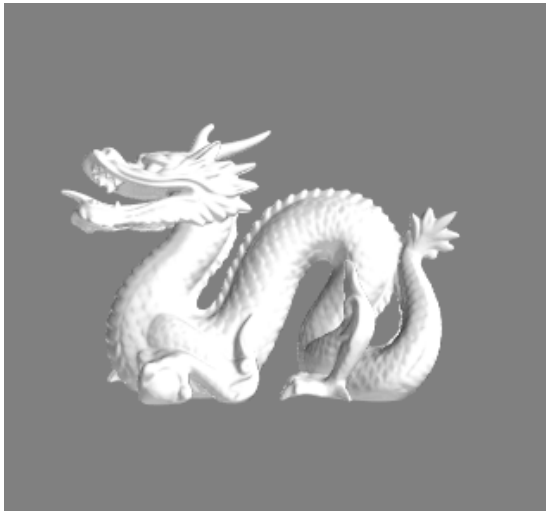
Mayavi可视化实例

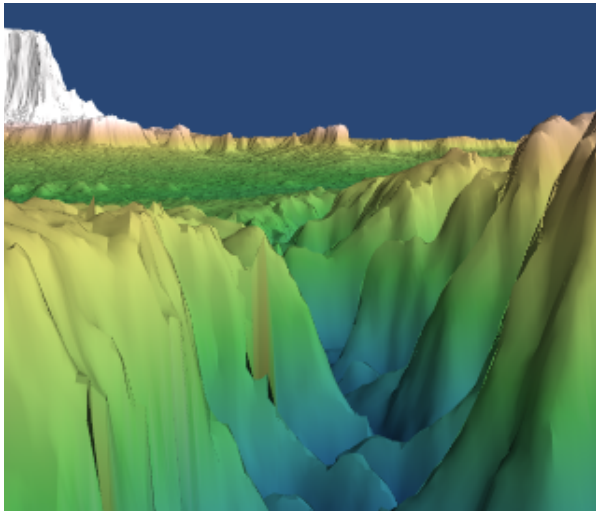
SV06

黄天羽

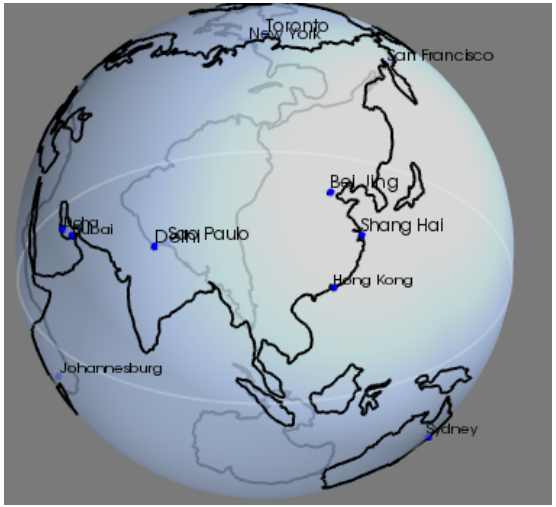
www.python123.org



Dragon

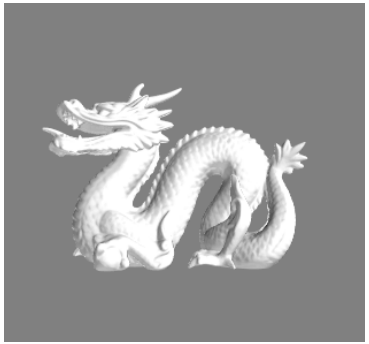


Canyon



Earth-graph

实例1：Dragon绘制



读取数据文件

dragon.tar.zip

```
import tarfile
#读取tar压缩文件
dragon_tar_file = tarfile.open('dragon.tar.gz')
try:
    os.mkdir('dragon_data')
except:
    pass
dragon_tar_file.extractall('dragon_data')
dragon_tar_file.close()
```

