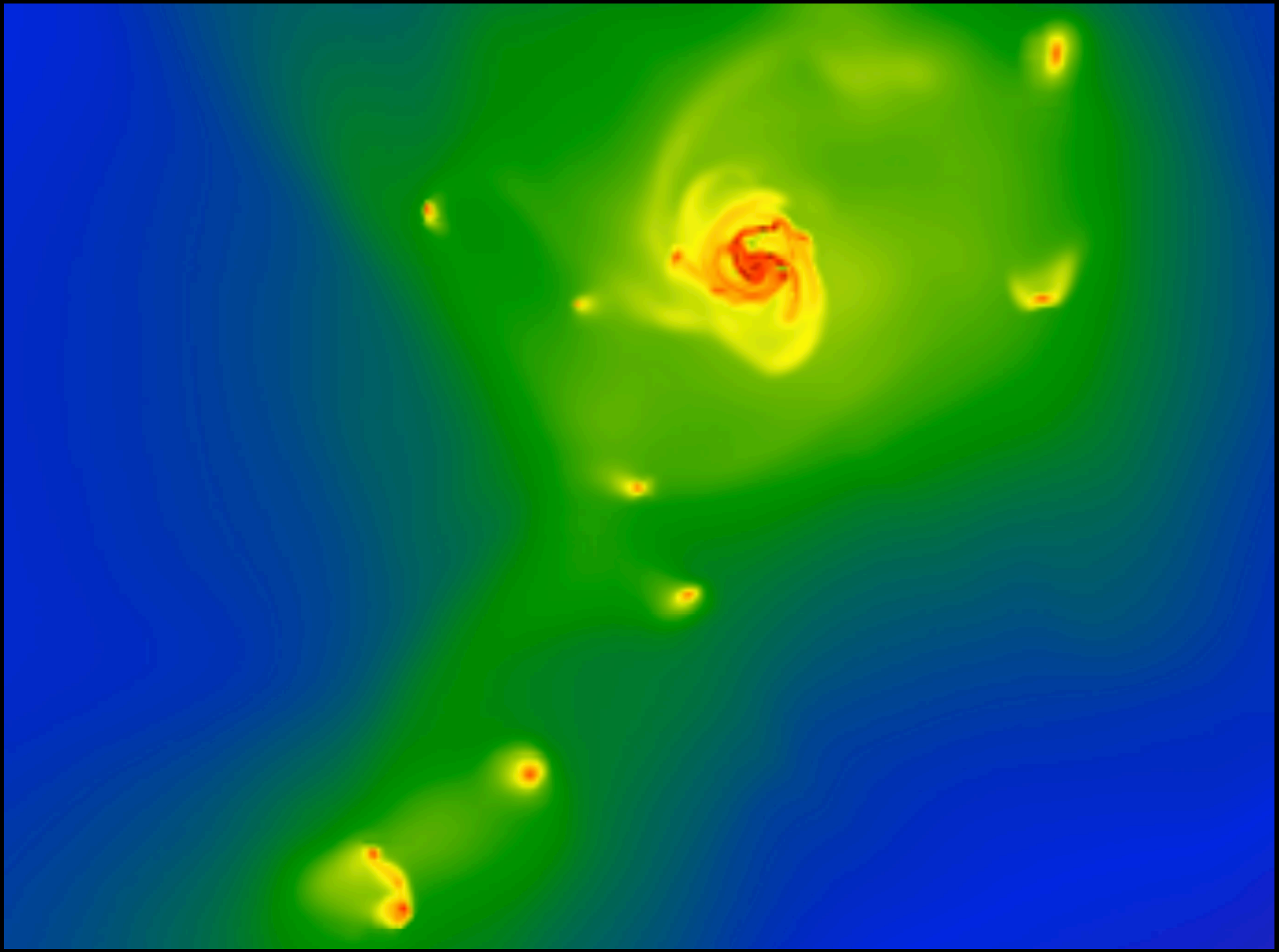
Volume Rendering

Learn yt workshop 2016
Nathan Goldbaum

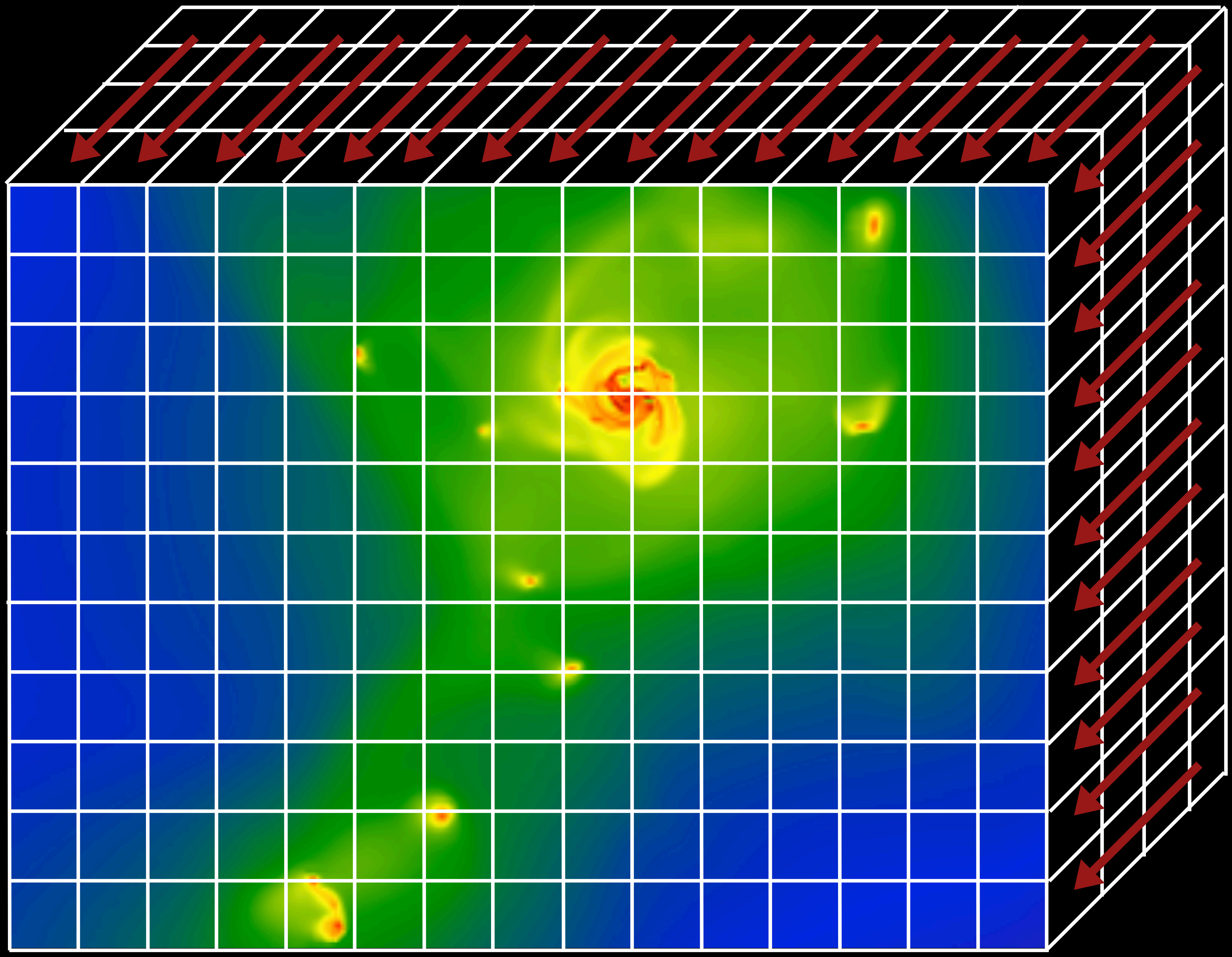
(some slides adapted from Cameron Hummels'
2012 volume rendering tutorial)

What is Volume Rendering?



http://yt-project.org/doc/visualizing/volume_rendering.html

What is Volume Rendering?

