Python, Data Visualization, and Astronomy

Dr. Brian R. Kent
National Radio Astronomy Observatory
June 2017

Overview

- History of Python
- Why use Python in Astronomy
- Short Introduction to Revision Control
- Some of my research - 3D rendering of Astronomical Data
- Resources to explore further
- Conferences (in-person and remote!)

Python

- High level programming language that is scriptable and supports many programming styles.
- Created in the late 80s/early 90s by Guido van Rossum, named after *Monty Python*

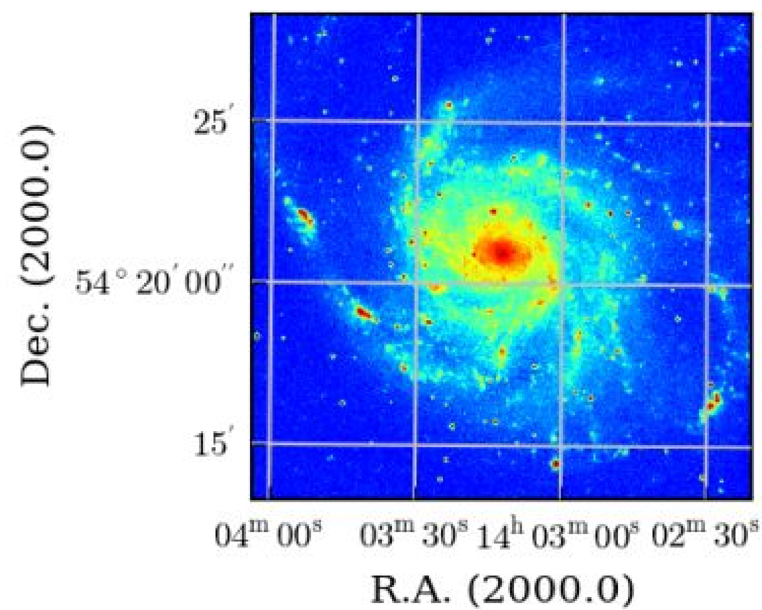
- The **Python Standard Library** can be used to accomplish a wide variety of tasks - always check there first!

tiobe.com - Usage of Languages

Jun 2017	Jun 2016	Change	Programming Language	Ratings	Change
1	1		Java	14.493%	-6.30%
2	2		C	6.848%	-5.53%
3	3		C++	5.723%	-0.48%
4	4		Python	4.333%	+0.43%
5	5		C#	3.530%	-0.26%
6	9	^	Visual Basic .NET	3.111%	+0.76%
7	7		JavaScript	3.025%	+0.44%

Python and Astronomy

- With iPython, it is used as the command line shell /interpreter for CASA
- Used for the framework for VLA/GBT/ALMA pipelines
- Can interface with AIPS via Obit
- Can interface with IRAF via PyRAF
- Used in PRESTO - pulsar data reduction
- Many modules and libraries available - matplotlib, Kapteyn, astropy, etc.



Popular Python Resources

iPython - more user friendly shell

Astropy - great for data import, manipulation, catalog queries

Matplotlib - general purpose plotting tool

Scipy - numpy and fitting routines (some overlap with astropy…)

Kapteyn - Good for mapping projections

AstroML - Machine Learning
http://www.astroml.org/index.html

PANDAS - Data Analysis Library
http://pandas.pydata.org/

Where to start?

- https://safe.nrao.edu/wiki/bin/view/Main/PythonResources

- http://www.astropython.org/resources

Example: Create a noisy spectrum and fit a Gaussian

http://cars9.uchicago.edu/software/python/mpfit.html

```
import pylab
import os
import sys
import numpy
from math import sqrt
import mpfit
```

Levenberg-Marquardt least-squares minimization

Usage:

```
m = mpfit.mpfit( myfunct, p0, functkw=fa )
```

p0 - initial guess for parameters

myfunct - what function do I want to fit

functsw = dictionary of kwargs I need

$$S(\boldsymbol{\beta}) = \sum_{i=1}^{m} [y_i - f(x_i,\ \boldsymbol{\beta})]^2$$

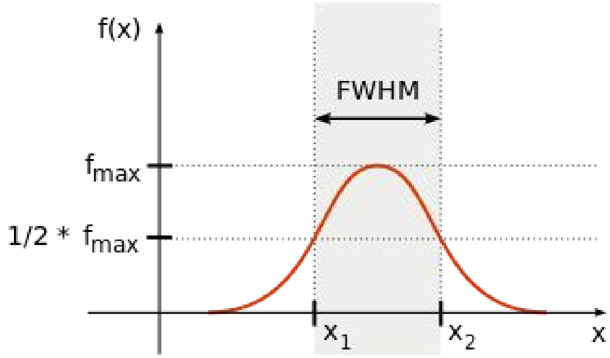
$$S(\boldsymbol{\beta}) = \sum_{i=1}^{m}[y_i - f(x_i,\ \boldsymbol{\beta})]^2$$

Functions to define

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(x-x_0)^2}{2\sigma^2}\right]$$

```
def peval(x, p):
    # The model function with parameters p
    return (1./sqrt(2*numpy.pi*p[1]**2))*numpy.exp(-(x-p[0])**2/(2*p[1]**2))

def myfunct(p, fjac=None, x=None, y=None, err=None ):
    # Function that return the weighted deviates
    model = peval(x, p)
    status = 0
    return([status, (y-model)/err])
```

$$\mathrm{FWHM} = 2\sqrt{2\ln 2}\,\sigma \approx 2.355\,\sigma.$$

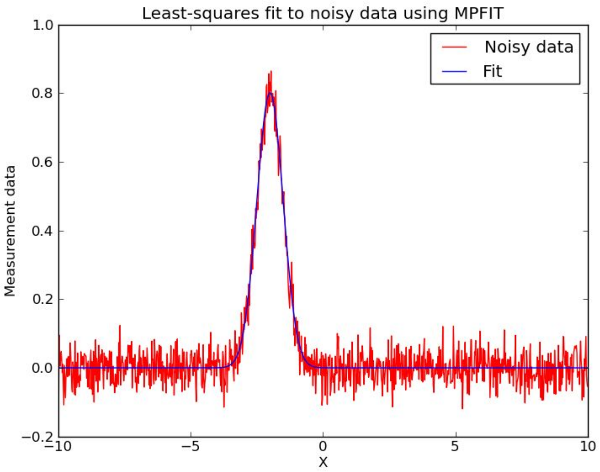


Implementation

```
# Generate model data for a Gaussian with param mu and sigma and add noise
x=numpy.arange(-10.,10.,20./1000) #1000 points
preal=[-2, .5]
y_true=peval(x,preal)
mu,sigma = 0,0.7

#Create my spectrum
y    = y_true + 0.06 * numpy.random.normal(mu,sigma, len(x) )

#Generate some pseudo random errors from a normal distribution
err = 1.0 + 0.01 * numpy.random.normal(mu,sigma, len(x) )

# Initial estimates for MPFIT
p0 = [-0.5, 0.5]
fa = {'x':x, 'y':y, 'err':err}   #Create a Dictionary
```

Execution

```
# Call MPFIT
m = mpfit.mpfit( myfunct, p0, functkw=fa )

# Plot the result with Matplotlib
pylab.clf()
pylab.plot(x,y,'r', label="Noisy data")
pylab.plot( x,peval(x,m.params), label="Fit" )
pylab.plot( x,y_true, 'g', label="True data" )
pylab.xlabel( "X" )
pylab.ylabel( "Measurement data" )
pylab.title( "Least-squares fit" )
pylab.legend()
pylab.show()
```



Revision Control

Revision Control for Software

Popular revision control systems (RCSs) include SVN, CVS, Mercurial, and git.

Tracks changes made to files - good for collaboration among teams or to see when and where changes were made to your code!

Let’s do an example with git…

https://bitbucket.org/brkent/demo

git revision control

```
mkdir demo
cd demo
git init
git remote add origin git remote add origin
https://brkent@bitbucket.org/brkent/demo.git

This will create and link your directory
to the git repository
```

git revision control

```
#Create a readme file
touch README
git add README
git commit -m "Initial README commit"
git push -u origin master

#Take a look and see what happened!
```

Back to our Gaussian Fit

Let's print out some information about our spectrum…

```
#RMS
import numpy as np
np.std(y[100:300])
m.params[0]
m.params[1]
m.perror #gives uncertainties

import math
FWHM = 2.*math.sqrt(2.*np.log10(2.)) * m.params[1]

print "FWHM: {!s}".format(FWHM)
```

$$\mathrm{FWHM} = 2\sqrt{2\ln 2}\,\sigma \approx 2.355\,\sigma.$$

Data Rates in Astronomy

- The Atacama Large Millimeter Array (ALMA) in Chile will produce:
  - over 1300 Terabytes of total data in 2014.
  - over 2700 Terabytes of total data in 2016
- The Very Large Array in New Mexico has the capability of producing a million simultaneous frequency channels.
- The Green Bank Telescope is beginning operations for multibeam spectroscopy, producing 800 MB/sec.

3D Graphics Software

MAYA

3DS MAX

blender™

CINEMA 4D

LightWave11

HOUDINI

PIXAR's RenderMan

3D Graphics, Python, and Astronomy

I use a non-traditional package called Blender to render different forms of astronomical data - catalogs, data cubes, simulations, etc.

What is Blender?

