In Situ Visualization and Analysis of Blood Flow Simulation

Connor Murphy, Nazar Tishchenko, Ryan Jeziorski
Department of Mechanical Engineering, Department of Computer Science
Northern Illinois University
Dekalb, Illinois
z1853683@students.niu.edu, z1910995@students.niu.edu, z1878823@students.niu.edu

***Abstract*— Blood flow simulation has come to play an important role in the medical field. High accuracy blood simulations require the visualization of both plasma (fluid) and blood cells (solid). With the recent advances in computational capabilities, a faster analysis of large data sets generated by these simulations is needed. In the past a user must wait for a simulation to complete before analyzing data. If any parameters needed to be changed, the simulation had to be run again with the new set parameters. It is beneficial for engineers to examine simulation results while it is running, and dynamically change settings to see their effects on the results in real time. This is called *in situ* visualization and analysis. This project describes significant steps taken towards the parallelization (allowing for a code to be run on multiple processors) and streamlining of a previously developed framework through coupling fluid flow with blood cell motion for blood flow simulation. The visualization and analysis are achieved though *in situ* software named *SENSEI*.**

I. Introduction

Study of the physical human vascular system presents several challenges such as humanitarian concerns and the difficulty of closely studying a functioning internal system. Because of this, study through simulation has become an important part of the medical field. When it comes to simulating blood flow in the vascular system, large scale simulations are typically accompanied by time consuming post processing of simulated data. A beneficial ability would be to enable *in situ* visualization and analysis of a simulation, e.g., performing analysis and visualization in real time while the simulation is running. This way, the user could quickly view and analyze data without the need to work with written data or having to run a simulation again. The final product will have several features that could have a great impact on medical applications. This includes both real time visualization of simulated data and the ability to manipulate settings while a simulation is running e.g., a surgeon could pinch a blood vessel in a simulated vascular system to see how it would affect surrounding vessels in surgery where he would need to close a given an artery. To accomplish this goal, the *in situ* framework called *SENSEI* [1] has been integrated with a preexisting fluid/solid interaction code that simulates blood cells suspended in plasma [2].

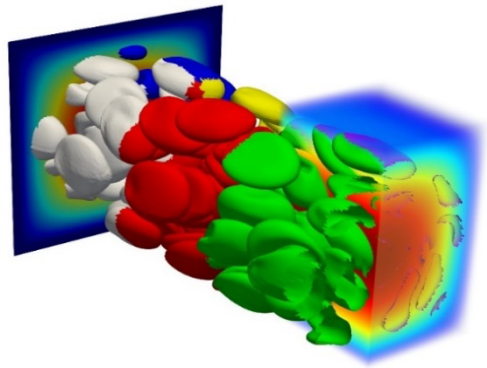

Figure 1 An *in situ* visualization of a blood flow simulation showing how fluid and cells are divided among six processors. Each color on the blood cells indicates a different processor responsible for computing that data. The contour in the back shows a sliced view of fluid velocity in a rectangular channel. Only one 3D subdomain of the fluid velocity is shown in the front for clear visualization.

II. Background

Complex fluid-solid interactions between cells and blood plasma require accurate simulation of the solid and fluid elements. To do this, two software were coupled together, each computing one component of blood. The plasma is handled by the fluid solver named *Palabos* while the cells are handled by the solid solver *LAMMPS*. During Summer of 2020, the foundation of this project was developed by integrating *SENSEI* with an already working blood flow simulation code. The workflow for this is described in Bucaro et al [3]. The project at the end of the Summer allowed for *in situ* writing of data that could be visualized, but the simulating was restricted to being run on one processor due to errors occurring if being run otherwise. Large simulation for vascular systems can take a very long time to run. Without the ability to split up the work among multiple processors, getting meaningful results in a relatively short time from such simulations is impossible. Work was needed to fix the code and allow it to be split up and run in parallel.