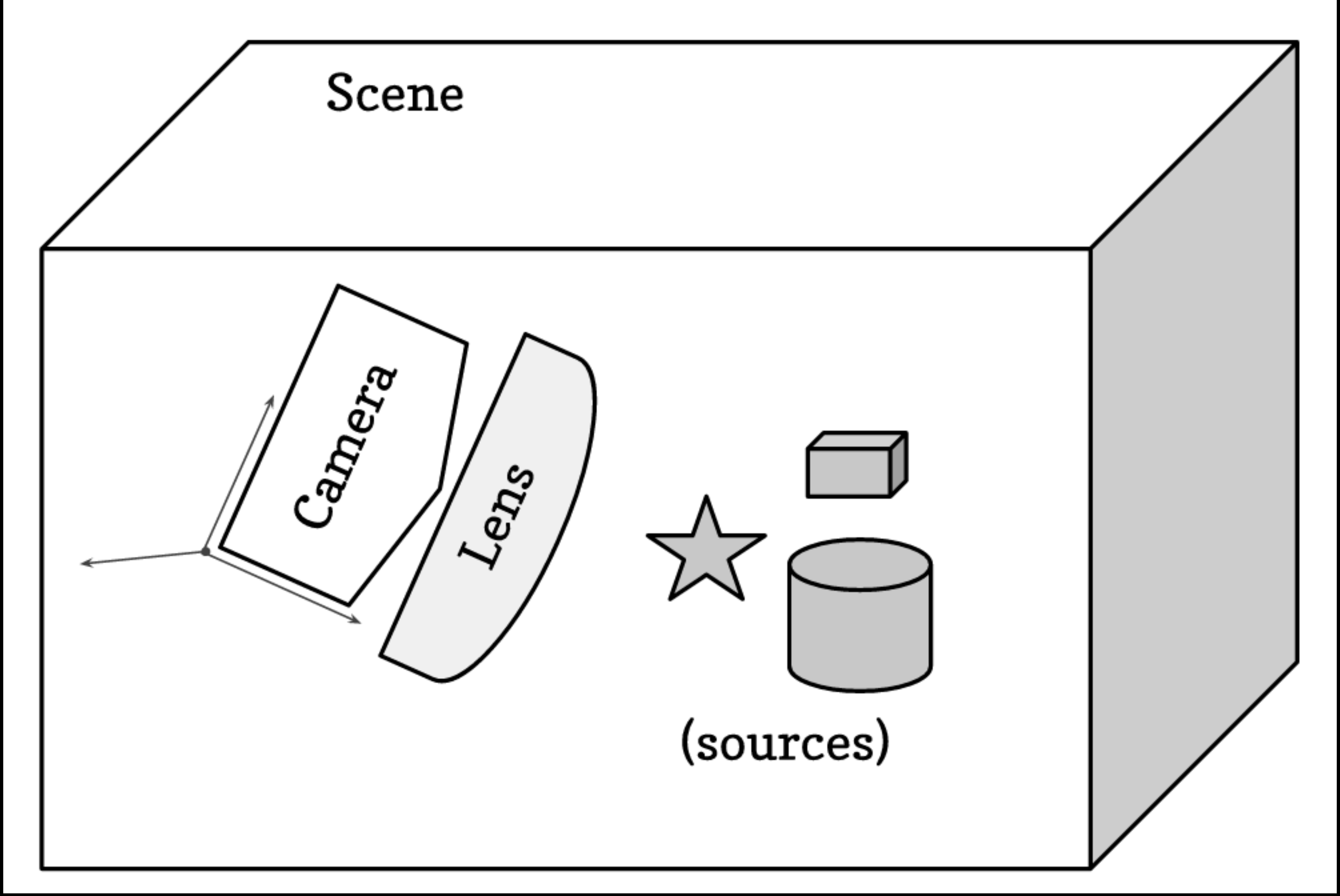


$$\frac{dI_\nu}{ds} = -\kappa_\nu I_\nu + \epsilon_\nu$$

Caveats

- Only grid data are fully supported
- Octree, particle data currently only support off-axis projections
- Unstructured mesh data can produce “hard-surface” VR
- OpenMP parallel
    - Clang (OSX) doesn’t support OpenMP until very recently. Must use gcc toolchain or recent clang toolchain to get OpenMP speedups on OSX
- New VR interface in yt 3.3

High Level Ideas



Volume Rendering Components

- Scene - container object describing a volume and its contents
    - Sources - objects to be rendered
        - VolumeSource - simulation volume tied to a dataset
            - TransferFunction - mapping of simulation field values to color, brightness, and transparency
        - OpaqueSource - Opaque structures like lines, dots, etc.
    - Annotations - Annotated structures like grid cells, simulation boundaries, etc.
    - Camera - object for rendering; consists of a location, focus, orientation, and resolution
        - Lens - object describing method for distributing rays through Sources

http://yt-project.org/doc/visualizing/volume_rendering.html#volume-rendering-components

Simple volume renderings

```
$ yt pastebin_grab 6850 > simple_vr.py
```

volume_render: create and save a volume rendering in one of line of code

```
im, sc = yt.volume_render(
    data_source,
    field,
    fname,
    sigma_clip,
    lens_type)
```

required:
data_source: dataset, data object
field: field to render

optional:
fname: filename of rendering
sigma_clip: adjust contrast
lens_type: ray projection method

http://yt-project.org/doc/reference/api/generated/yt.visualization.volume_rendering.volume_rendering.volume_render.html

The Scene Object

```
$ yt pastebin_grab 6851 > scene.py
```

create_scene: set up scene object for further customization

Scene objects contain sources of emission and absorption

```
sc = yt.create_scene(
         data_source,
         field,
         lens_type)
```

required:
data_source: dataset, data object

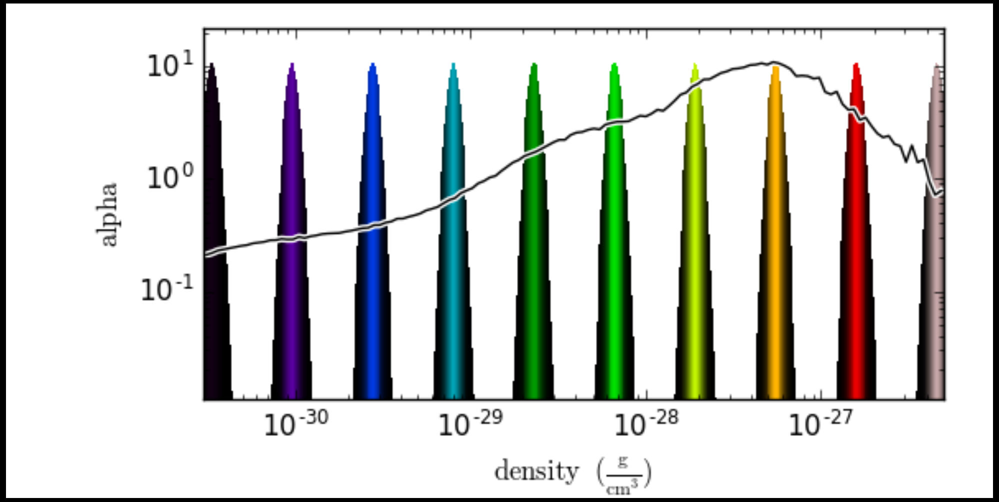
optional:
field: field to render
lens_type: ray projection method