读取数据文件

os.path.join( )

```
from os.path import join
dragon_ply_file = join('dragon_data', 'dragon_recon', 'dragon_vrip.ply')
```

/dragon_data/dragon_recon/dragon_vrip.ply

ply格式文件

ply（Polygon File Format，
Stanford Triangle Format）

的米开朗基罗作品“大卫”雕像。

绘制数据的Surface

完整程序框架

- 打开文件
- 使用modules绘制数据的surface
- 显示可交互的结果

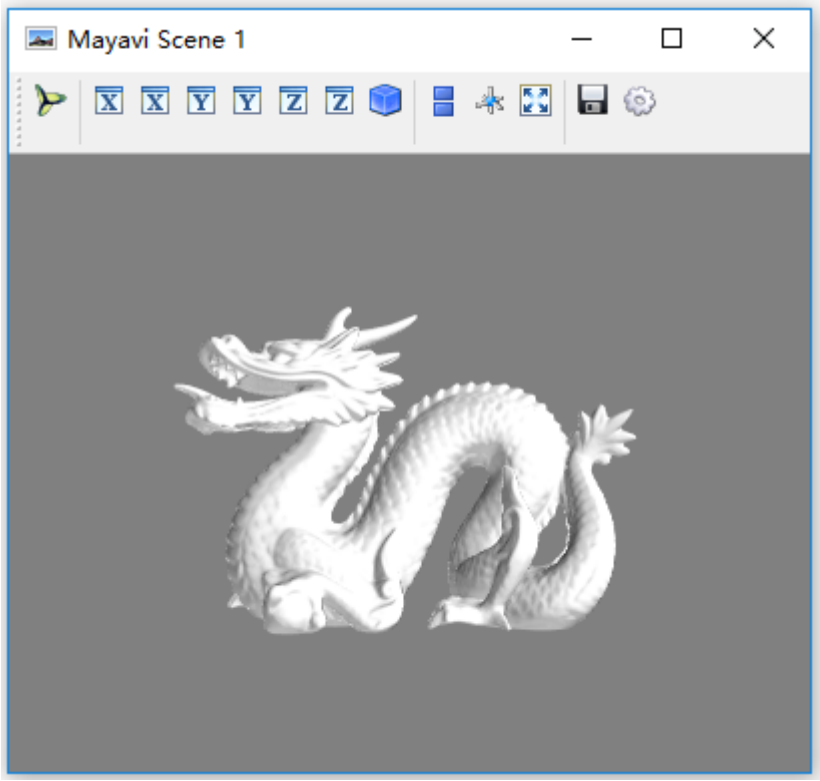
绘制数据的Surface

```
# 渲染dragon ply文件
from mayavi import mlab
mlab.pipeline.surface(mlab.pipeline.open(dragon_ply_file))
mlab.show()
```

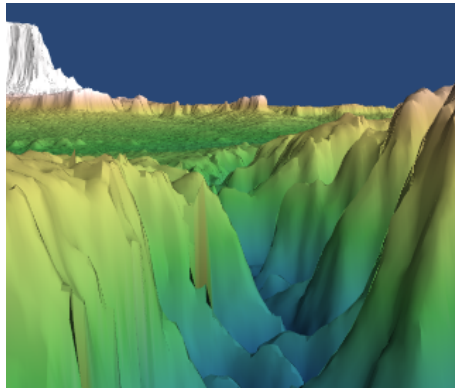
将解压数据删除

```
#删除解压的文件夹
import shutil
shutil.rmtree('dragon_data')
```

```
from os.path import join
from mayavi import mlab
import tarfile

#读取压缩文件
dragon_tar_file = tarfile.open('dragon.tar.gz')
try:
    os.mkdir('dragon_data')
except:
    pass
dragon_tar_file.extractall('dragon_data')
dragon_tar_file.close()
dragon_ply_file = join('dragon_data', 'dragon_recon', 'dragon_vrip.ply')

# 渲染dragon ply文件
mlab.pipeline.surface(mlab.pipeline.open(dragon_ply_file))
mlab.show()

#删除解压的文件夹
import shutil
shutil.rmtree('dragon_data')
```


