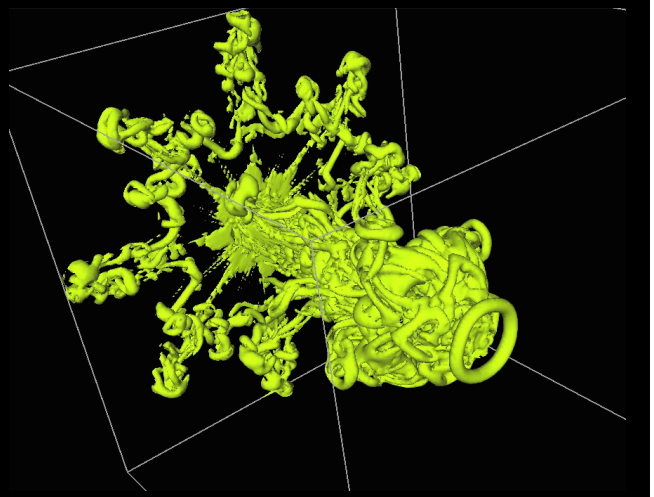


SCENE NODE EDITOR





Movie file



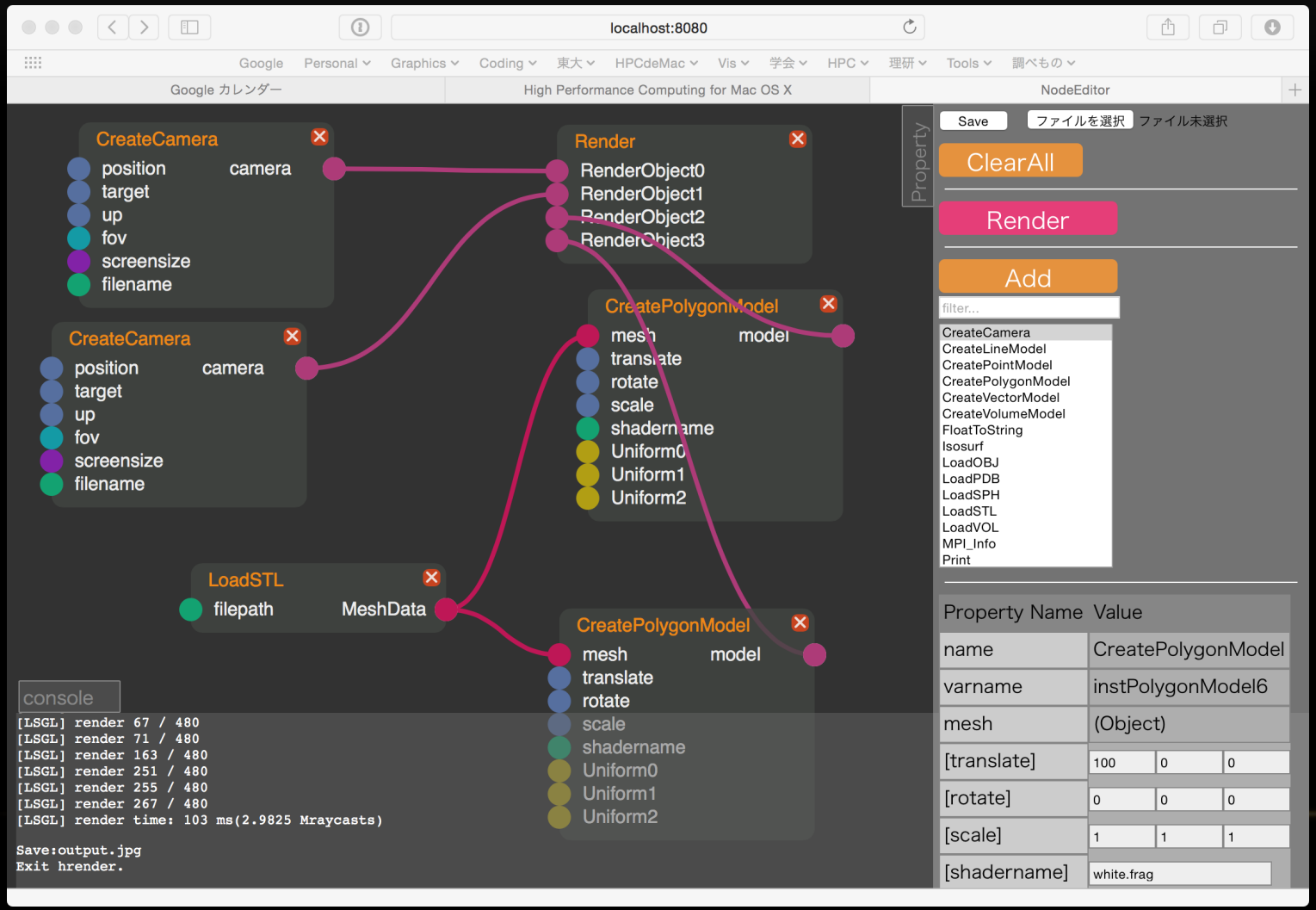
output.jpg

hrender

scene.scn

Enable us to construct rendering pipeline flexibly

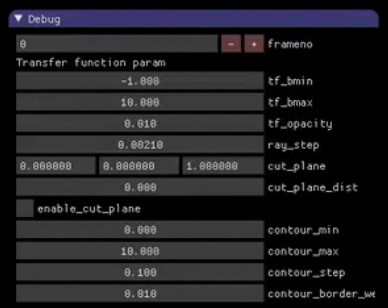
MORE


localhost:8080
Google
Personal
Graphics
Coding
東大
HPCdeMac
Vis
学会
HPC
理研
Tools
調べもの
Google カレンダー
High Performance Computing for Mac OS X
NodeEditor
CreateCamera
position
target
up
fov
screensize
filename
camera
Render
RenderObject0
RenderObject1
RenderObject2
RenderObject3
CreateCamera
position
target
up
fov
screensize
filename
camera
CreatePolygonModel
mesh
translate
rotate
scale
shadername
Uniform0
Uniform1
Uniform2
model
LoadSTL
filepath
MeshData
CreatePolygonModel
mesh
translate
rotate
scale
shadername
Uniform0
Uniform1
Uniform2
model
console
[LSGL] render 67 / 480
[LSGL] render 71 / 480
[LSGL] render 163 / 480
[LSGL] render 251 / 480
[LSGL] render 255 / 480
[LSGL] render 267 / 480
[LSGL] render time: 103 ms(2.9825 Mraycasts)
Save:output.jpg
Exit hrender.
Property
Save
ファイルを選択
ファイル未選択
ClearAll
Render
Add
filter...
CreateCamera
CreateLineModel
CreatePointModel