Blender is:

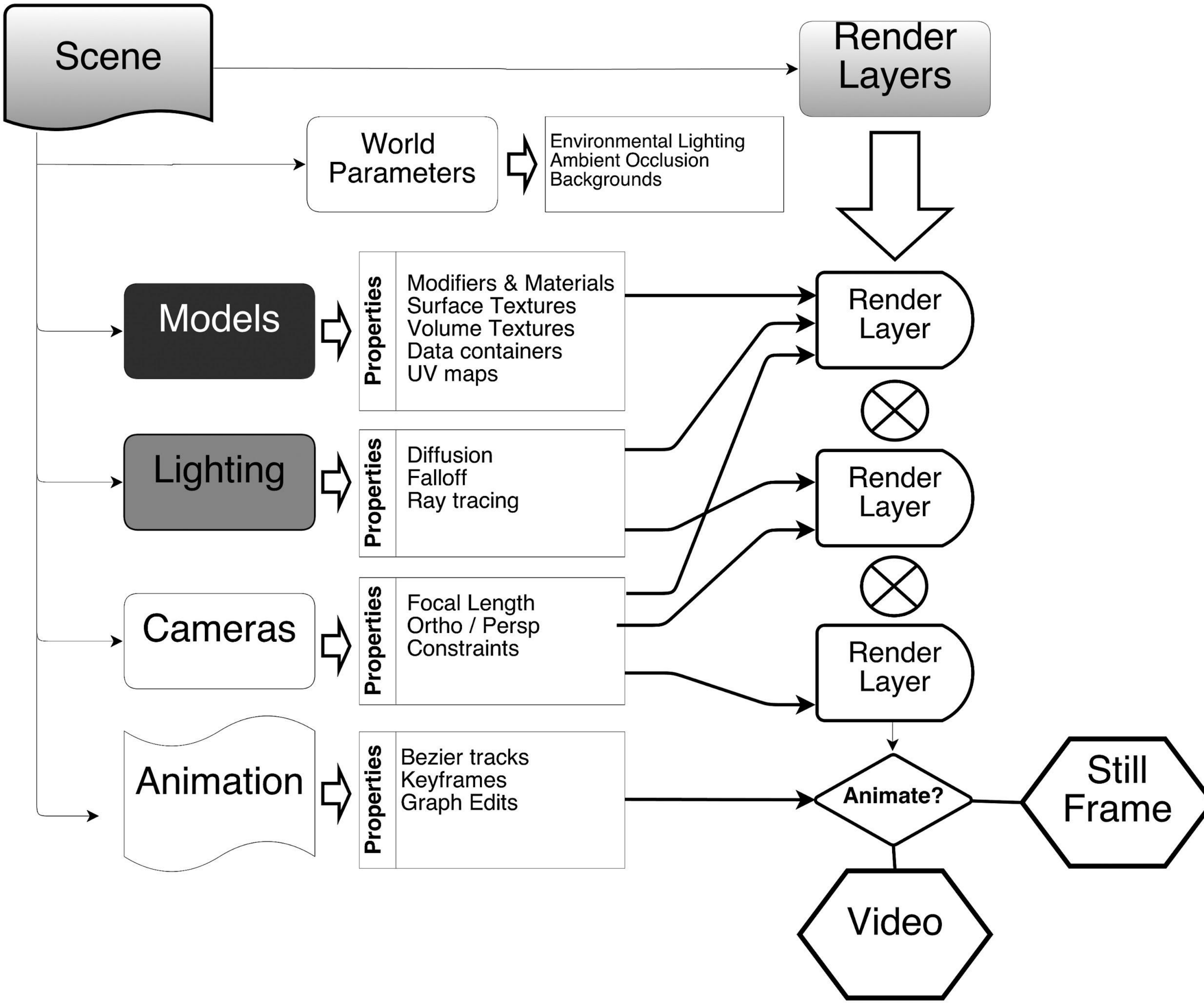
- 3D graphics software for modeling, animation, and visualization
- Open-source
- a real-time 3D viewer and GUI
- A Python scriptable interface for loading data

http://www.blender.org

Elements of 3D Graphics

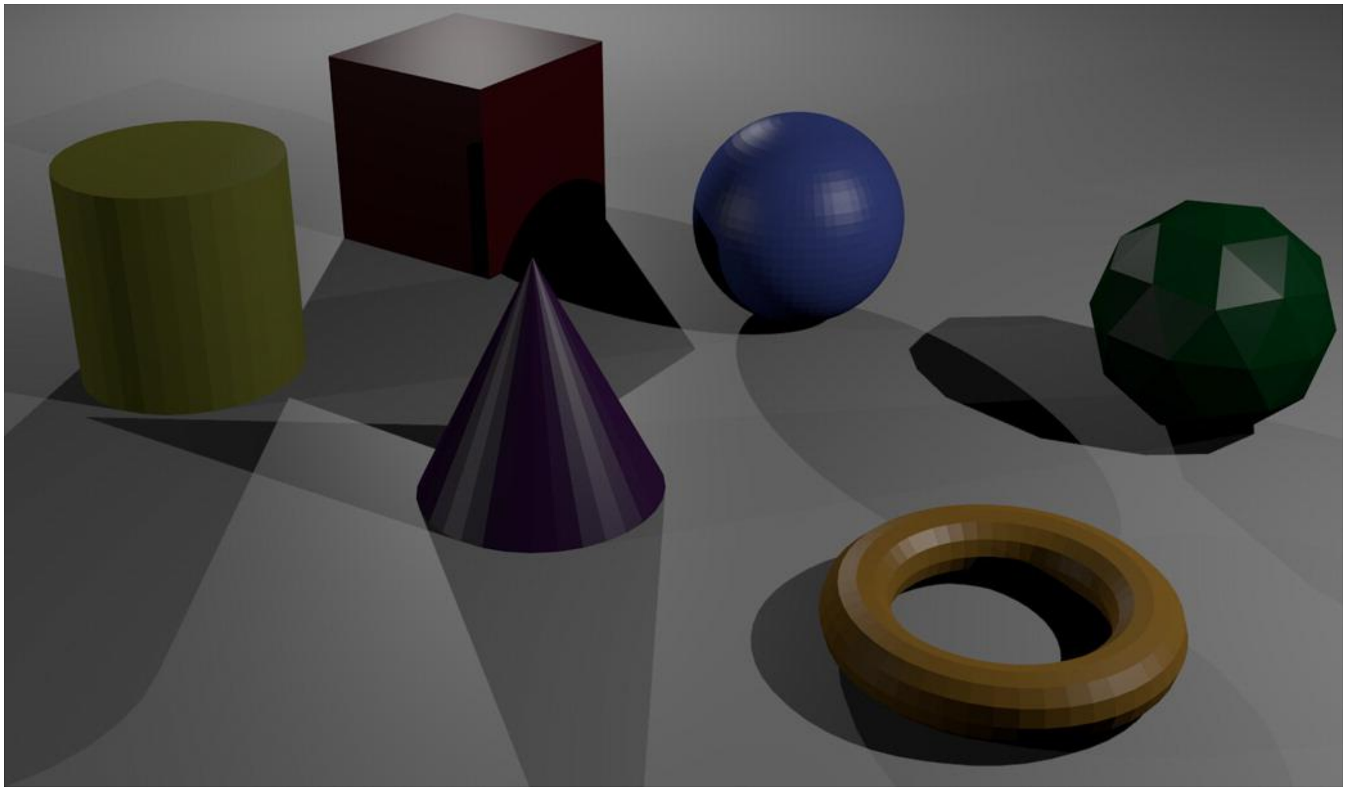
We need to consider:

- Models - physical or data containers?
- Textures - 2D, 3D, and projections?
- Lighting - illumination of data - physical or artistic
- Animation - How will the model move and change?
- Camera control - lens selection, angle, image size, and movement and tracking
- Rendering - backend engine choice
- Compositing - layering final output

