Structure of 1 Billion Star Python 3D Isosurface Render Program

Load Python libraries:
- os: file operations
- pandas: read CSV files in pieces
– numpy: fast array operations
- pyvista: 3D visualization/isosurface rendering
– scipy [gaussian_filter]: smooth the 3D grid
- multiprocessing: allow code to be run on multiple CPU cores
- time: keep track of how long the program is running

Import csv files

Set the program variables: max distance in light years, grid size, sigma, number of cores, chunk size [# of rows], exported file name

Define 1st function to process each chunk of data:
- Check data for consistency and clean data
- Convert parallax to light years
- Use triginometry to convert right ascension, declination and light years to x, y, z coordinates
- Build out 3D histogram
- Return histogram array and star count

Define 2nd function to send 1 chunk of data to be processed:
- Read in csv file
- Cut off proper chunk size from file, eg: 500k lines
- Send to 1st function to be processed
- Wait for request to cut off the next chunk
- **This is key part of the program, as it allow the computer to process one small chunk at a time, not the whole file**

Start Main function:
- Build out the grid edges
- Get the size of the file, initialize 3D histogram array, start the star & chunk count
- Call 2nd function to generate a chunks of 500k stars which feeds into 1st function to start processing chunks
- Use the multiprocessing library to spread the processing among several CPU cores
- Accumulate histogram results in master array

Smooth out the grid:
- Use the Gaussian filter to start smoothing process

Calculate the top 20 densest regions

Use the PyVista library to start generating the grid

Set up colors for density thresholds, ultra sparse to maximum

Render Earth in the middle of the visualization

Generate legend to indicate density colors

Export MP4 video of rotating star clusters

Generate summary data: runtime, number of stars, grid size and number of surfaces

PyVista

SciPy

pandas

NumPy

OS Module