CreatePolygonModel
CreateVectorModel
CreateVolumeModel
FloatToString
Isosurf
LoadOBJ
LoadPDB
LoadSPH
LoadSTL
LoadVOL
MPI_Info
Print
Property Name
Value
name
CreatePolygonModel
varname
instPolygonModel6
mesh
(Object)
[translate]
100
0
0
[rotate]
0
0
0
[scale]
1
1
1
[shadername]
white.frag

SHADER PROGRAM

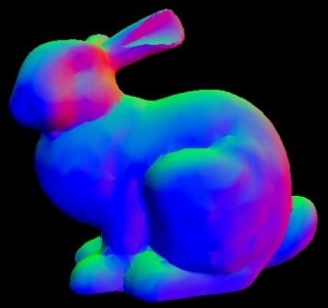
GL Shading Language

- Users can add their own codes for special purpose rendering

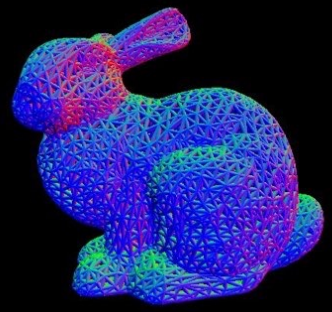


```
#extension GL_LSGL_trace : enable
#extension GL_OES_texture_3D : enable

#ifdef GL_ES
precision mediump float;
#endif

uniform sampler3D tex0;
uniform vec2  resolution;
varying float matID;
varying vec3  mnormal;
uniform vec4  color;

uniform vec3 volumescale;
uniform vec3 volumedim;
uniform vec3 offset;
uniform vec3 eye;
uniform vec3 lookat;
uniform vec3 up;

void main(void)
{
  vec3 p;
  vec3 n;
  vec3 dir;
  isectinfo(p, n, dir);

  vec3 rayorg = eye;
  vec3 raydir = p - eye;

  vec3 texpos = (p - offset) / volumescale + 0.5; // [0,
1]^3
  vec4 dens = texture3D(tex0, texpos);

  gl_FragColor = vec4(normalize(dens.xyz), 1.0);
  return;
}
```