Mayavi Scene 1
X
X
Y
Y
Z
Z

实例2：Canyon地形可视化



读取地形数据文件

N36W113.hgt.zip

```
#解压一个zip包
import zipfile
hgt=zipfile.zipFile('N36W113.hgt.zip').read('N36W113.hgt')
```

hgt格式文件

hgt（height File Format）

N36W113.hgt：北纬36-37度， 西经113-114度地形高程数据。

帕里亚峡谷Paria Canyon

处理地形数据

```
import numpy as np
data = np.fromstring(hgt,'>i2')
data.shape = (3601, 3601)
data = data.astype(np.float32)
data = data[:1000, 900:1900]
data[data == -32768] = data[data > 0].min()
```

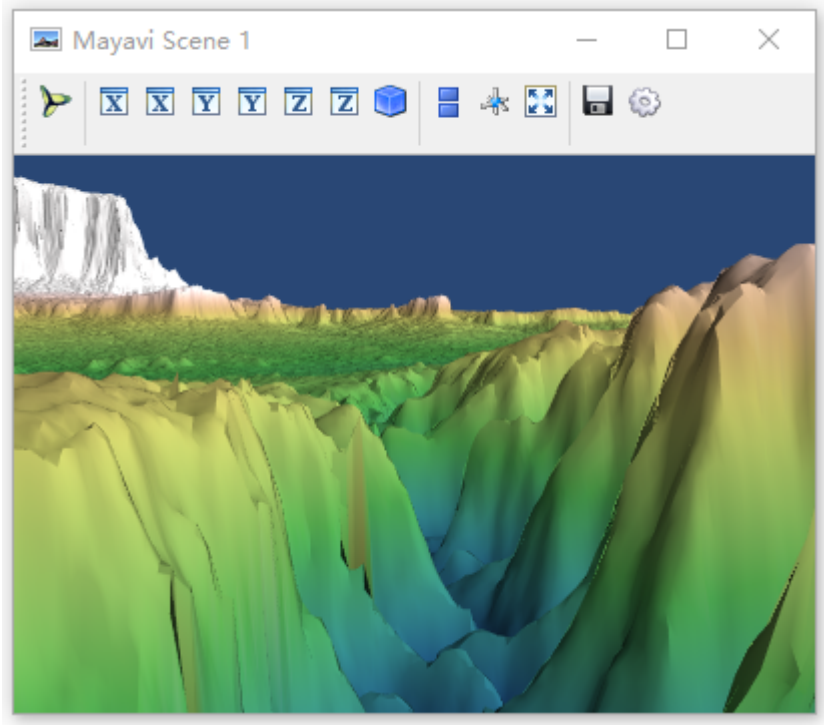
渲染地形数据

```
#渲染地形hgt的数据data
from mayavi import mlab
mlab.figure(size=(400, 320), bgcolor=(0.16, 0.28, 0.46))
mlab.surf(data, colormap='gist_earth', warp_scale=0.2,
              vmin=1200, vmax=1610)
```