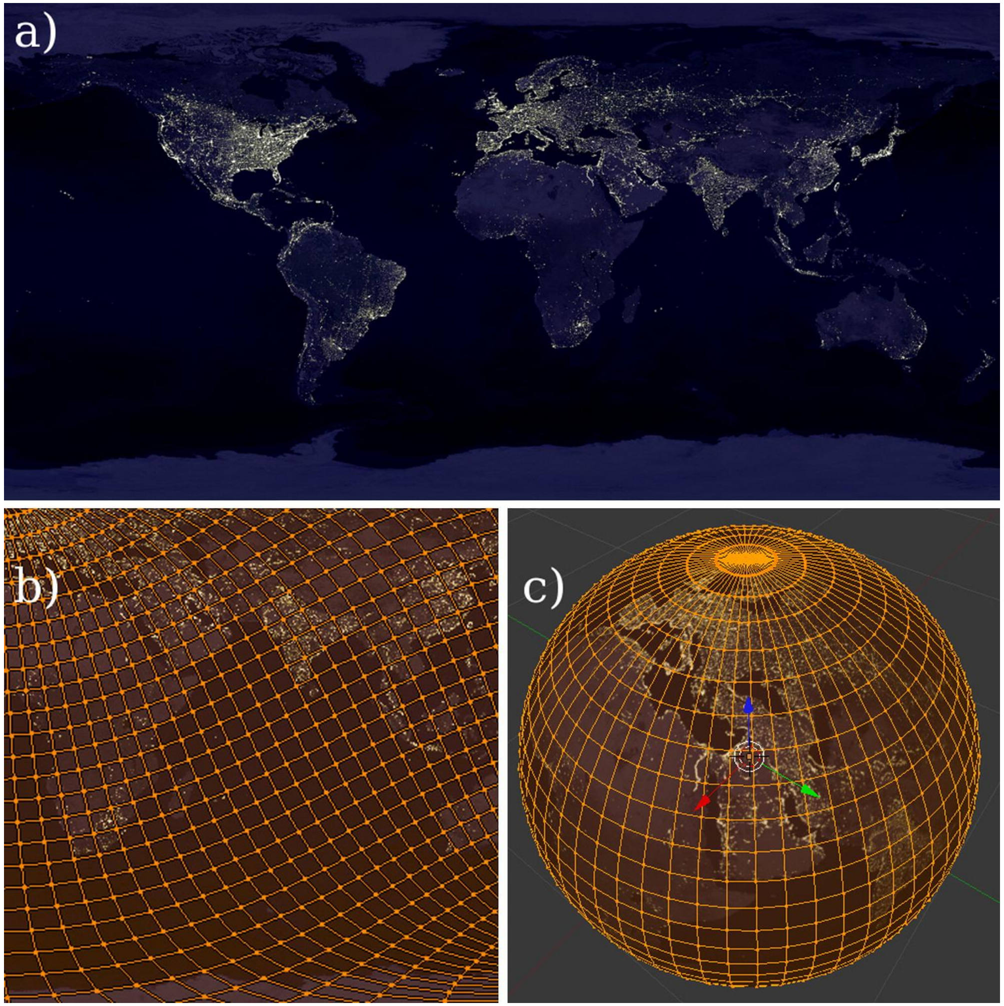


Brian Kent, NRAO

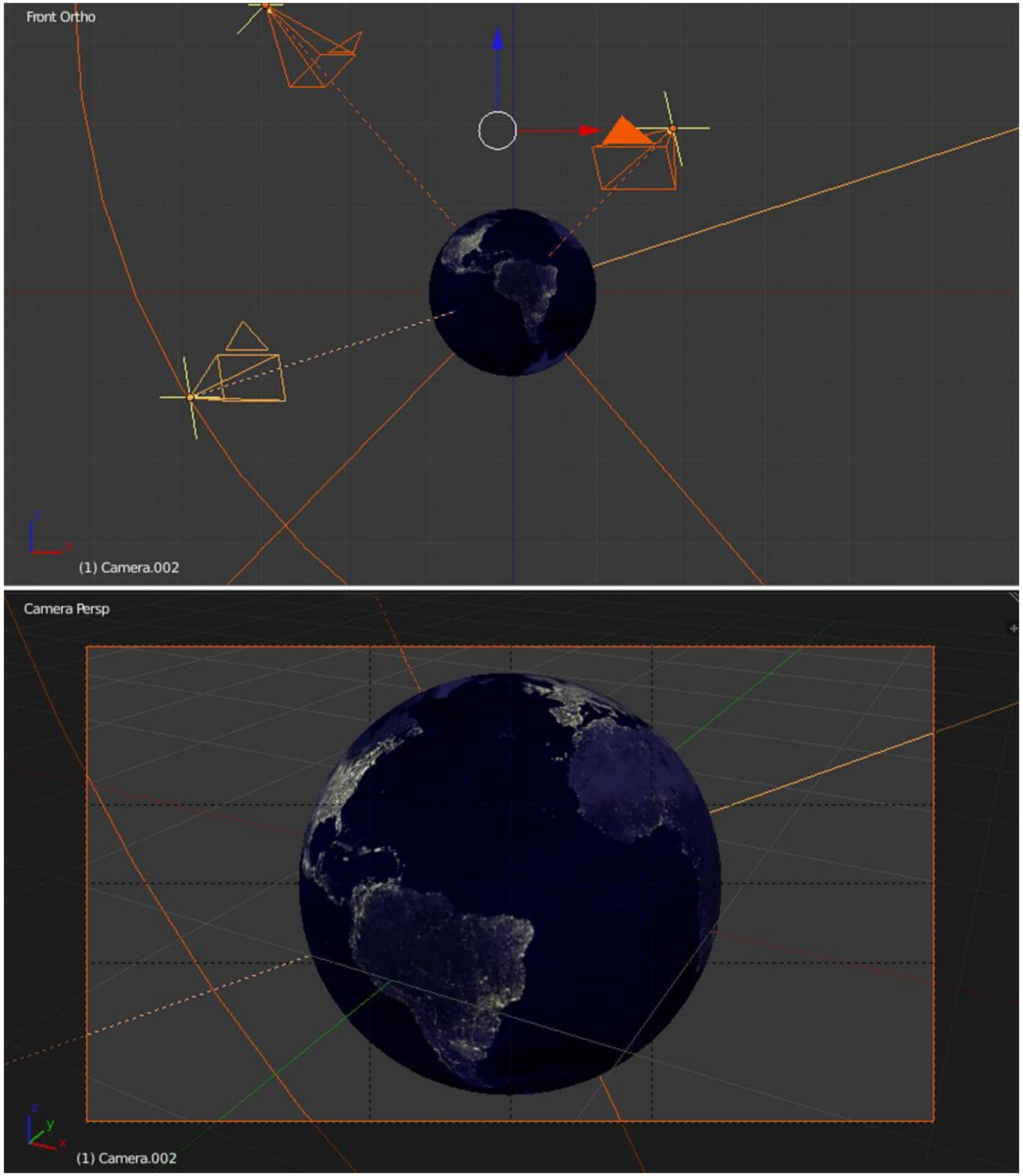
Modeling - basic shapes and containers



Texturing and Mapping

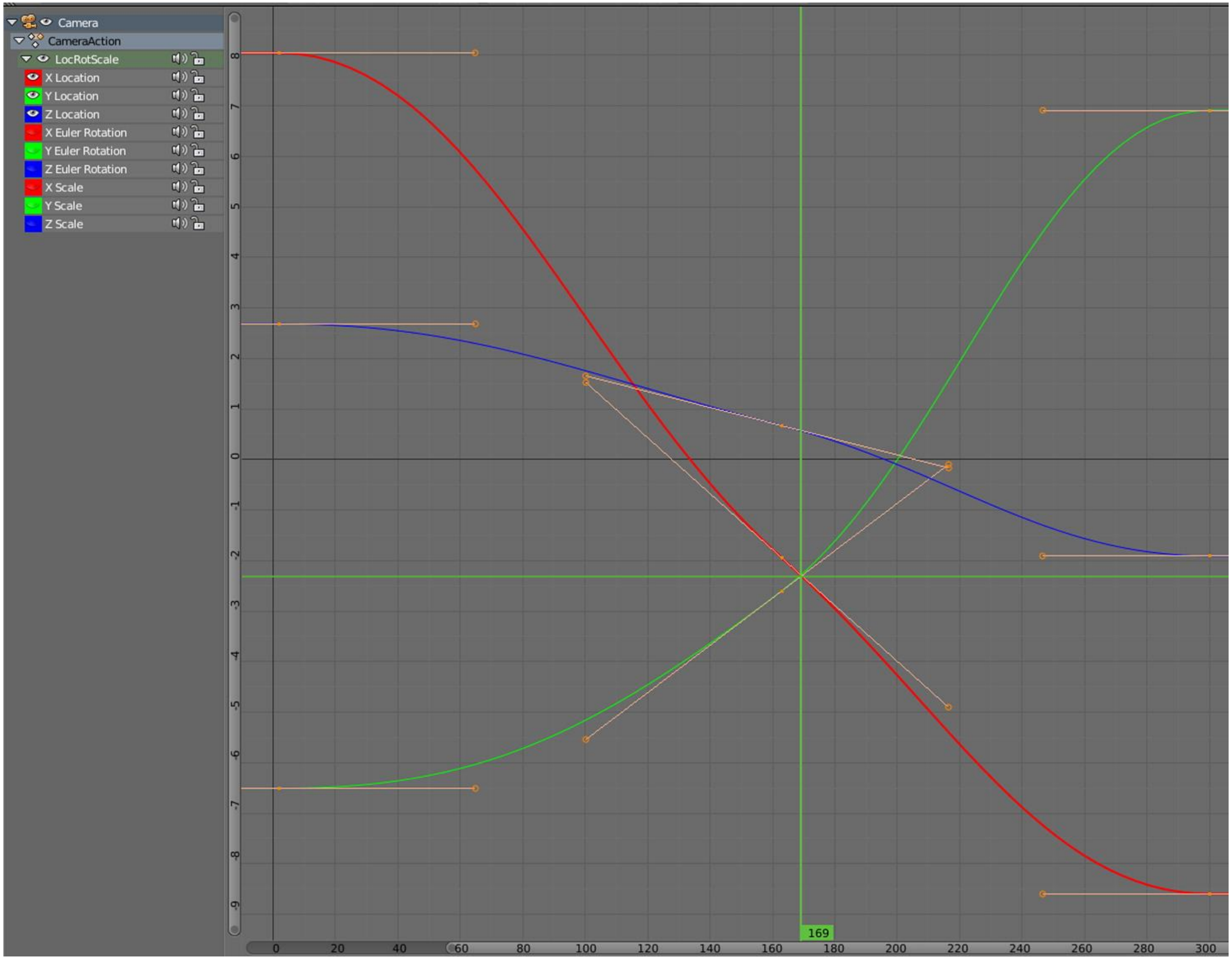


Camera Control and Movement



Brian Kent, NRAO

Animation

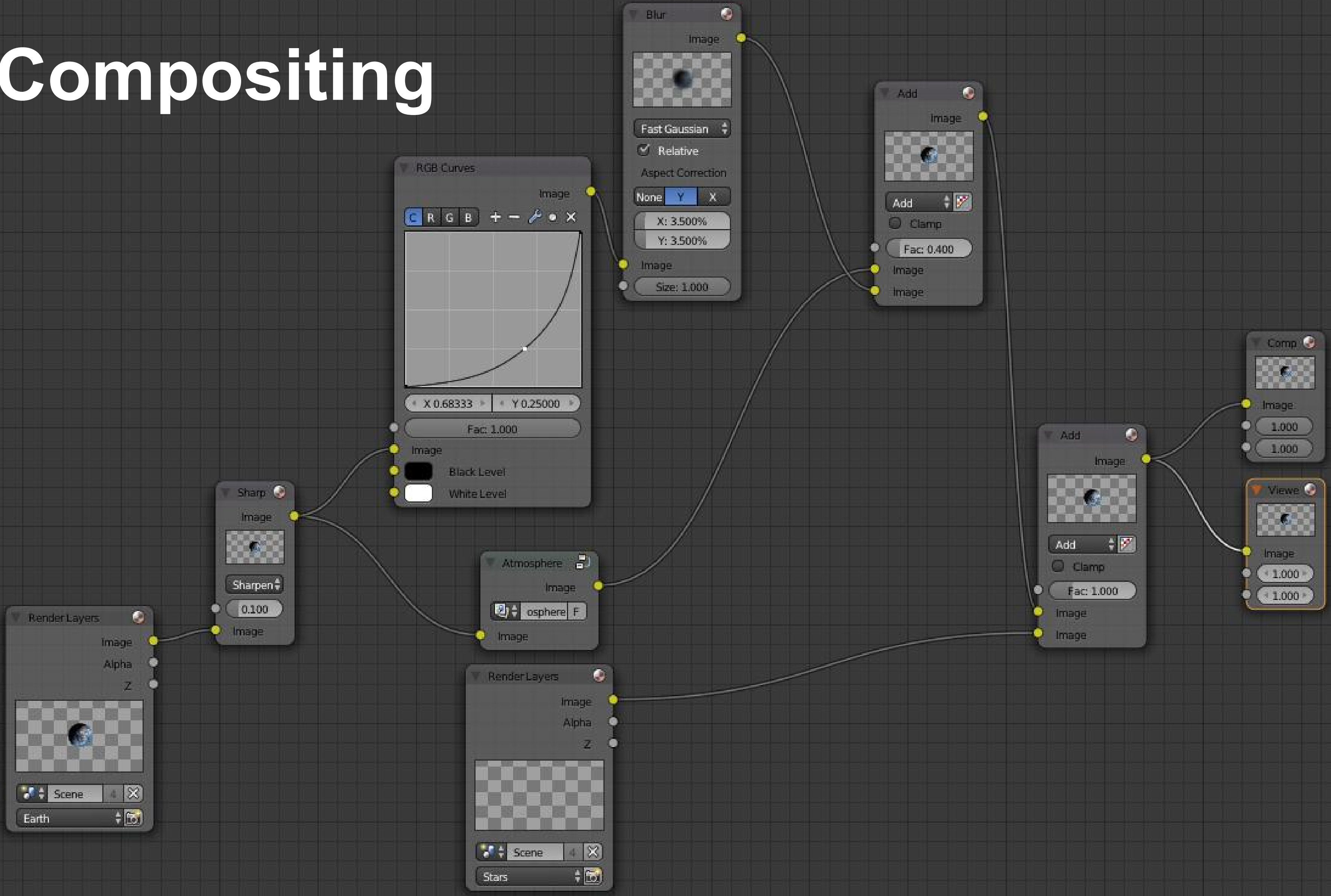


Camera
CameraAction
LocRotScale
X Location
Y Location
Z Location
X Euler Rotation
Y Euler Rotation