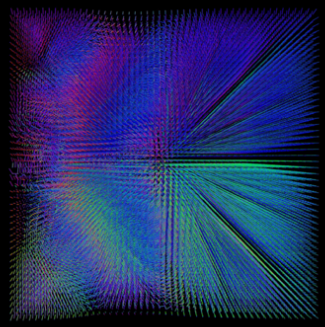
RENDERING BY DIFFERENT SHADER



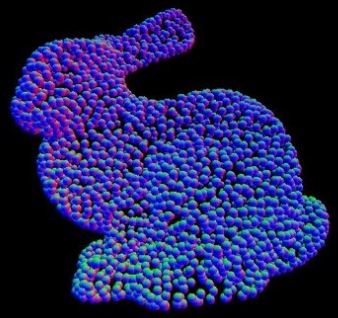
Polygon



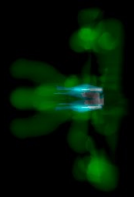
Line



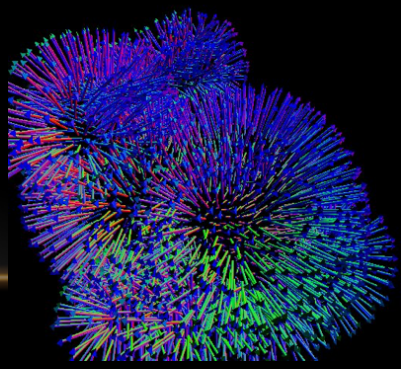