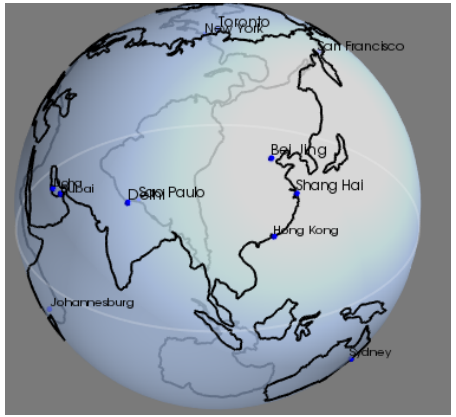
清空内存、创建窗口

```
#清空内存
del data
#创建交互式的可视化窗口
mlab.view(-5.9, 83, 570, [5.3, 20, 238])
mlab.show()
```

```
import zipfile
import numpy as np
from mayavi import mlab

#读取压缩文件
hgt = zipfile.ZipFile('N36W113.hgt.zip').read('N36W113.hgt')
data = np.fromstring(hgt,'>i2')
data.shape = (3601, 3601)
data = data.astype(np.float32)
data = data[:1000, 900:1900]
data[data == -32768] = data[data > 0].min()
#渲染地形hgt的数据data
mlab.figure(size=(400, 320), bgcolor=(0.16, 0.28, 0.46))
mlab.surf(data, colormap='gist_earth', warp_scale=0.2,
            vmin=1200, vmax=1610)
#清空内存
del data
#创建交互式的可视化窗口
mlab.view(-5.9, 83, 570, [5.3, 20, 238])
mlab.show()
```


Mayavi Scene 1

实例3：Earth Graph



搜索全球城市坐标数据

“city, longtitude，latitude”

```
# 城市经纬度数据
cities_data = """
Bei Jing, 116.23,39.54
Shang Hai, 121.52, 30.91
Hong Kong,114.19,22.38
Delhi,77.21,28.67
Johannesburg,28.04,-26.19
Doha,51.53,25.29
Sao Paulo,-46.63,-23.53
Toronto,-79.38,43.65
New York,-73.94,40.67
San Francisco,-122.45,37.77
Dubai,55.33,25.27
Sydney,151.21,-33.87
"""
```

读取城市数据文件

```
# 建立城市-城索引的字典、城市经纬度的列表
import csv
cities = dict()
coords = list()
for line in list(csv.reader(cities_data.split('\n')))[1:-1]:
    name, long_, lat = line
    cities[name] = len(coords)
    coords.append((float(long_), float(lat)))
```

坐标转换

$$\begin{cases} x = \cos(long) * \cos(lat) \\ y = \cos(long) * \sin(lat) \\ z = \sin(long) \end{cases}$$

```
import numpy as np
coords = np.array(coords)
# 将经纬度的位置转换为三维坐标
lat, long = coords.T * np.pi / 180
x = np.cos(long) * np.cos(lat)
y = np.cos(long) * np.sin(lat)
z = np.sin(long)
```

绘制部分

1、建立窗口

2、绘制地球（绘制球体、修饰效果）

3、绘制城市名字

4、绘制大洲边界

5、绘制赤道

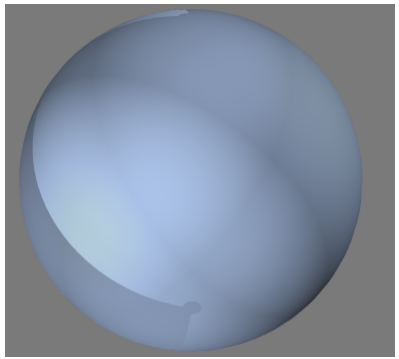
6、显示可交互窗口

建立窗口

```
# 建立窗口
from mayavi import mlab
mlab.figure(bgcolor=(0.48, 0.48, 0.48), size=(400, 400))
… …

# 设定相机及焦点位置
mlab.view(100, 60, 4, [-0.05, 0, 0])
mlab.show()
```