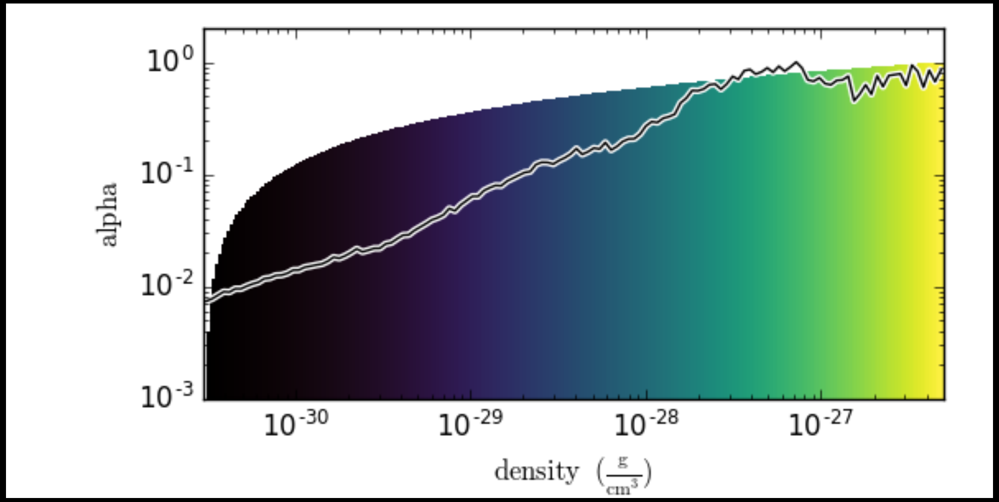
http://yt-project.org/doc/reference/api/generated/yt.visualization.volume_rendering.volume_rendering.create_scene.html

Volume Sources and Transfer Functions

Volume Source: source for emission from a yt dataset

Transfer functions determines the color and brightness of a field as a function of the field values





Building a Transfer Function

```
$ yt pastebin_grab 6853 > transfer_function_helper.py
```

```
$ yt pastebin_grab 6854 > transfer_function_gray.py
```

TransferFunctionHelper helper functions:

set_bounds
set_log
build_transfer_function
gray_opacity
plot

http://yt-project.org/doc/visualizing/volume_rendering.html#transferfunctionhelper
http://yt-project.org/doc/visualizing/transfer_function_helper.html#transfer-function-helper-tutorial

Ghost Zones?

```
$ yt pastebin_grab 6855 > ghost_zones_vr.py
```

Generating ghost zones can be slow, so it is turned off by default

Turning it on can eliminate artifacts

Customizing Transfer Functions

```
$ yt pastebin_grab 6858 > transfer_function_add_layers.py
```

```
$ yt pastebin_grab 6859 > transfer_function_sample_colormap.py
```

```
$ yt pastebin_grab 6860 > transfer_function_add_gaussian.py
```

```
$ yt pastebin_grab 6861 > transfer_function_map_to_colormap.py
```

Annotating Volume Renderings

```
$ yt pastebin_grab 6862 > box_and_grids.py
```

```
$ yt pastebin_grab 6863 > vol_annotated.py
```

```
$ yt pastebin_grab 6865 > vol_points.py
```

The Camera

- Camera object

  position: position of the camera in the scene

  width: “width” of the camera (plane parallel)

  resolution: resolution of the render (# of rays)

  focus: The camera’s focus (where is it pointed)