Volume-VectorArrow



Point

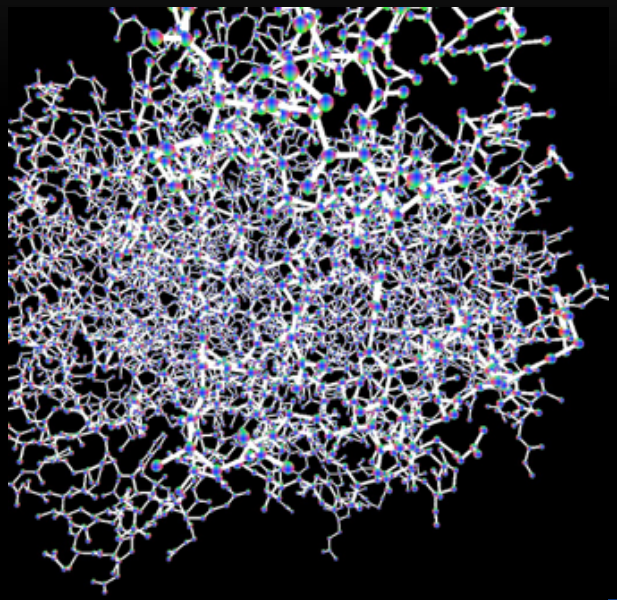


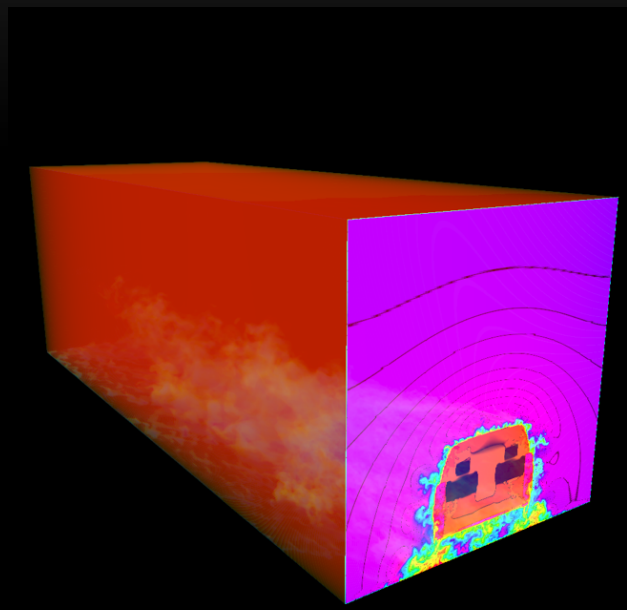
Volume

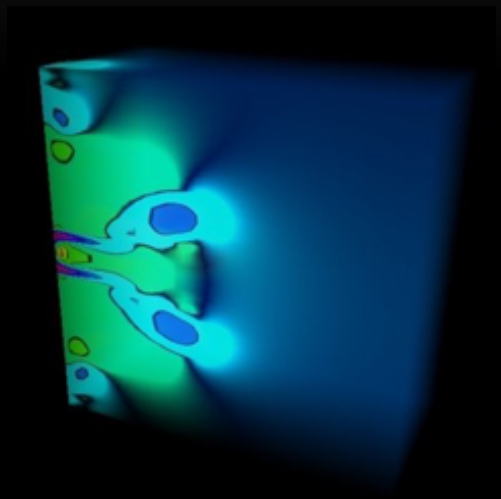


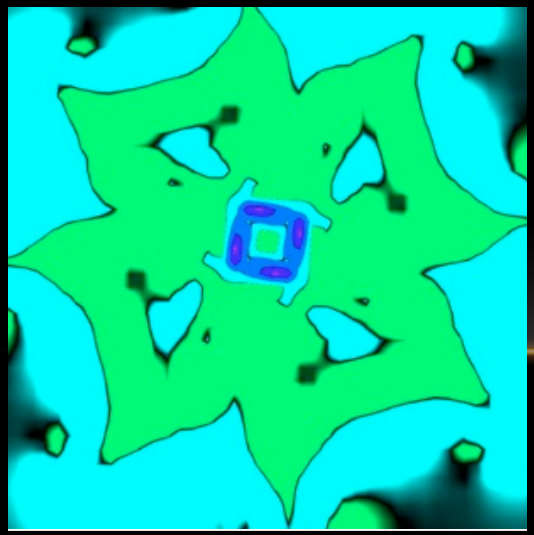
VectorArrow(Normal)