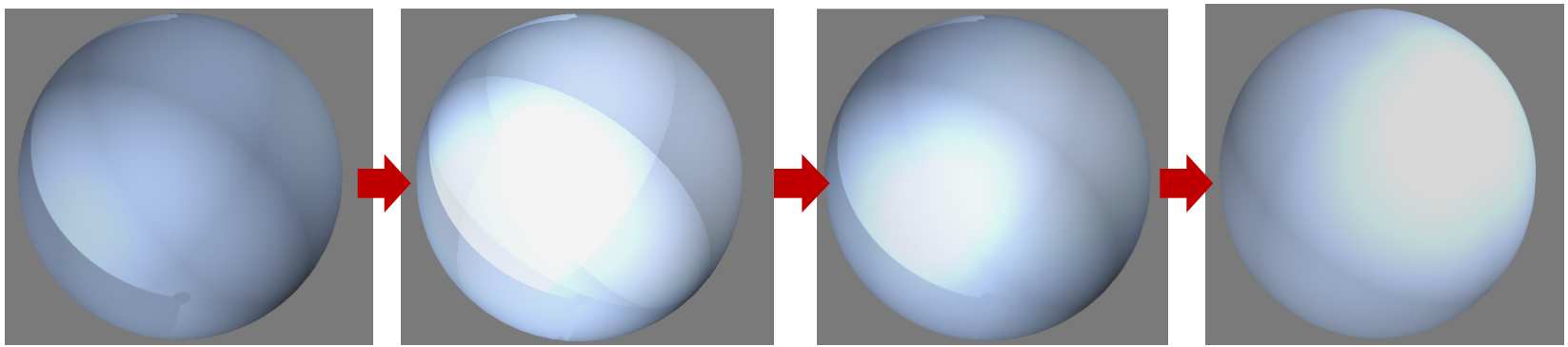
绘制地球-球体

```
# 绘制半透明球体，表示地球外表面
sphere = mlab.points3d(0, 0, 0, scale_factor=2,
                               color=(0.67, 0.77, 0.93),
                               resolution=50,
                               opacity=0.7,
                               name='Earth')
```



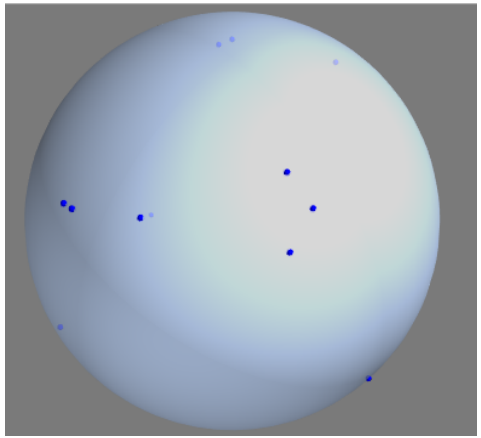
绘制地球-效果

```
# 调整镜面反射参数
sphere.actor.property.specular = 0.45
sphere.actor.property.specular_power = 5
# 设置背面剔除，以更好的显示透明效果
sphere.actor.property.backface_culling = True
```