Z Euler Rotation
X Scale
Y Scale
Z Scale
169
0
20
40
60
80
100
120
140
160
180
200
220
240
260
280
300

Rendering Engine

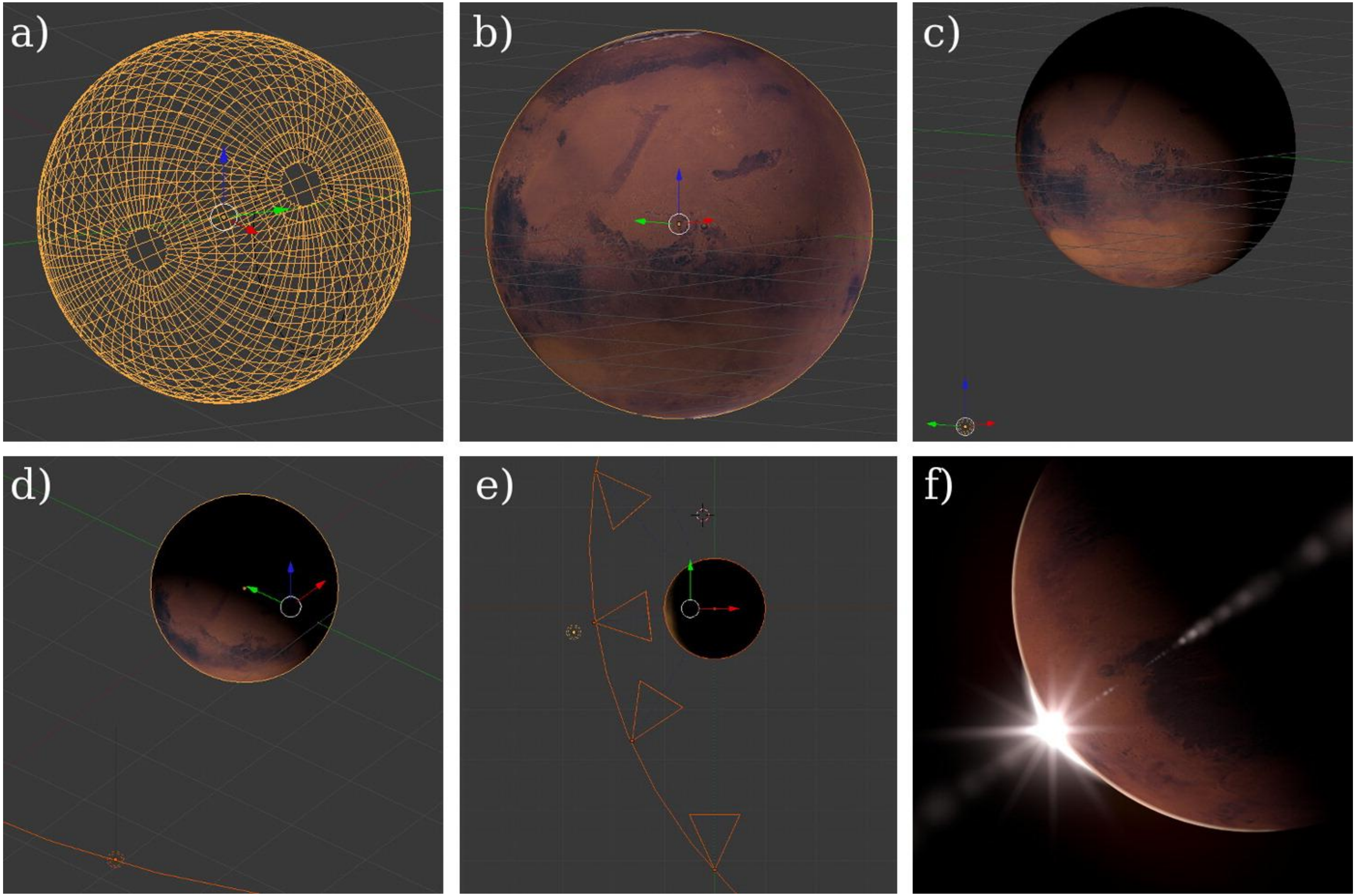
- Blender (included)
- Cycles (included)
- Yafaray (open source ray tracing engine http://www.yafaray.org/)
- Luxrender (http://www.luxrender.net/en_GB/index)
- Octane (http://render.otoy.com/)
- Renderman (http://renderman.pixar.com/view/renderman)