III. Fixes to blood cell data

Running a simulation with more than one processor would give cryptic segmentation fault errors which were very challenging to decipher. Testing finally found that cell data such as cell membrane particle positions were being

wrongly set before the time looping starts. It was previously assumed that these variables would be updated dynamically without being explicitly reset. Instead, they needed to be reset by placing them after the *LAMMPS* script is called to update cell data each timestep. After this was completed, running a simulation with four processors produced a full blood cell. However, a small gap existed between the discretized triangles making up the cell membrane that were owned by different processors. The solution was to send data to *SENSE*I and analyze it before running coupling steps that exchange force and velocities between the fluid and cell data. After addressing the above issues, the blood cell data was fully parallelized.

IV. Fixes to fluid data

Data calculated for the fluid includes velocity, vorticity, and normalized velocity. The data returned by the *Palabos* fluid software is stored in *Palabos* defined objects called *MultiTensorField3D* and *MultiScalarField3D,* which aren't recognized by *SENSEI*. Thus, before fluid data gets passed to *SENSEI*, it must be converted to a type called *vtkDoubleArray.* This is done by looping through each index in the given tensor/scalar array and writing it to the correct index in the *vtkDoubleArray*. Previously, each processor calculated the entire fluid domain and passed it to be written as a *vtkDoubleArray*. Now, individual arrays composed of sub domains are sent by each processor with dimensions that include an extra "ghost" layer for the communication between processors. This splits up and minimizes the work each processor must accomplish. The extra "ghost" layer is needed to create a seamless connection between data sets for visualization. However, this layer must be specified as ghost layers, so ParaView does not visualize the overlapping between processor domains.

V. Workflow Improvements

One of the focus areas for this project had to do with ease of use. Several aspects of user experience were considered. The initial installation of all required packages can be frustrating for a new user. To streamline the installation, several scripts have been developed to reduce the number of inputs a user would need to make to download, install, and build the required software needed to run the simulation. It is important to have a user-friendly installation for an open-source project.

VI. Future Work

In its current state, the project can produce *.pvd* files for both solid and fluid meshes, which can be used with Catalyst, a framework from ParaView visualization software for visualizing *in situ* processes. The next logical step is to enable the bidirectional steering of the simulations. Bidirectional steering is a process where the simulation settings can be modified from visualization software and vice versa.

One of the main issues with the current iteration of the project is the lack of optimization. With all factors in mind, while the capability to split up the fluid domain to compute in parallel, one of the potential improvements is adding sparse block structure domain decomposition. This is a smarter way to split up a simulation domain depending on where the most computations are taking place. For more complex vascular networks, a blood vessel will only take up a portion of the domain. With a sparse block decomposition, processors can be given only sections of the domain where the vessel exists while the other sections are ignored. This can increase the efficiency of a simulation. Further improvement can be achieved by utilizing processor load balancing to weigh regions by complexity cost. Furthermore, the dynamic domain decomposition can be used with the above changes to ensure that computational power is focused on the high-cost region. Finally, expanding the scale of the multiphysics simulation for large scale blood flow can be helpful in clinical applications.

VII. Conclusion

In conclusion, the capability for *in situ* visualization and analysis of blood flow simulations could have a large impact on the medical field. Parallelization of this *in situ* project is complete, making it able to run large simulations across many processors. The project is now ready for the next stage which consists of enabling the real-time simulation parameter manipulation

VIII. Acknowledgment

This work was supported by scientists Silvio Rizzi and Victor Mateevitsi from Argonne National Laboratory. This work was also supported by professors Dr. Jifu Tan and Joseph Insley from Northern Illinois University.

IX. References

[1] "The SENSEI generic in situ interface," *2016 Second Workshop on In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization (ISAV),* pp. 40-44, 2016.

[2] J. Tan, T. R. Sinno and L. S. Diamond, "A parallel fluid-solid coupling model using LAMMPS and Palabos based on the immersed boundary method," *Journal of computational science,* vol. 25, pp. 89-100, 2018.

[3] A. Bucaro et al. "Instrumenting Multiphysics Blood Flow Simulation Codes for In Situ Visualization and Analysis," *2021 IEEE 11th Symposium on Large Data Analysis and Visualization (LDAV),* pp. 88-89.