绘制城市名称

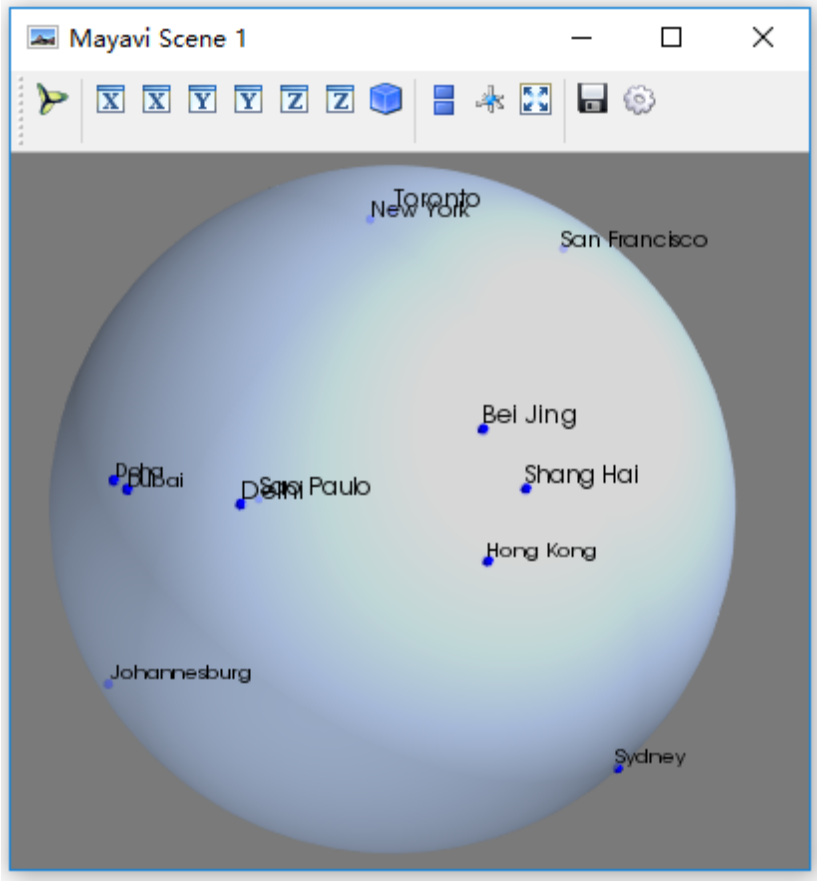
```
points = mlab.points3d(x, y, z,
                       scale_mode='none',
                       scale_factor=0.03,
                       color=(0, 0, 1))
```



绘制城市名称

mlab.text(x, y, text...)

```
# 显示城市名字
for city, index in cities.items():
    label = mlab.text(x[index], y[index], city, z=z[index],
                      width=0.016 * len(city), name=city)
    label.property.shadow = True
```


Mayavi Scene 1
Toronto
New York
San Francisco
Bei Jing
Shang Hai
Hong Kong
Sao Paulo
Johannesburg
Sydney

绘制部分

1、建立窗口

2、绘制地球（绘制球体、修饰效果）

3、绘制城市名字

4、绘制大洲边界

5、绘制赤道

6、显示可交互窗口

绘制大洲边界

```
from mayavi.sources.builtin_surface import BuiltinSurface
```

```
20    ######################################################################
21    # `BuiltinSurface` class.
22    ######################################################################
23    class BuiltinSurface(Source):
24        # The version of this class.  Used for persistence.
25        __version__ = 0
26                                                          'earth':tvtk.EarthSource()
27        # Flag to set the poly data type.
28        source = Enum('arrow','cone','cube','cylinder','disk','earth','line',
29                      'outline','plane','point', 'polygon','sphere',
30                      'superquadric','textured sphere', 'glyph2d',
31                      desc='which poly data source to be used')
```

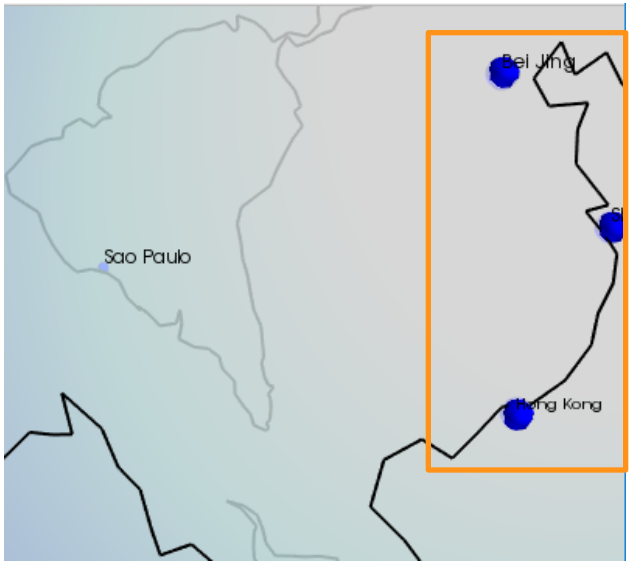
绘制大洲边界

```
# 显示大洲边界
from mayavi.sources.builtin_surface import BuiltinSurface
continents_src = BuiltinSurface(source='earth', name='Continents')
continents = mlab.pipeline.surface(continents_src, color=(0, 0, 0))
```

