CS677 Assignment 2: Parallel 3D Volume Rendering (100 marks)
Submission deadline: November 10, 2024, 11:59 PM (hard deadline)

Assume a 3D scalar dataset, you are required to perform parallel volume rendering using MPI. The domain will be decomposed in 3 dimensions. Each subdomain will be handled by 1 process to perform the ray casting algorithm as we discussed in class (and as was given in Assignment 1). You will use front-to-back compositing. Assuming orthogonal projection, the view will be aligned with the XY plane. Assume that the rays pass through only the grid points of the data along the Z direction. The step size (distance between two sample points along the ray) is an input parameter. The output of your program is the final volume rendered image.

Link to test dataset: Isabel_High_Resolution_Raw

Example of 3D domain decomposition: 3D domain decomposition along all 3 axes

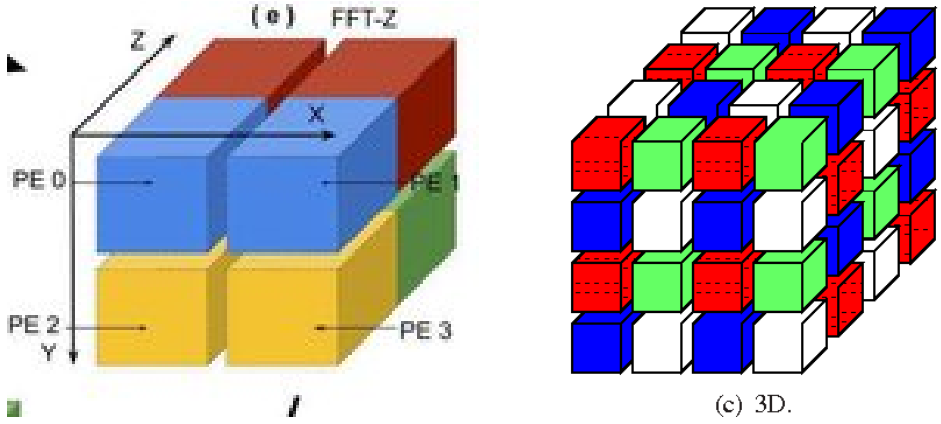

Figure source: https://www.researchgate.net/

Please ignore the annotations given on the figure.
X-axis: Horizontal
Y-axis: Vertical
Z-axis: Going into the screen

Logical code sequence:

- File read by rank 0
- Rank 0 distributes required data to all ranks
- Volume rendering in parallel by each rank
- Parallel composition (feel free to optimize using binary swap-like algorithm)
- Stitch the image and generate a png or jpeg file
- Do not hardcode any input parameter in the code

Input:

- Number of processes P (i.e. specified using -np)
- Dataset (Input file name)
- PX (number of processes in X-dimension)
- PY (number of processes in Y-dimension)
- PZ (number of processes in Z-dimension)
- Stepsize (default is 0.5)
- Opacity TF
- Color TF

Note that P = PX * PY * PZ

Output:

- Output the final image as “PX_PY_PZ.png” (filename format)
- Output the communication time and computation time and total execution time separately (the maximum time taken by any process)

Execution and submission instructions

- Run your code for three test cases (run 2 times each test case):
    - mpirun -np 8 -f hostfile ./executable Isabel_1000x1000x200_float32.raw 2 2 2 0.5 opacity.json color.json
    - mpirun -np 16 -f hostfile ./executable Isabel_1000x1000x200_float32.raw 2 2 4 0.5 opacity.json color.json
    - mpirun -np 32 -f hostfile ./executable Isabel_1000x1000x200_float32.raw 2 2 8 0.5 opacity.json color.json

      hostfile contents may be something like below (specify the host name along with number of processes per host):

      csews1:8
      csews2:8
      csews3:8
      csews4:8

      (include any four nodes that are available when you are running the code - for example “ping csews1” should indicate whether it is up and running or not)

- You have to run your code on csews* systems (172.27.19.1 - 40). Please report unavailability of nodes through issues.cse.iitk.ac.in and cc the TAs and instructors.

- To run on multiple processes, please follow these instructions
    - Enable password-less SSH to 172.27.19.1 - 30 (login to any IP)
    - Login to 172.27.19.1 – 172.27.19.40 (note some systems may be down)
        - Download MPICH from https://www.mpich.org/downloads/

    - Installation guide: https://www.mpich.org/documentation/guides/
    - Install MPICH in your home directory (not /tmp)
        - ./configure --prefix=<path-to-your-installation-directory>
        - make [This will take some time]
        - make install
    - Login to any csews system with your cse ID (IP: 172.27.19.1 – 172.27.19.40)
    - Verify: which mpirun (should point to your installed binary)
        - E.g. /home/abc/install/mpirun (not /usr/bin/mpirun)
    - In case of difficulty, please contact the TAs
    - If any Python library is missing, please report through issues.cse.iitk.ac.in

- Submit on hello.iitk.ac.in a zip file containing the code, scripts, readme explaining your code and clear instructions to run your code, final output images and report (see the next point). Include your files in a folder before compressing it. Name it as GroupXY.zip

- Submit a detailed report of your runs and the results obtained.
    - Include a table with the time split (compute, communication, total)
    - Show a scalability plot of the three test cases (box plot using the 2 runs per test case)
    - Explain observations from your code

- BONUS: Show scalability on the larger dataset and analyze load imbalance. Bonus marks will also be given for efficient code. (15 marks)

- **Deadline: November 10, 2024, 11:59 PM (hard deadline) We will evaluate your code during class hours on November 11 and 12.**

Post your queries/doubts here: https://forms.gle/iAPaW97ABfGxd8wM9

FAQ (will be updated frequently)

1. Can I start the assignment 2 days before the deadline?
   NO, preferably start today!

2. Can I use Python?
   Yes. Please report through issues.cse.iitk.ac.in regarding missing libraries if any.

3. Will MPI work in WSL?
   We have not tested.