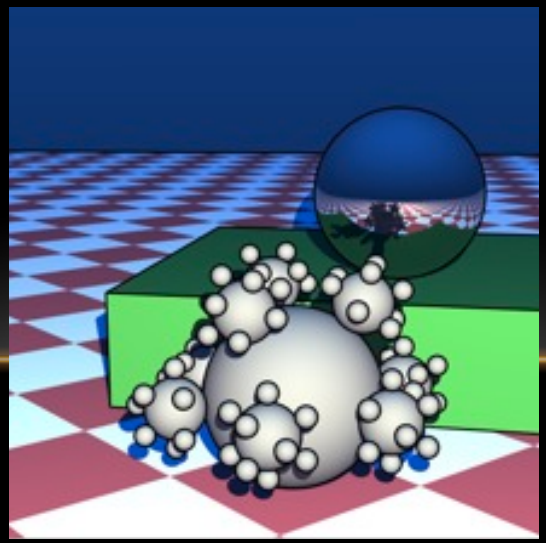
OTHER EXPRESSION



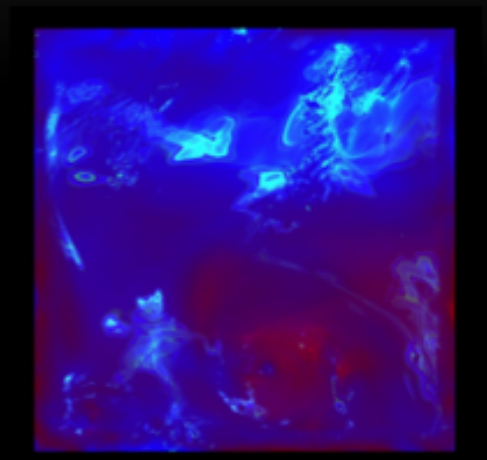


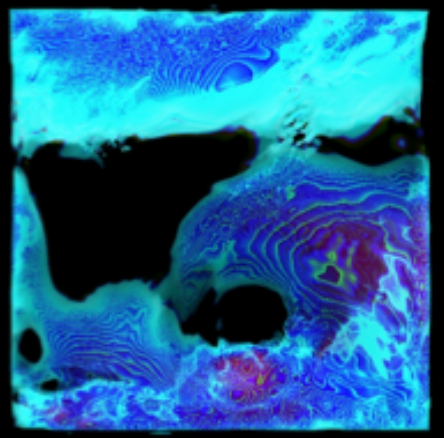






EXAMPLE OF HURRICANE DATASET





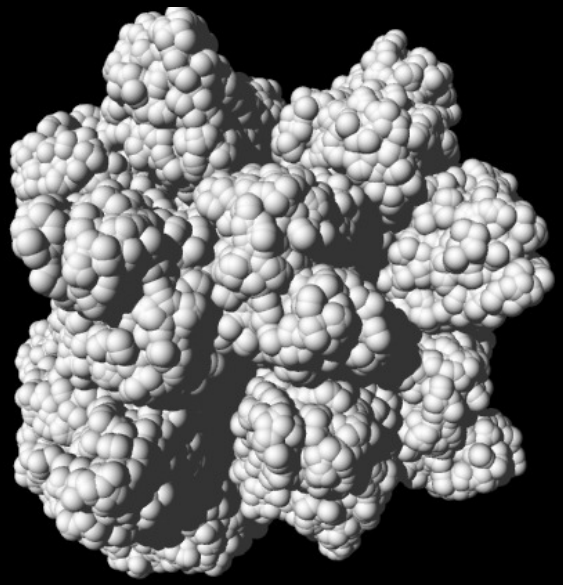
OFF-LINE RENDERING OF PDB DATA

Data :

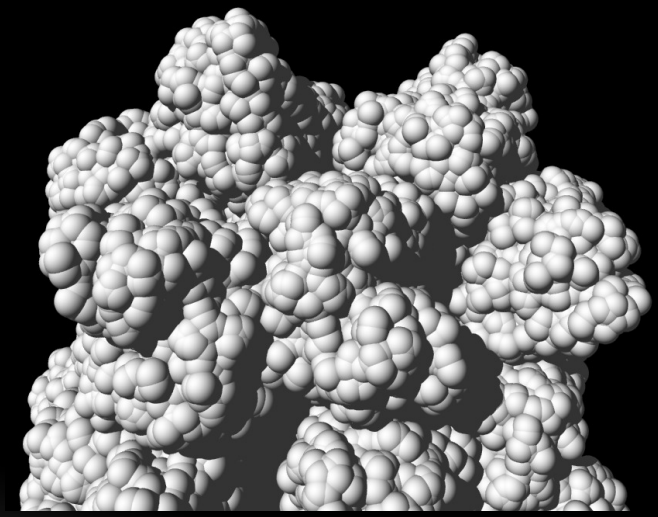
http://www.rcsb.org/pdb/explore.do?structureId=1mt5