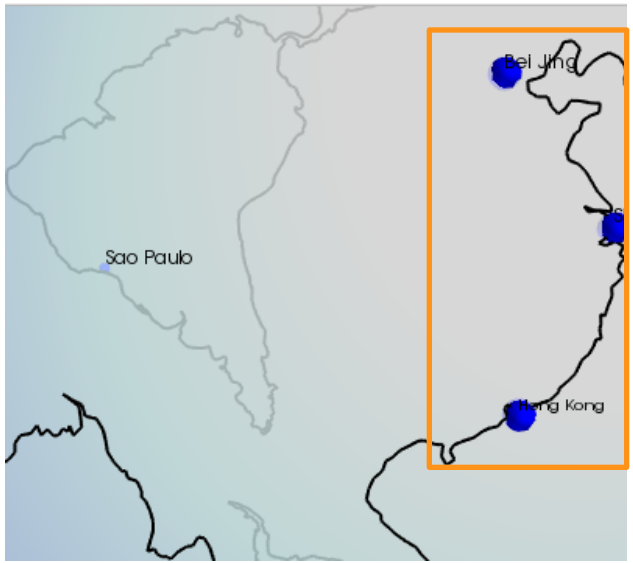
绘制大洲边界

问题：大洲边界细节粗糙

解决：使用LOD（Levels-of-details）



一级LOD



二级LOD

绘制大洲边界

BuiltinSurface.data_source.on_ratio设置模型LOD的层级

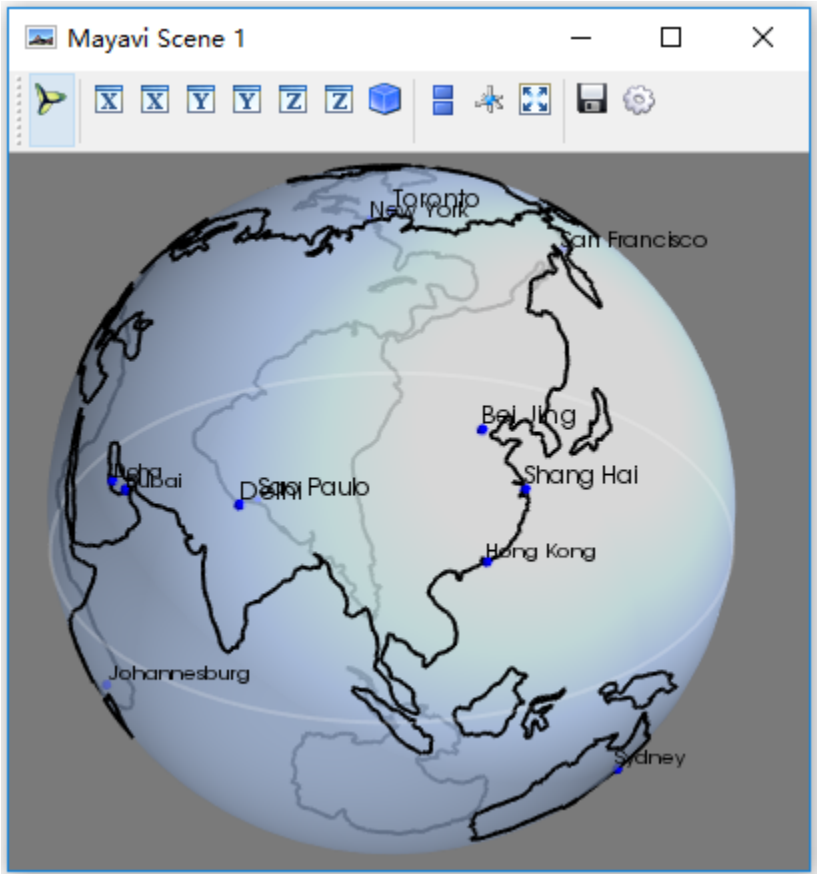
```
# 显示大洲边界
from mayavi.sources.builtin_surface import BuiltinSurface
continents_src = BuiltinSurface(source='earth', name='Continents')
# 设置LOD为2
continents_src.data_source.on_ratio = 2
continents = mlab.pipeline.surface(continents_src, color=(0, 0, 0))
```