Compositing



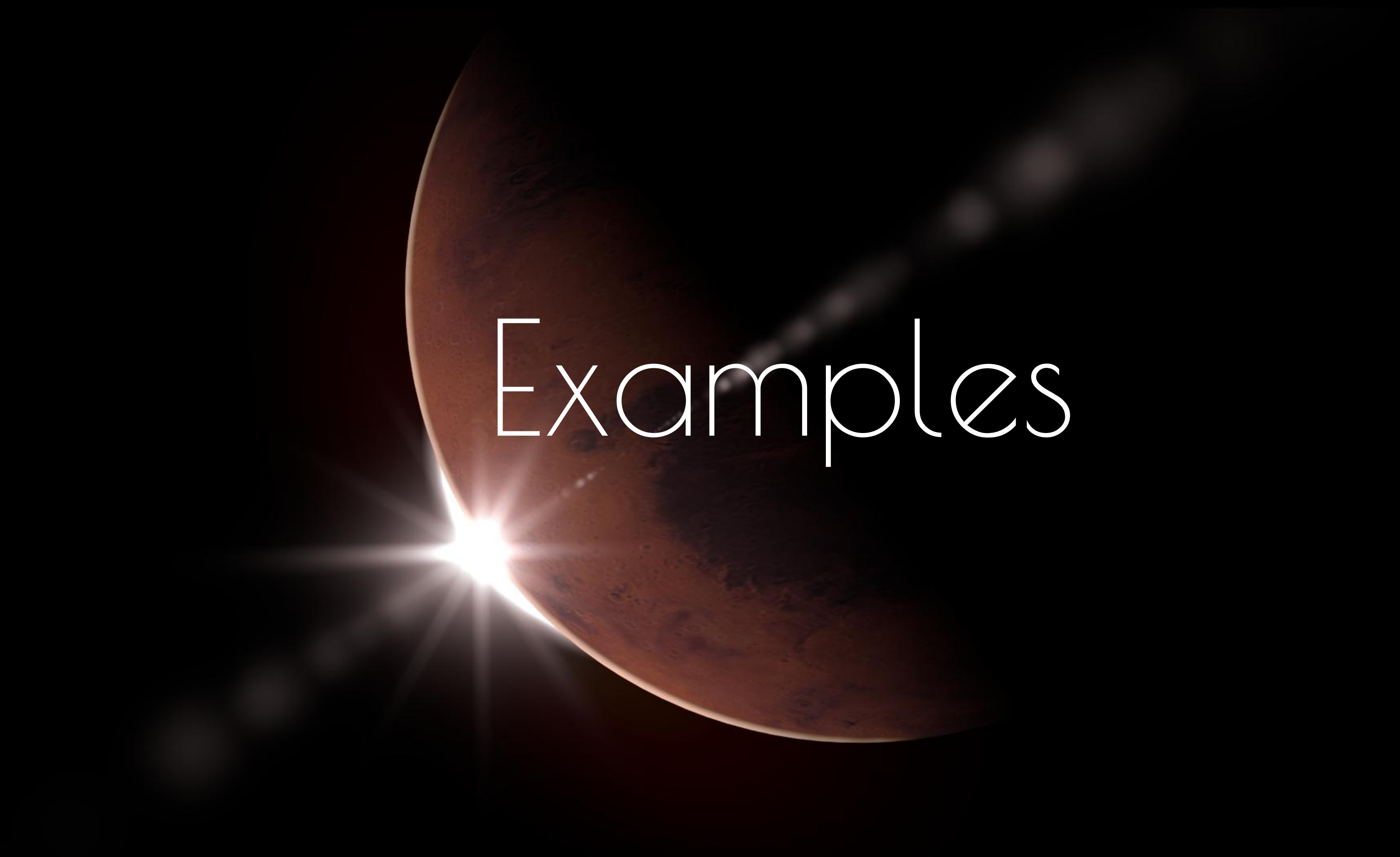
Brian Kent, NRAO

Rendering and Compositing



Brian Kent, NRAO

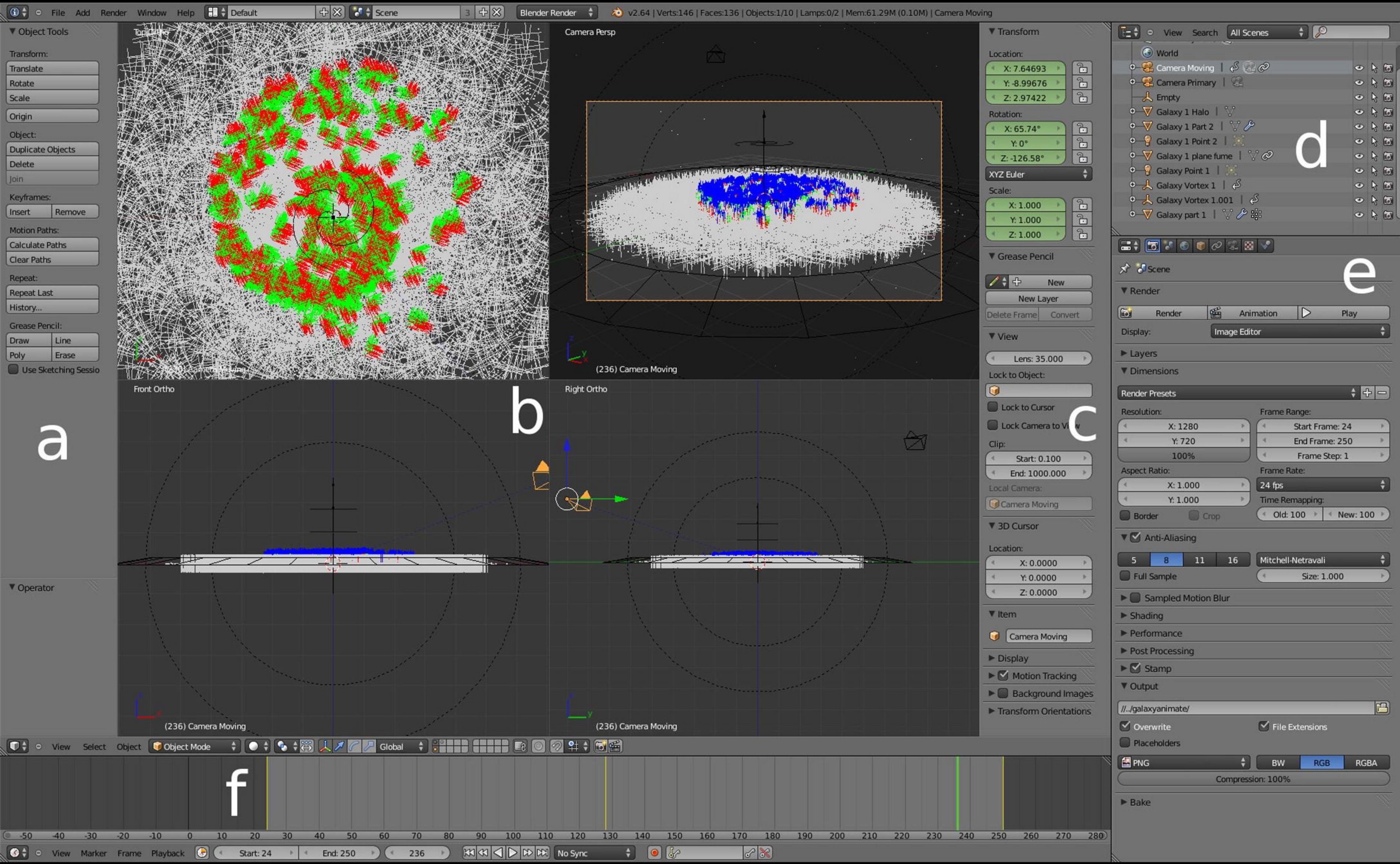
Examples



Brian Kent, NRAO

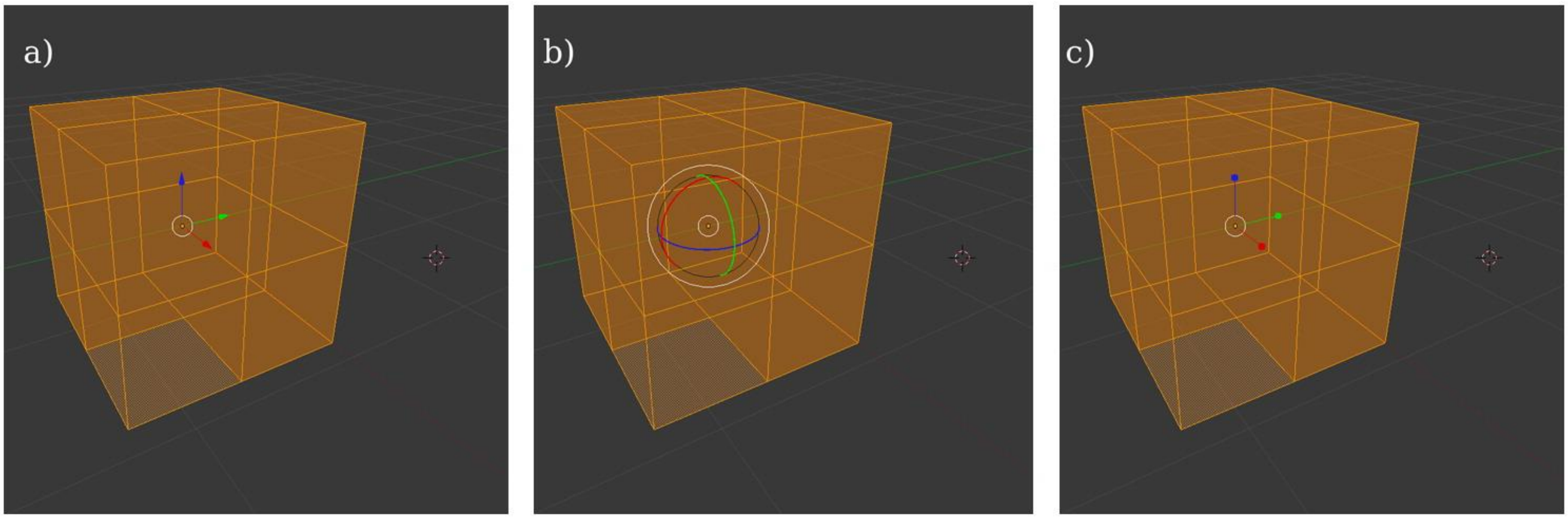
A Tour of the Blender Interface

Brian Kent, NRAO



Brian Kent, NRAO

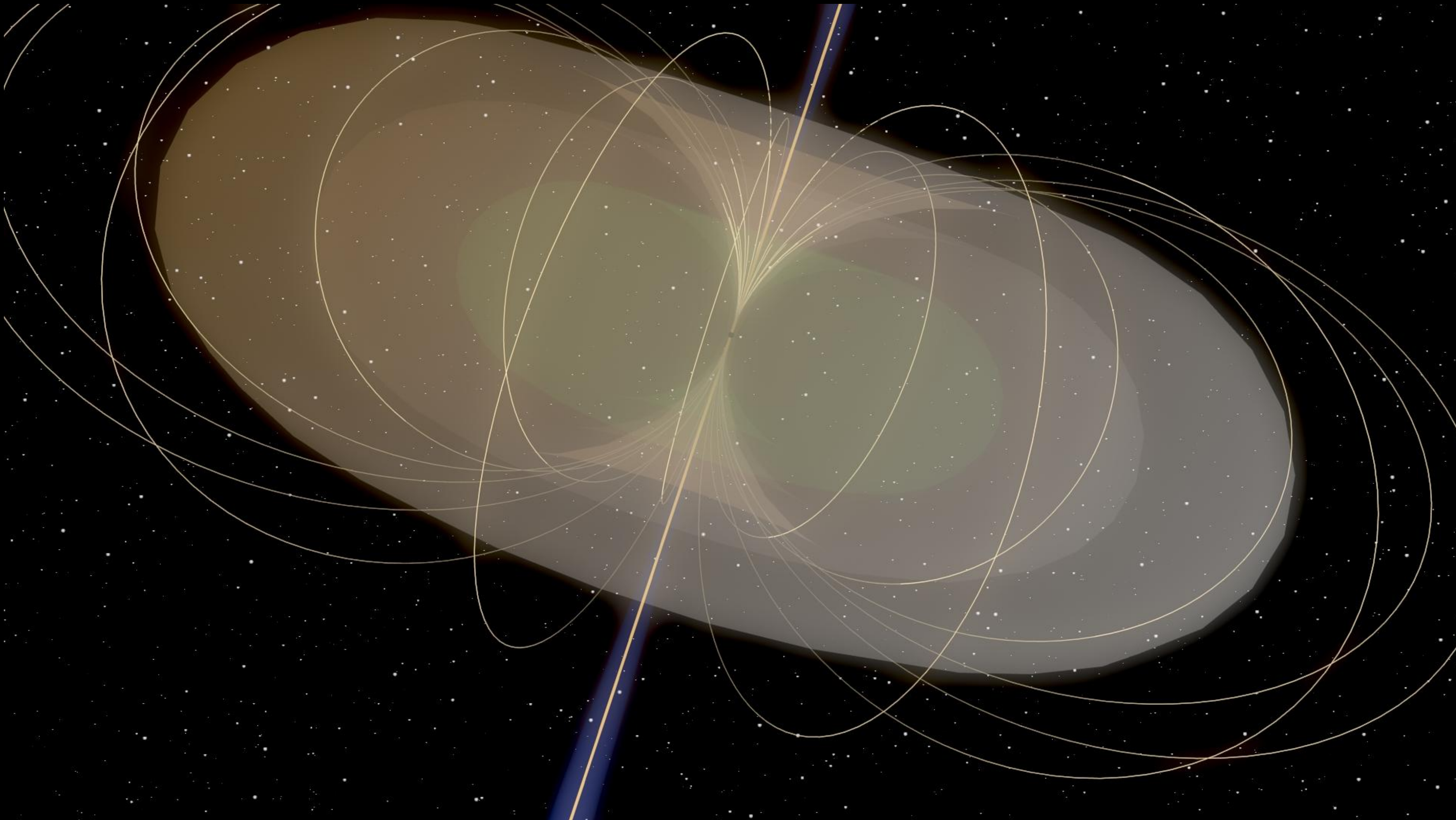
Blender interface



Translation **Rotation** **Scaling**

Brian Kent, NRAO

Brian Kent, NRAO



Brian Kent, NRAO

Data Cubes

- Gridded data can come from telescopes or simulations
- Radio telescopes produce grids that cover...
  - Two sky coordinates (RA and Decl.)
  - Frequency (Z - the doppler shifted velocity)
- These cubes can show the dynamics of galaxies, planetary disks, and large scale structure formation of clusters

https://www.youtube.com/watch?v=RDUVZ9MIW2I