Only Atom, 1M

Renderig point primitives with Lambert shader and ray casting



Almost same image



Result on Intel PC

Result on K

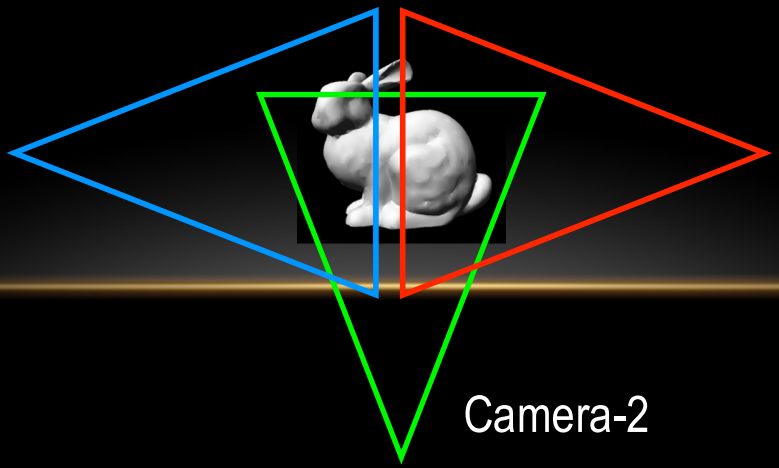
MULTI CAMERA SCENARIO

- Script allows us to describe more than one scenario

Camera-2.jpg

Camera-0.jpg

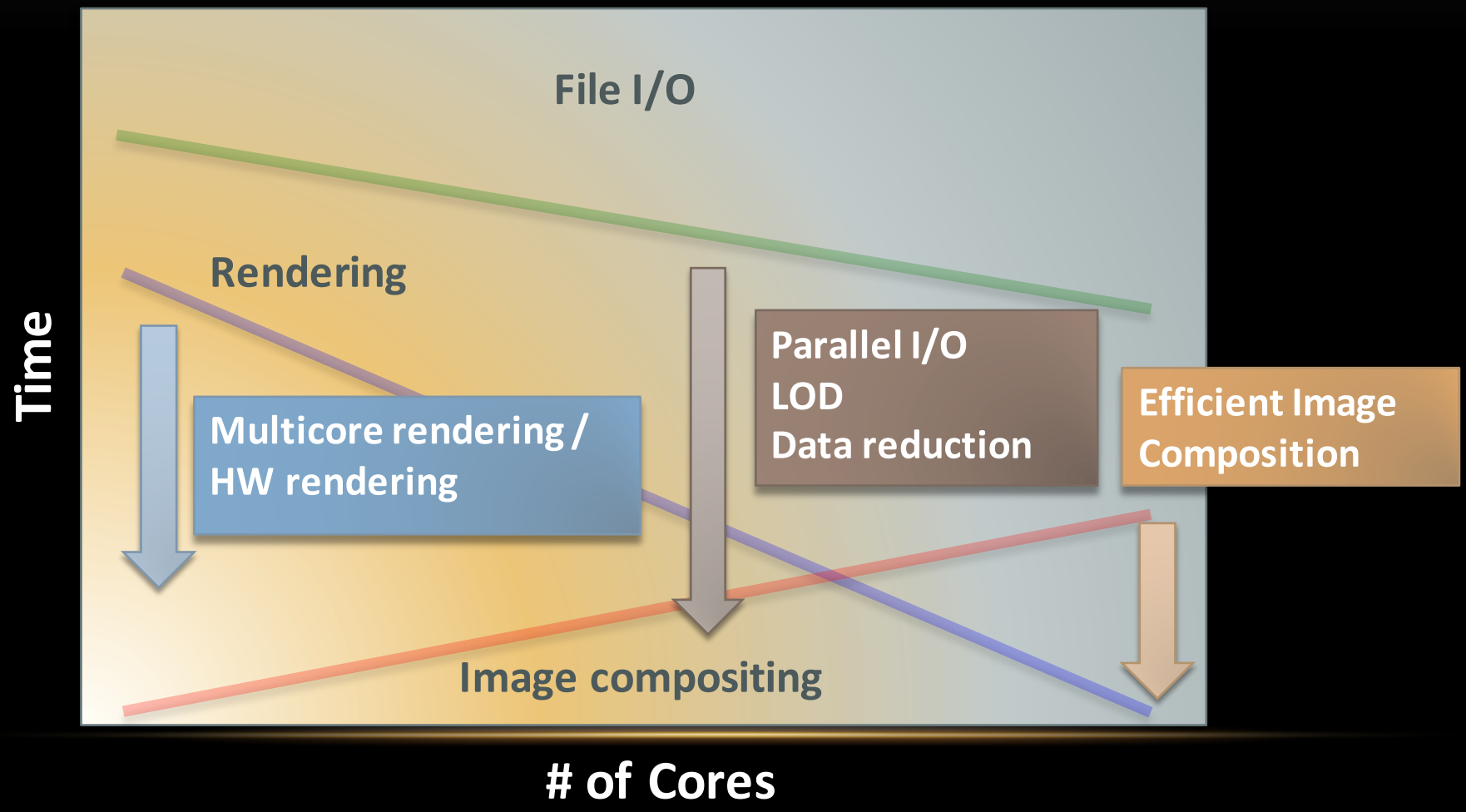