  north_vector: The “up” direction in the image

  lens: Which camera projection to use

http://yt-project.org/doc/visualizing/volume_rendering.html#camera

Position and Orientation

- Moving and rotating the camera

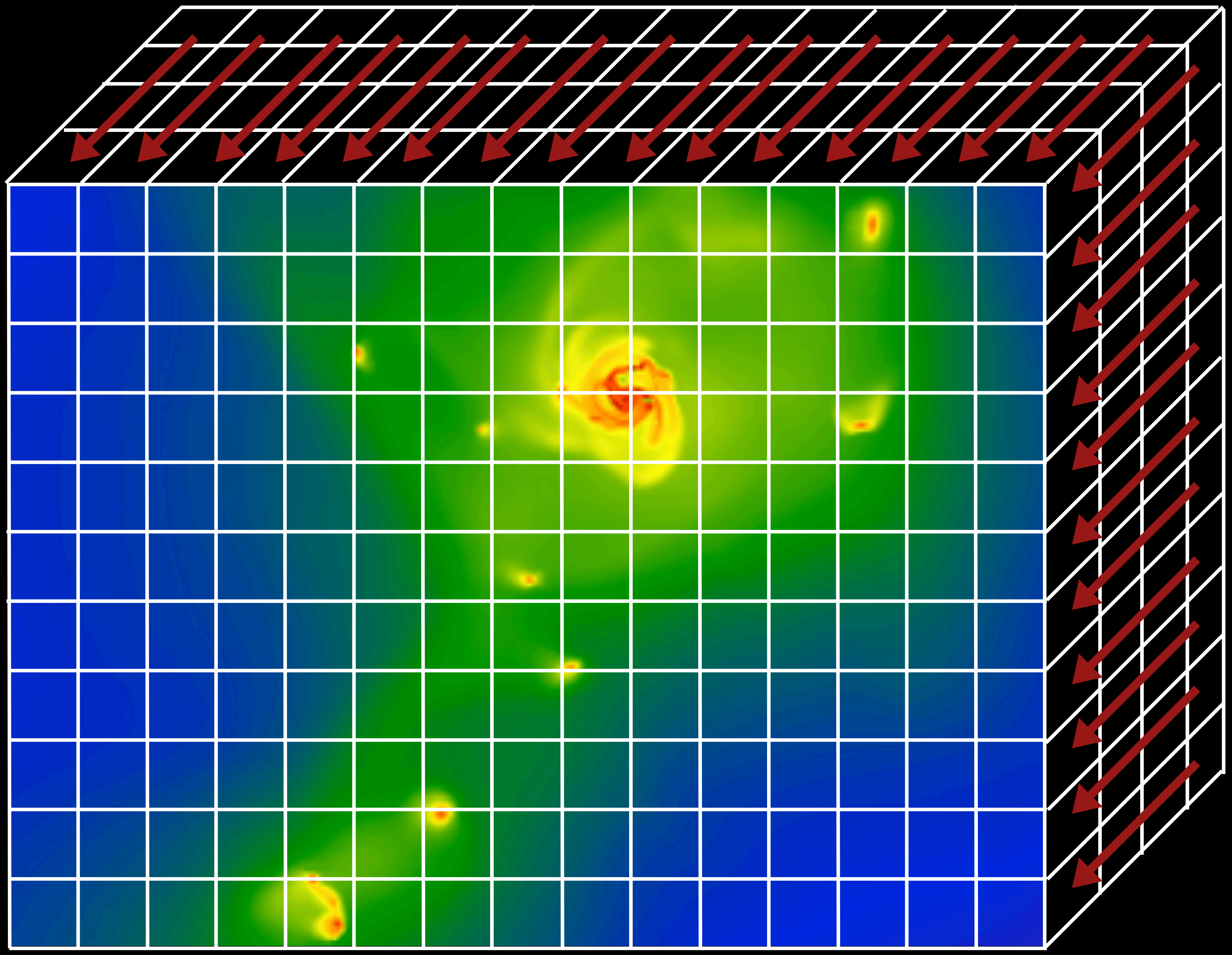
  pitch, yaw, roll, rotate, iter_rotate

  zoom, iter_zoom

  set_position, iter_move

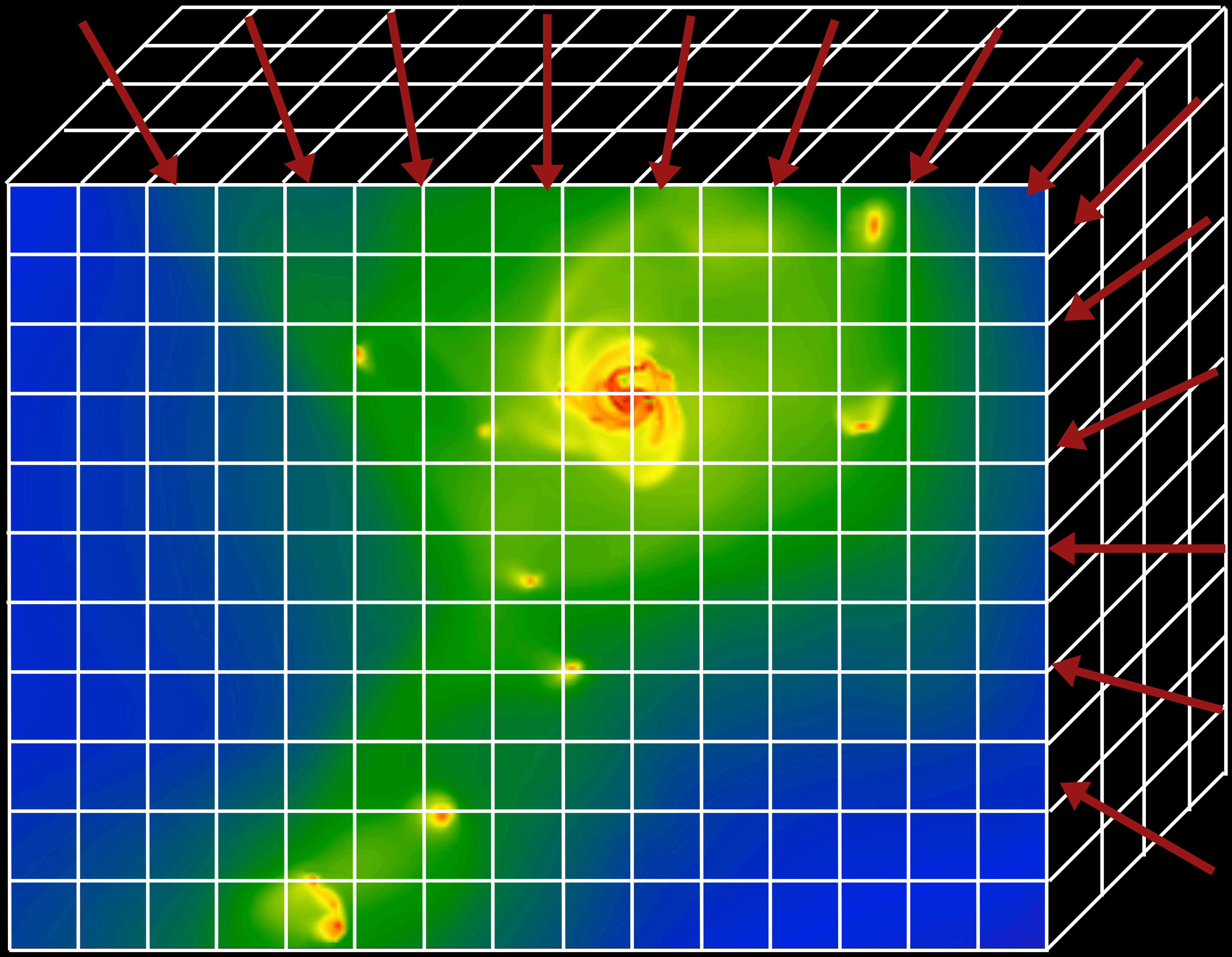
```
$ yt pastebin_grab 6866 > camera_movement.py
```

