CS 677 Assignment 1: Parallel Distributed Axis-aligned Volume Rendering using MPI

Submission deadline: September 12, 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 1 dimension or 2 dimensions (input parameter). Each subdomain will be handled by 1 process to perform the ray casting algorithm as we discussed in the class. 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. In your ray casting algorithm, implement early ray termination and report what fraction of the rays are terminated early. The output of your program is the final volume rendered image. While running your program, users will provide a bounding box (2D) as input parameter such that the output image should be a volume-rendered image only from the specified bounding block. This bounding box may be the entire XY data extent or a subset of it.

Link to test dataset: Isabel_High_Resolution_Raw

Example of domain decompositions:

Domain decomposition along y-axis (left figure)
Domain decomposition along x-axis and y-axis (right figure)

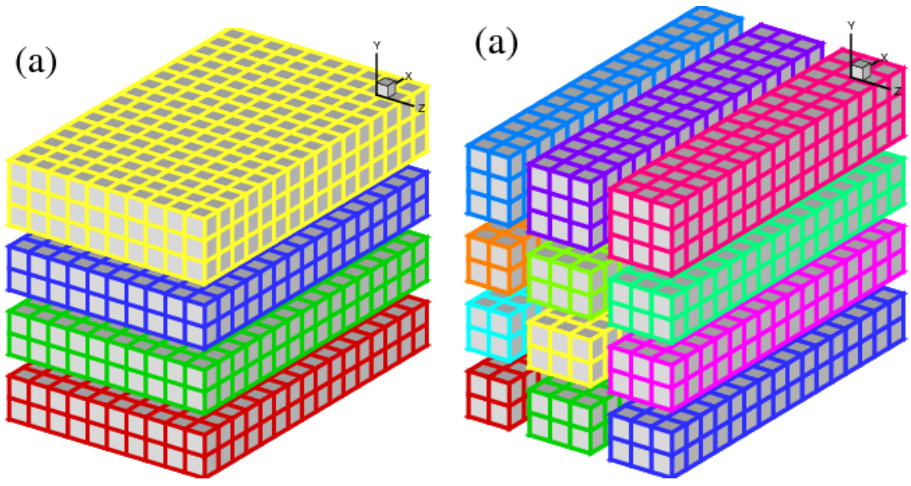

Figure source: hector.ac.uk

Please ignore the axis notations 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
- Output sub-image by each rank
- Stitch the image and generate a png or jpeg file

Input:

- Number of processes (i.e. specified using -np)
- Dataset
- 1D (only X-dimension) or 2D (XY dimension).
- Stepsize (default is 0.5)
- XY bound (default is full data extent)
- Transfer function

Constraints

- In case of 2D partitioning, please decompose in a way such that the number of processes in x-direction is more than the number of processes in the y-direction. For example: 15 = 5 X 3.

Output:

- Every process outputs partial image and you have to combine them to form the final image (this can be done as a post-processing step)
- Output the fraction of the rays that are terminated early
- Output the total time taken (is the maximum time taken by any process)

Test case example:

- mpirun -np 4 ./executable Isabel_1000x1000x200_float32.raw 1 0.75 0 999 0 999

    - executable: name of the executable of your program
    - Name of the input dataset
    - 1 = 1D partitioning (along X or Y), 2 = 2D partitioning (along both X and Y)
    - Integration step size during for ray casting algorithm
    - X_bound_min
    - X_bound_max
    - Y_bound_min
    - Y_bound_max

Execution and submission instructions

- Run your code for four test cases:
    - mpirun -np 4 ./executable Isabel_1000x1000x200_float32.raw 1 0.75 0 999 0 1000

    - mpirun -np 8 ./executable Isabel_1000x1000x200_float32.raw 2 0.25 0 500 0 999
    - mpirun -np 15 ./executable Isabel_1000x1000x200_float32.raw 1 0.5 0 999 0 700
    - mpirun -np 32 ./executable Isabel_1000x1000x200_float32.raw 2 0.35 0 500 0 999
    -
- You may test your code on your laptop/desktop if you have a high-end system
- To run on multiple processes, please follow these instructions
    - Fill this form if you do not have a CSE email ID (deadline: August 25)
    - Enable password-less SSH to 172.27.19.1 - 30 (login to any IP)
    - Login to 172.27.19.1 – 172.27.19.10 (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 – 172.27.19.30)
        - 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

- Submit on hello.iitk.ac.in a zip file containing the code, scripts, readme explaining your code and instructions to run your code, a brief report of your runs and the results obtained. You may time various sections of the code separately and report your observations. The main timing reported should be related to the parallel volume rendering, you may report the data distribution timing separately.
- Deadline: September 12, 2024, 11:59PM (hard deadline)

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.

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