Camera-1.jpg



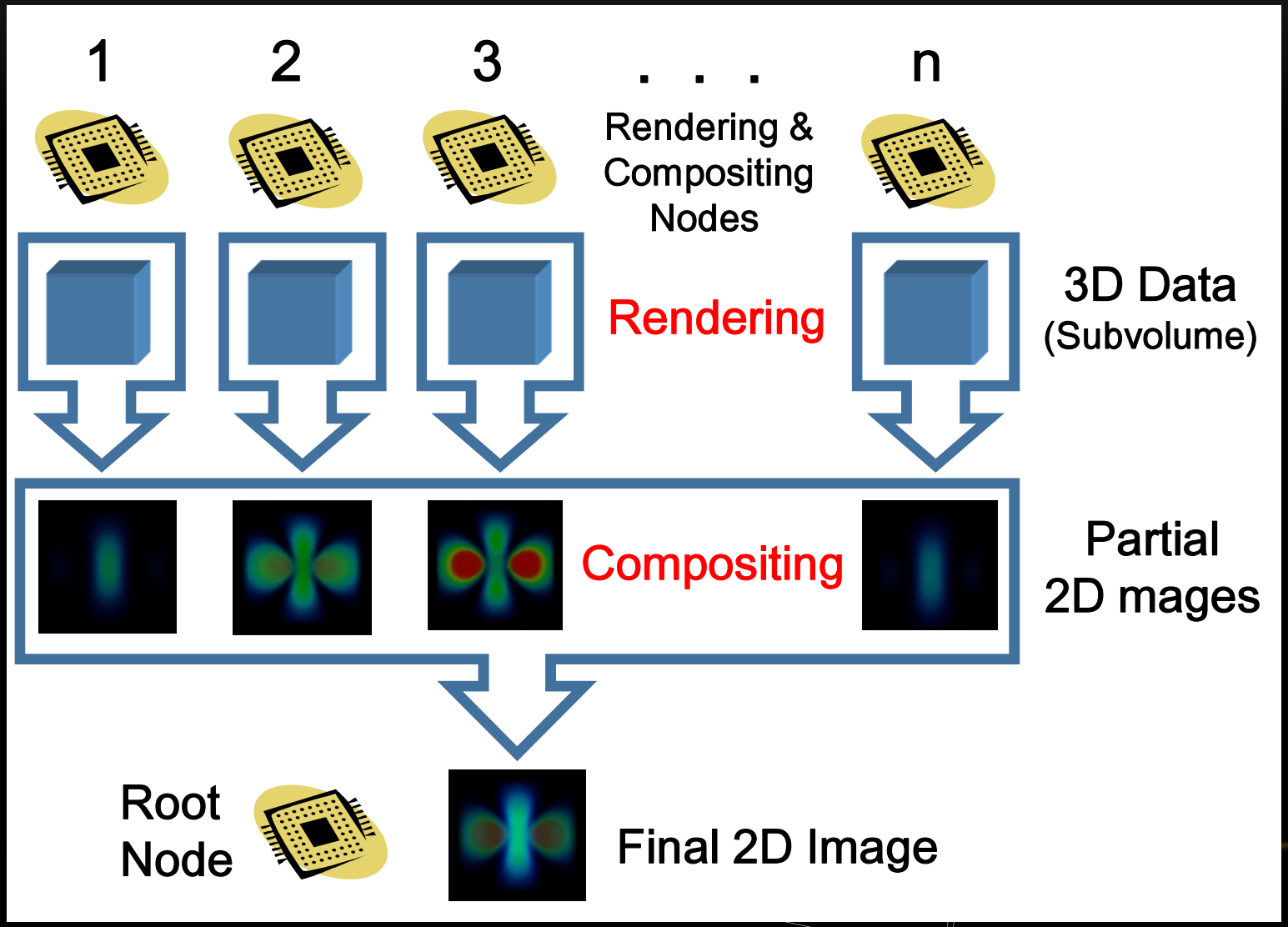
DEVELOPMENT IN THE FUTURE

- High-Performance Vis.
  - In-Situ
  - Vis for Parallel in Time method
  - Vis for Capacity Computing
- Analytics
  - Parallel Coordinate
  - Fiber
- User-Interface
  - Different view like a stand alone application