Lenses



Plane-parallel

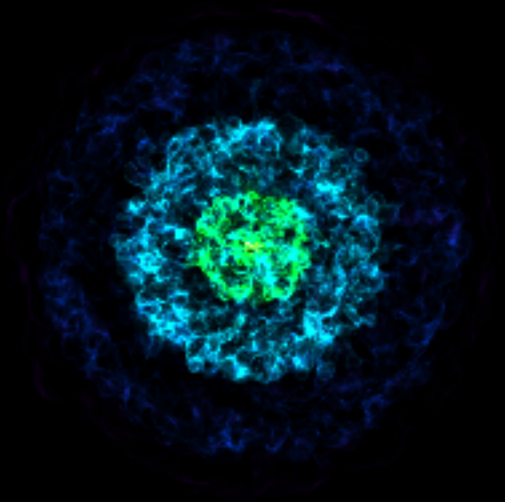
Lenses



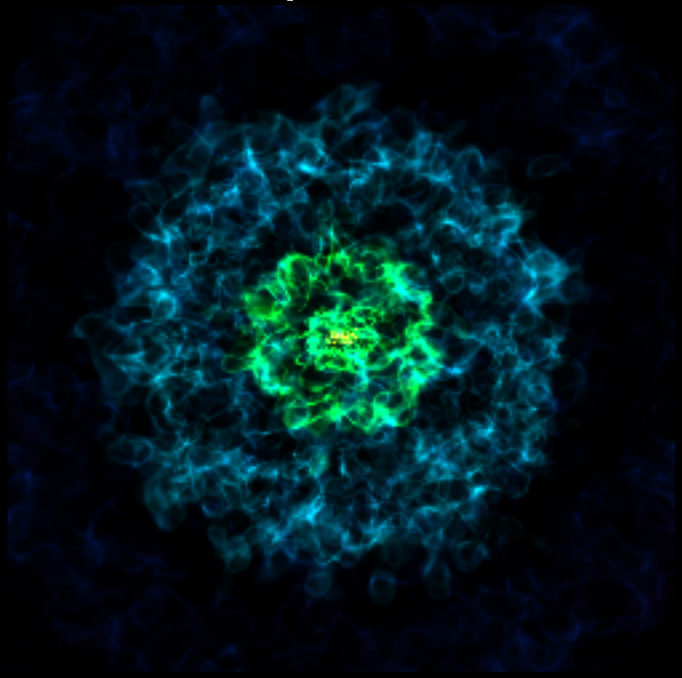
Perspective

Lenses

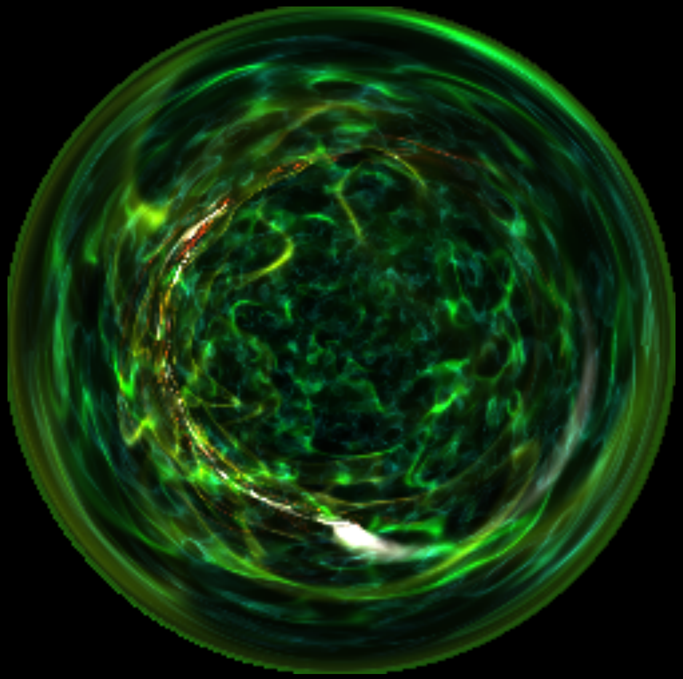
Plane parallel