绘制赤道

```
#赤道线numpy数组的构造过程
theta = np.linspace(0, 2 * np.pi, 100)
x = np.cos(theta)
y = np.sin(theta)
z = np.zeros_like(theta)
#绘制赤道线
mlab.plot3d(x, y, z, color=(1, 1, 1),
            opacity=0.2, tube_radius=None)
```

```python
# 城市经纬度数据
cities_data = """
Bei Jing, 116.23,39.54
Shang Hai, 121.52, 30.91
Hong Kong,114.19,22.38
Delhi,77.21,28.67
Johannesburg,28.04,-26.19
Doha,51.53,25.29
Sao Paulo,-46.63,-23.53
Toronto,-79.38,43.65
New York,-73.94,40.67
San Francisco,-122.45,37.77
Dubai,55.33,25.27
Sydney,151.21,-33.87
"""
########## 读取数据##########
# 建立城市-城索引的字典、城市经纬度的列表
import csv
cities = dict()
coords = list()
for line in list(csv.reader(cities_data.split('\n')))[1:-1]:
    name, long_, lat = line
    cities[name] = len(coords)
    coords.append((float(long_), float(lat)))

########## 坐标转换##########
# 将经纬度的位置转换为三维坐标
import numpy as np
coords = np.array(coords)
lat, long = coords.T * np.pi / 180
x = np.cos(long) * np.cos(lat)
y = np.cos(long) * np.sin(lat)
z = np.sin(long)

##########建立窗口##########
from mayavi import mlab
mlab.figure(bgcolor=(0.48, 0.48, 0.48), size=(400, 400))
```

```python
##########绘制地球##########
# 绘制半透明球体表示地球
sphere = mlab.points3d(0, 0, 0, scale_factor=2,
                               color=(0.67, 0.77, 0.93),
                               resolution=50,
                               opacity=0.7,
                               name='Earth')
# 调整镜面反射参数
sphere.actor.property.specular = 0.45
sphere.actor.property.specular_power = 5
# 设置背面剔除，以更好的显示透明效果
sphere.actor.property.backface_culling = True

##########绘制城市##########
# 绘制城市位置
points = mlab.points3d(x, y, z, scale_factor=0.03,color=(0, 0, 1))
# 绘制城市名称
for city, index in cities.items():
    label = mlab.text(x[index], y[index], city,
                      z=z[index], color=(0,0,0),
                      width=0.016 * len(city), name=city)

##########绘制大洲边界##########
from mayavi.sources.builtin_surface import BuiltinSurface
continents_src = BuiltinSurface(source='earth', name='Continents')
# 设置LOD为2
continents_src.data_source.on_ratio = 2
continents = mlab.pipeline.surface(continents_src, color=(0, 0, 0))

##########绘制赤道##########
theta = np.linspace(0, 2 * np.pi, 100)#平分360为100份
x = np.cos(theta)
y = np.sin(theta)
z = np.zeros_like(theta)
mlab.plot3d(x, y, z, color=(1, 1, 1),opacity=0.2, tube_radius=None)

##########显示可交互窗口##########
mlab.view(100, 60, 4, [-0.05, 0, 0])
mlab.show()
```


Mayavi Scene 1
Toronto
New York
San Francisco
Bei Jing
Shang Hai
Hong Kong
Sao Paulo
Johannesburg
Sydney