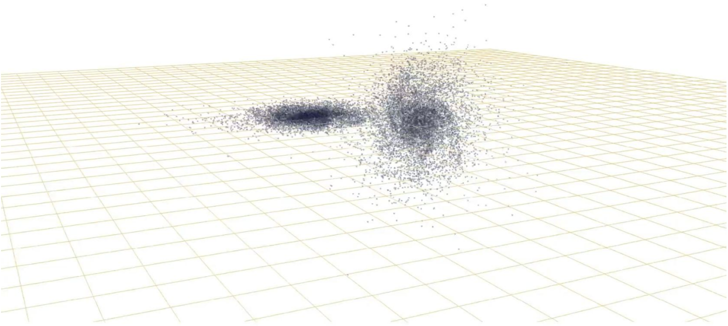
Data Cubes

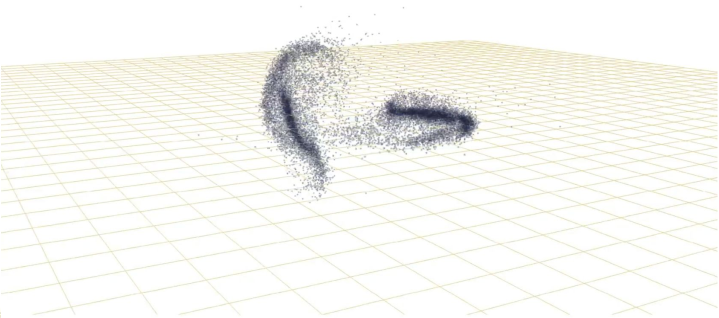
- Density maps of the nearby Universe can be created on regularly spaced grids.
- The results of these surveys allow to study not only the density of galaxies in 3D, but also the effects of gravity in the same regions of space...

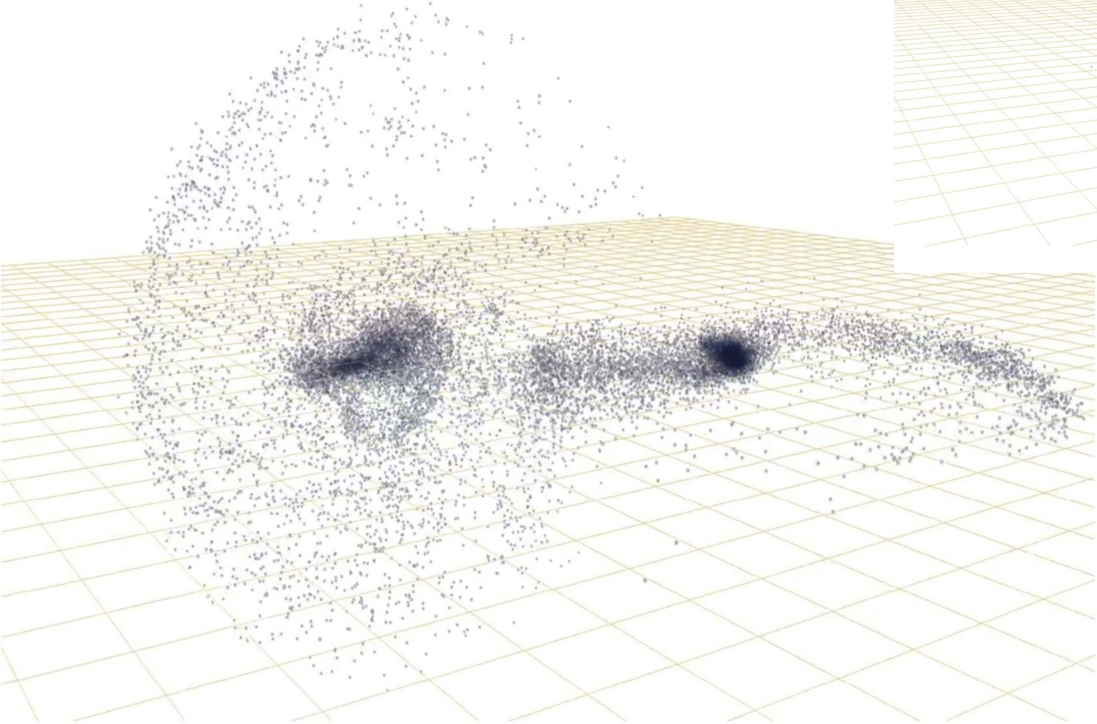
https://www.youtube.com/watch?v=3cuNT8_YEF0

N-body Simulations

- Data generated from GADGET-2 (Galaxies and Dark Matter Interacting 2) N-body/SPH code
  - http://www.mpa-garching.mpg.de/gadget/
- 30,000 particles, 1100 snapshots run for 2 billion years
- Blender Python interface used to bring XYZ position data into the vertices of Blender objects
- Objects are “textured” with Halos.
- Each grid square is approximately 33,000 light years