STRATEGY FOR INTERACTIVITY AND SCALABILITY

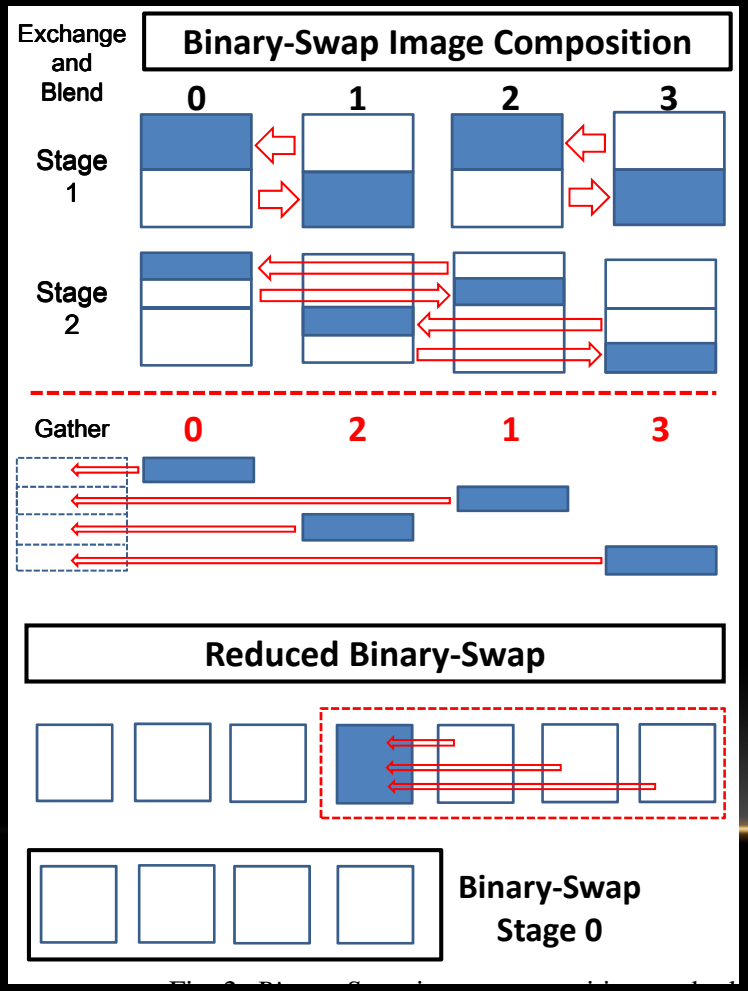


PARALLEL SORT-LAST RENDERING

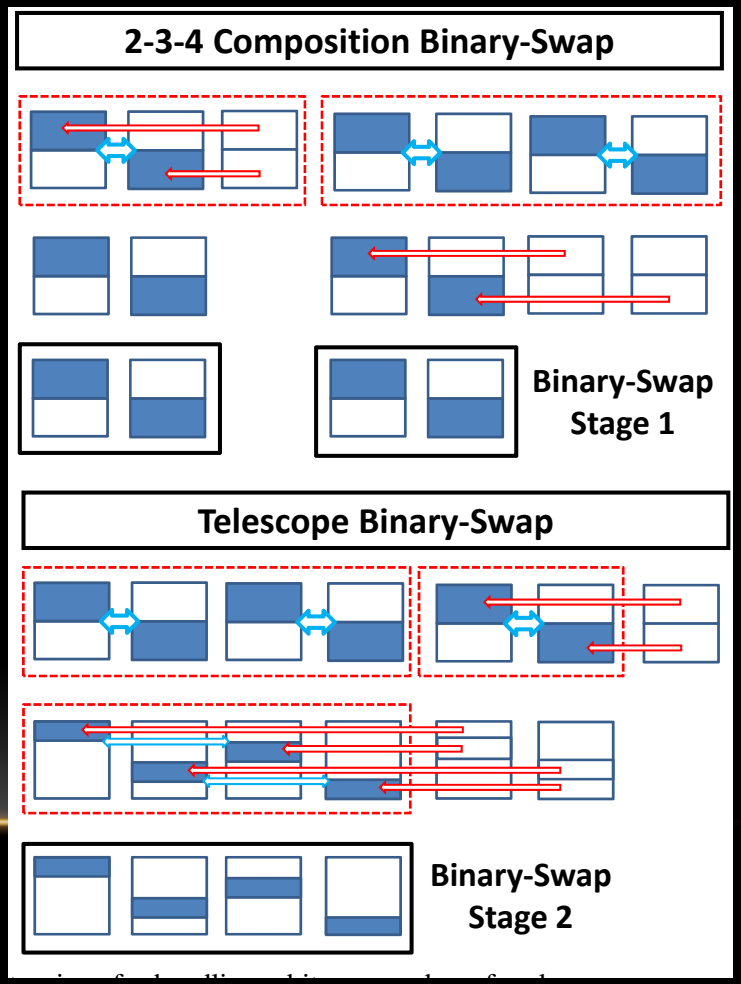


BINARY-SWAP BASED METHOD

Keep busy in all nodes
Scale out, but only for power of two



Preconditioned BS to avoid the limitation of power of two



Thank you !