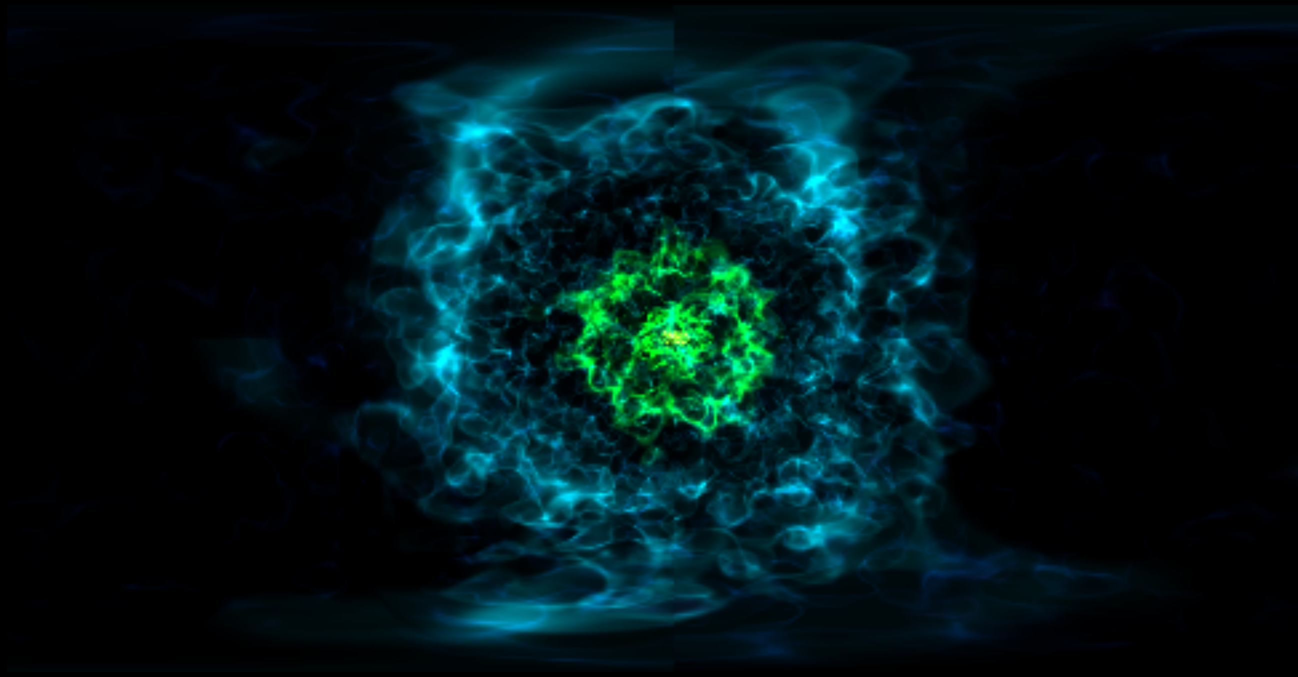
Perspective



Fisheye

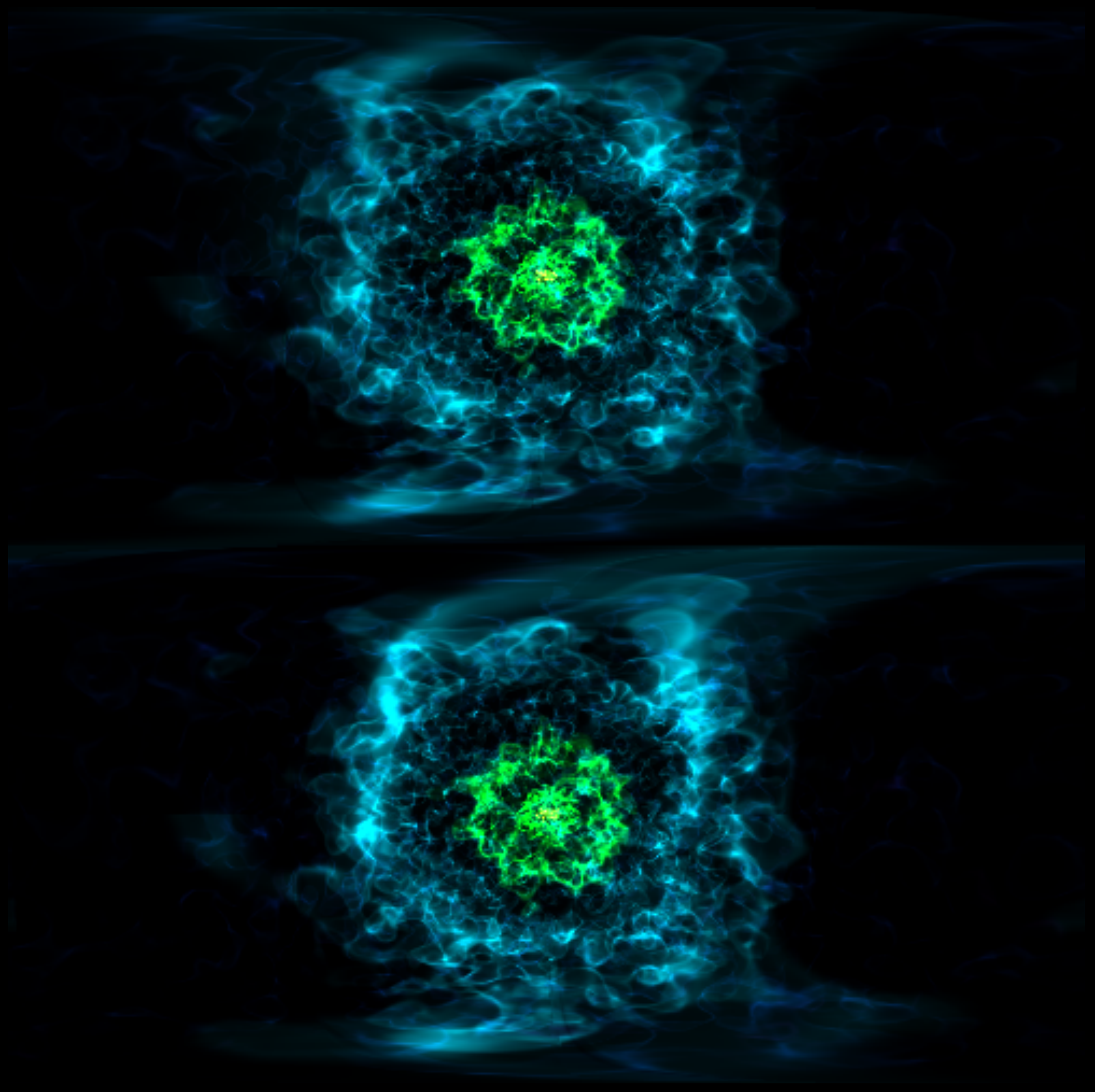


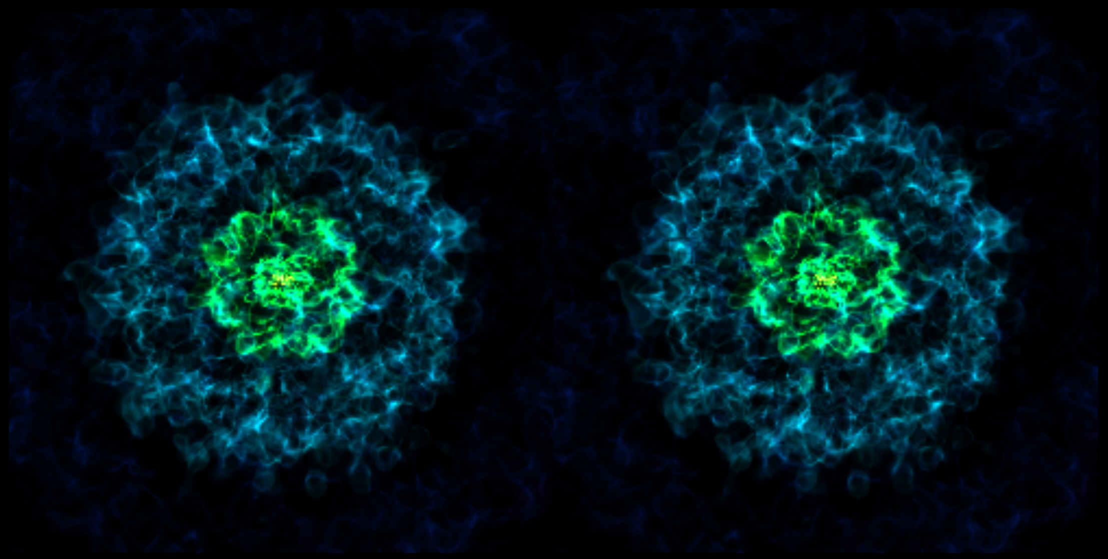
Spherical



http://yt-project.org/doc/cookbook/complex_plots.html#various-lens-types-for-volume-rendering

Stereo lenses too!





https://www.youtube.com/watch?v=ZYWY53X7UQE

Real-time interactive data visualization

- Prerequisites:
  - glfw3, cyglfw3, pyOpenGL
- yt.interactive_render()