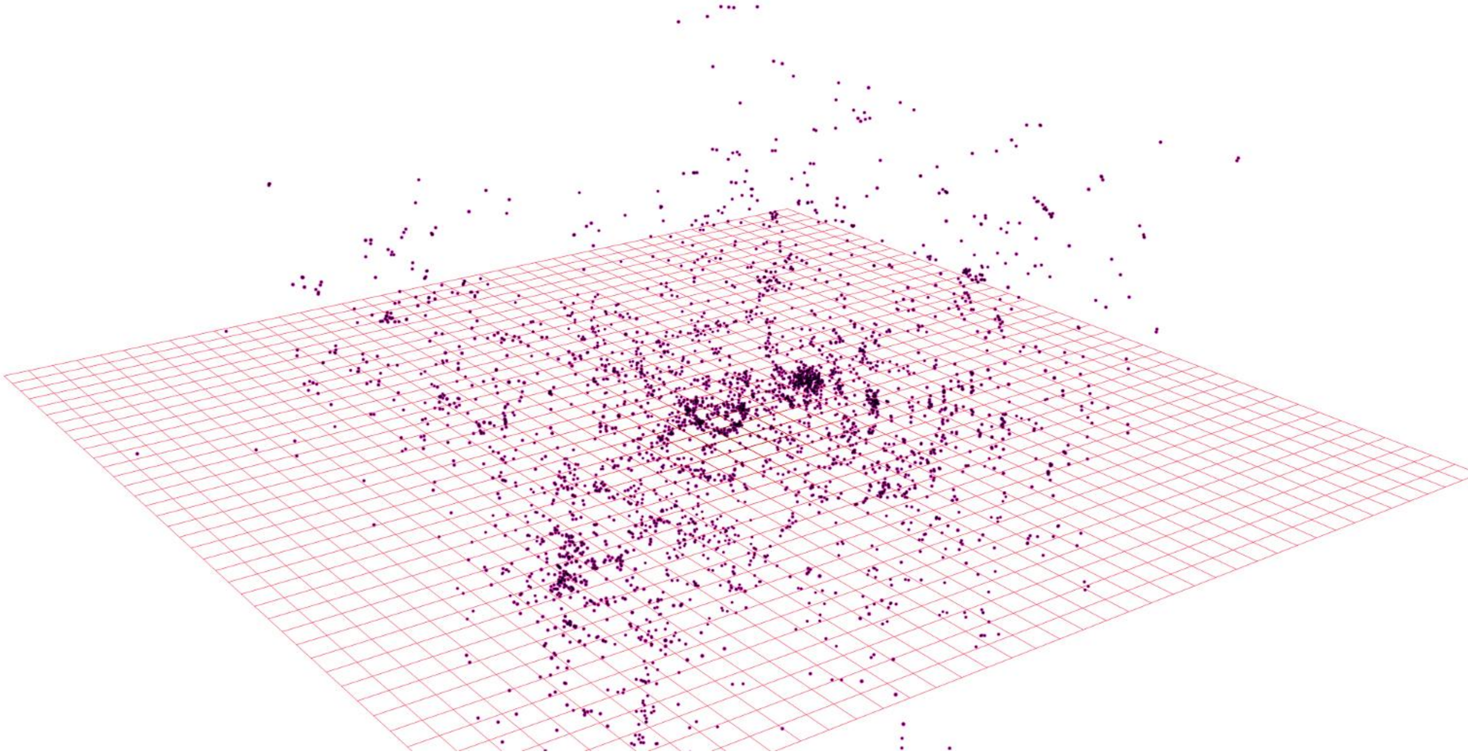






Brian Kent, NRAO

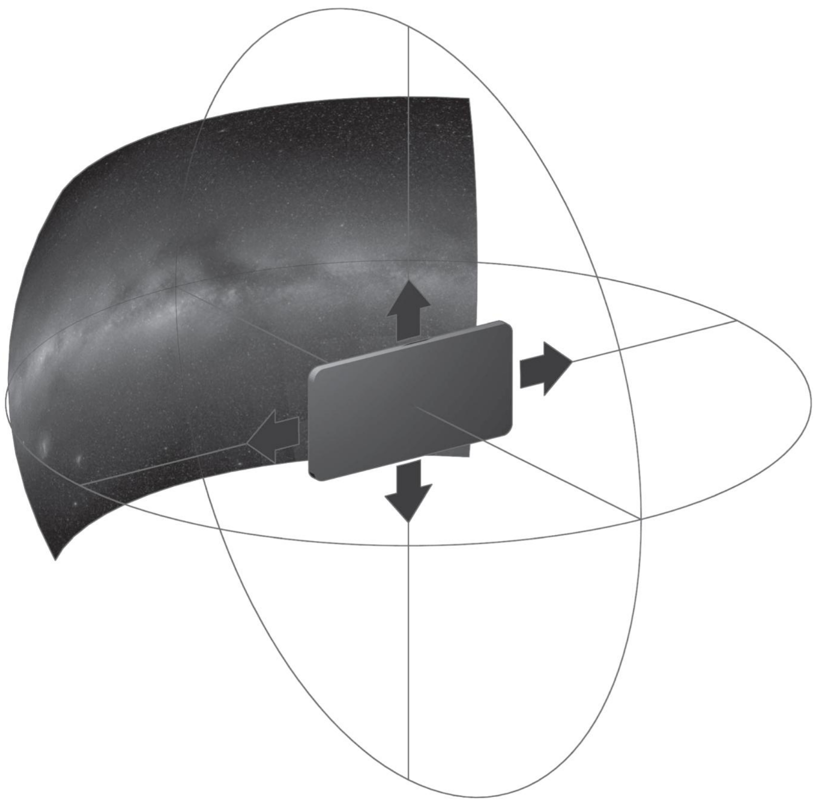
Galaxy Catalogs

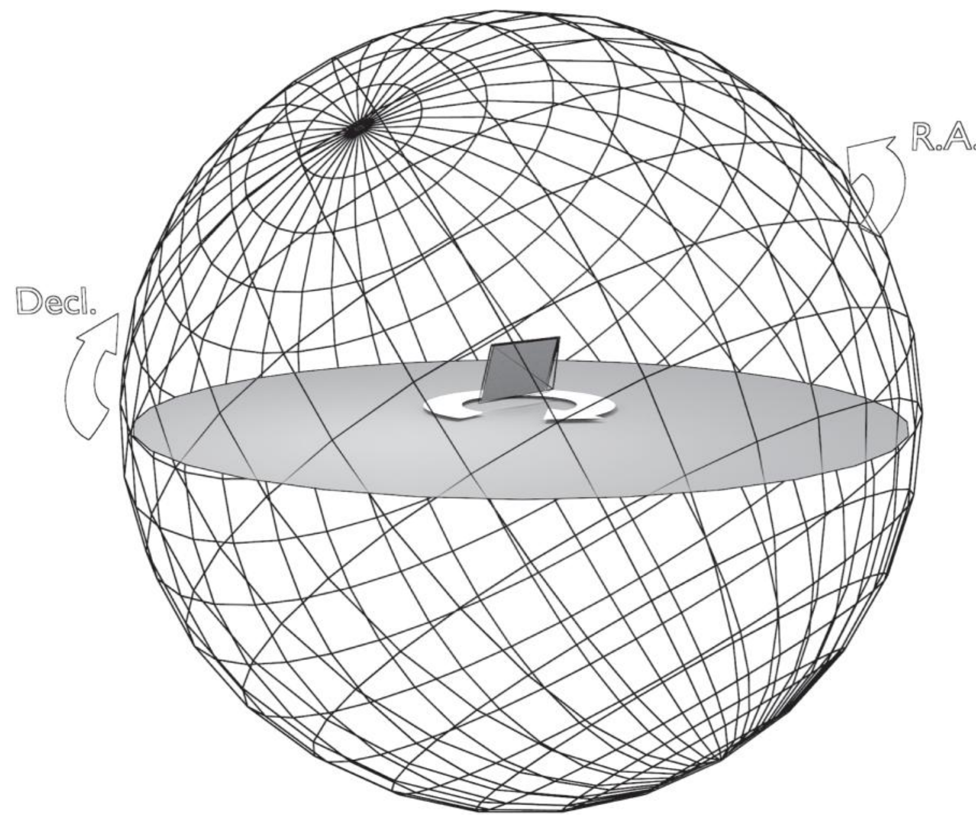


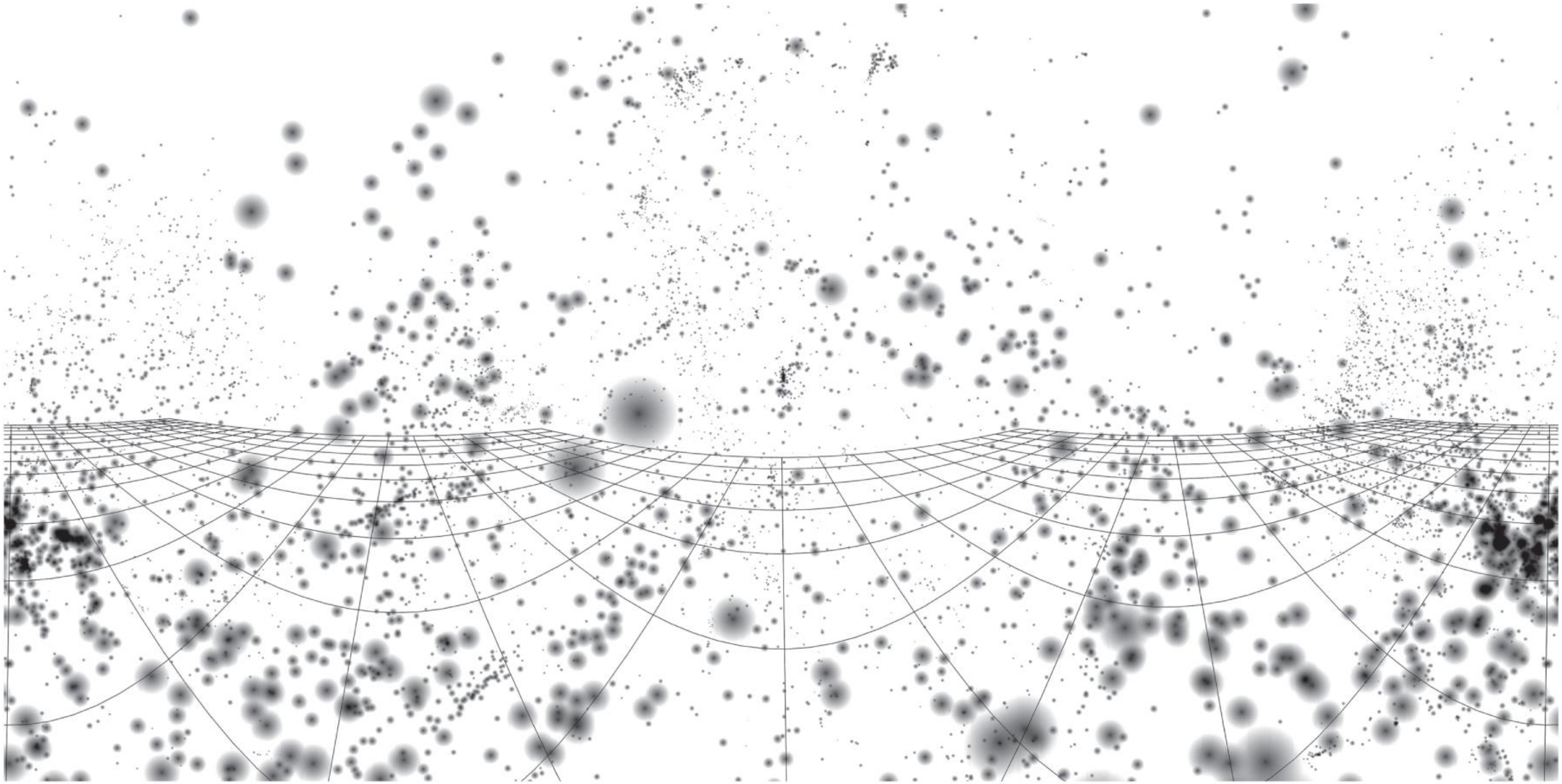
Brian Kent, NRAO

https://www.youtube.com/watch?v=n04M39SbfW0

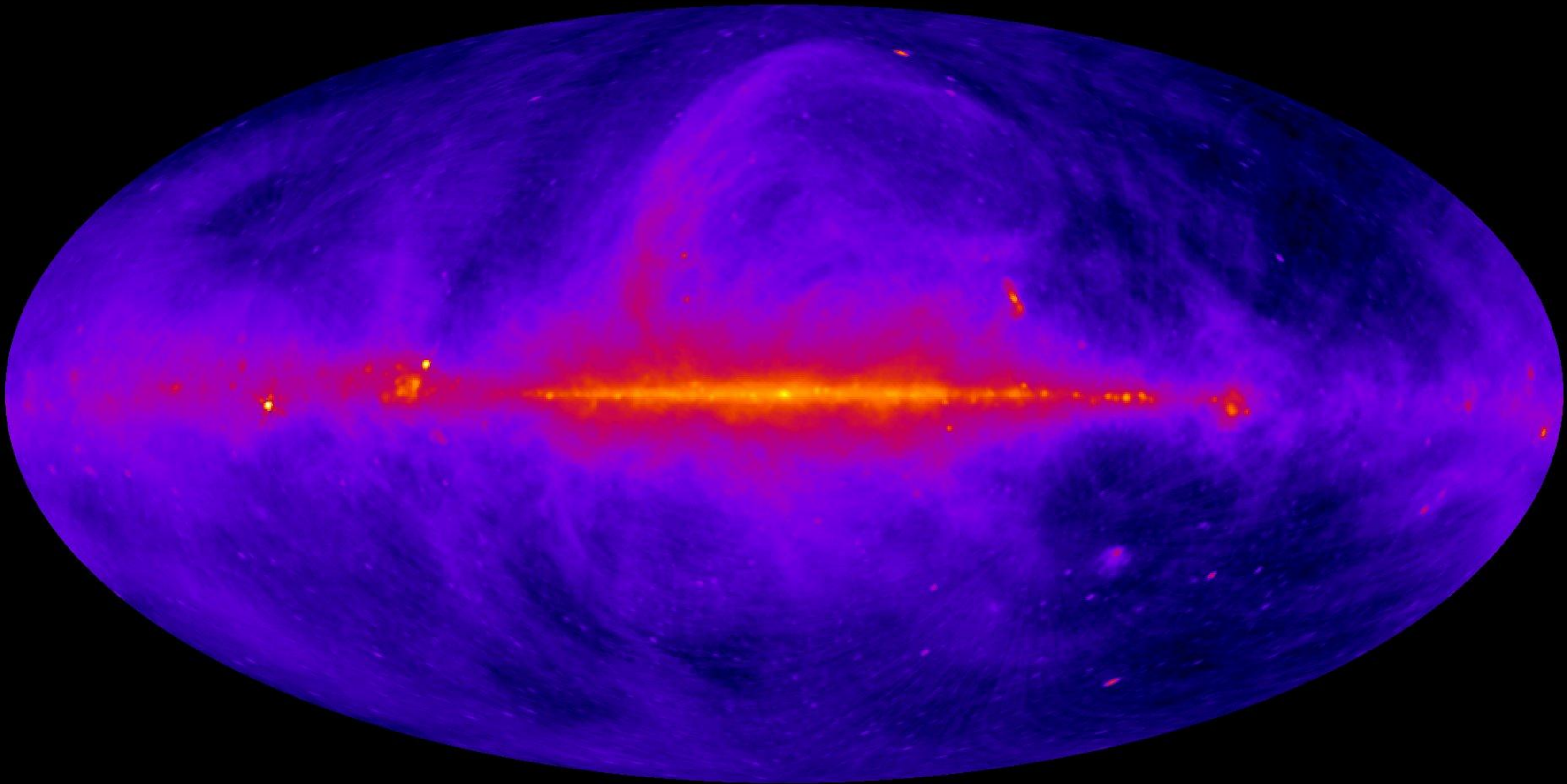
360 Panoramas







Tully et al. Cosmic Flows data



408 MHz

https://www.youtube.com/user/VisualizeAstronomy

Techniques and Methods for Astrophysical Data Visualization

Brian R. Kent, *National Radio Astronomy Observatory, Charlottesville, VA, USA*

Astrophysics continues to be a leader in the data sciences, with innovative methods being developed to handle new analysis challenges. The higher rates of data acquisition in both observational and theoretical astrophysics demand innovative solutions in scientific visualization. The Publications of the Astronomical Society of the Pacific (PASP) has published a special focus issue titled **Techniques and Methods for Astrophysical Data Visualization**. Refereed submissions for this issue cover a wide variety of visualization topics, including new software packages, visualization techniques, software from other industries, and new science results. These methods and techniques can serve as a complement to data analysis software, stand on their own for data exploration, or