Copilot

Sure, I can help with that. Here's how you can modify your code to make `key_theta` and `key_phi` represent the orientation of the largest axis of the ellipsoid:

```python
import numpy as np
import scipy as sp
import matplotlib.pyplot as plt
import matplotlib.animation as animation
from matplotlib.animation import PillowWriter

def draw_ellipsoid(t, dims, ax):
    rotation = slerp(t)
    print(rotation.as_euler('zxy'))
    c = [cx_interp(t), cy_interp(t), cz_interp(t)]
    x = dims[0] * np.outer(np.cos(u), np.sin(v)) + c[0]
    y = dims[1] * np.outer(np.sin(u), np.sin(v)) + c[1]
    z = dims[2] * np.outer(np.ones(np.size(u)), np.cos(v)) + c[2]
    rotated = rotation.apply(np.column_stack([x.ravel(), y.ravel(), z.ravel()]))
    x = rotated[:, 0].reshape(x.shape)
    y = rotated[:, 1].reshape(y.shape)
    z = rotated[:, 2].reshape(z.shape)
    ax.cla()
    ax.set_xlim([-2 * dims[0], 2 * dims[0]])
    ax.set_ylim([-2 * dims[1], 2 * dims[1]])
    ax.set_zlim([-2 * dims[2], 20 * dims[2]])
    ax.plot_surface(x, y, z, color = 'b', alpha = .75)
    ax.set_xlabel('X')
    ax.set_ylabel('Y')
    ax.set_zlabel('Z')

lx = 2.5
ly = 1.5
lz = .5
dims = [lx, ly, lz]
cx = 0.0
cy = 0.0
cz = 0.0
c = [cx, cy, cz]
nangs = 100
u = np.linspace(start = 0, stop = 2 * np.pi, num = nangs)
v = np.linspace(start = 0.0, stop = np.pi, num = nangs)
key_times = [0.0, 1.2, 2.6, 3.2, 4.1, 4.8]
cx = [0.0, 1.0, 2.0, 3.0, 4.0, 5.0]
cy = [0.0, 1.0, 2.0, 3.0, 4.0, 5.0]
cz = [0.0, 1.0, 2.0, 2.0, 2.0, 2.0]
key_theta = np.radians([0.0, 23.5, 48.79, 67.2, 80.0, 90.0])
key_phi = np.radians([0.0, 0.0, 0.0, 10.0, 17.5, 17.5])

# Convert spherical coordinates to Cartesian coordinates
key_direction = np.column_stack([
    np.sin(key_theta) * np.cos(key_phi),
    np.sin(key_theta) * np.sin(key_phi),
    np.cos(key_theta)
])

# Create a rotation matrix that aligns the z-axis with the key direction
key_rot = sp.spatial.transform.Rotation.align_vectors(
    np.array([[0, 0, 1]] * len(key_times)),  # z-axis
    key_direction  # key direction
)[0]

times = np.linspace(np.min(key_times), np.max(key_times), 50)
slerp = sp.spatial.transform.Slerp(key_times, key_rot)
cx_interp = sp.interpolate.interp1d(key_times, cx)
cy_interp = sp.interpolate.interp1d(key_times, cy)
cz_interp = sp.interpolate.interp1d(key_times, cz)
fig = plt.figure()
ax = fig.add_subplot(111, projection = '3d')
ani = animation.FuncAnimation(fig, draw_ellipsoid, frames=times, fargs=(dims, ax,))
```

```
# ani.save("ellipsoid_animation.gif", writer = PillowWriter(fps = 30))
plt.show()
```

In this modified code, `key_theta` and `key_phi` are converted to Cartesian coordinates to get the direction of the largest axis in 3D space. Then, the `align_vectors` function is used to create a rotation matrix that aligns the z-axis (which represents the largest axis of the ellipsoid before rotation) with this direction. The resulting `key_rot` represents the rotations that align the largest axis of the ellipsoid with the desired direction.

I hope this helps! Let me know if you have any other questions. 😊