inspire with impressive visuals for science, technology, education, mathematics (STEM) and public outreach. A number of articles from our special focus issue feature videos and tutorials as well as interactive 3D content.

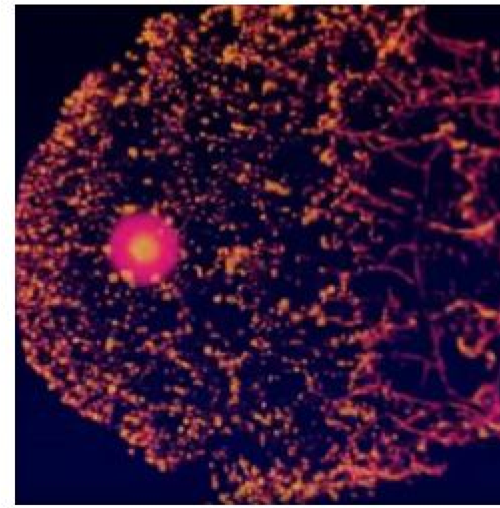
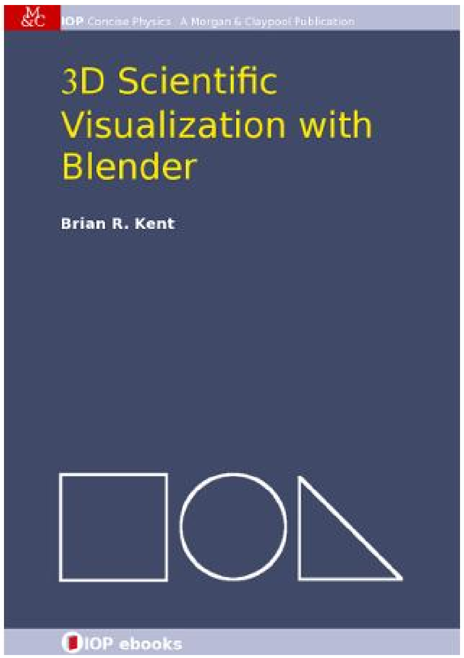


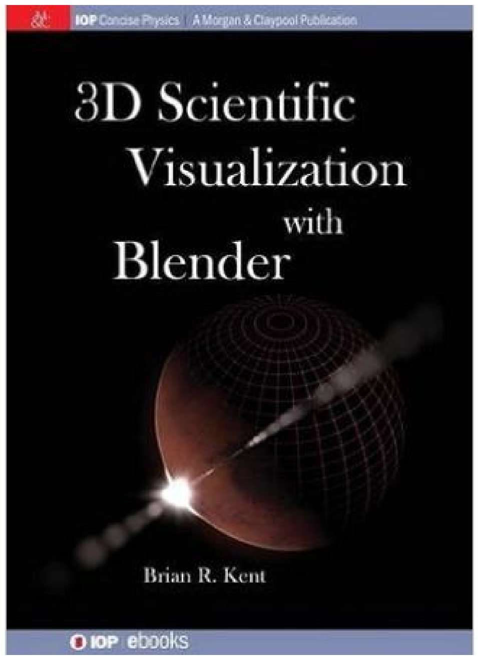
Figure. Stellar winds from a Wolf-Rayet star moving through the galactic center (Garate 2017). Simulation data provided by Cuadra et al. (2008).

"Visualization allows astronomers to break down and understand large data and explore multi-dimensional phase spaces. We encourage scientists to explore the tools and techniques presented in this issue and apply them to their own data and research."
- Brian R. Kent, Guest Editor, PASP

http://iopscience.iop.org/journal/1538-3873/page/Techniques-and-Methods-for-Astrophysical-Data-Visualization

Interesting in learning more?





Book and tutorials available at:

http://www.cv.nrao.edu/~bkent/blender/

http://amzn.com/1627056114

https://www.youtube.com/VisualizeAstronomy

Resources and Opportunities

3D Graphics and Python
http://www.cv.nrao.edu/~bkent/blender/

Blender
http://www.blender.org/

General Visualization
http://visual.ly/

http://www.visualizing.org/

SIGGRAPH and Pycon

http://www.siggraph.org

https://us.pycon.org/2015/