computers

MDPI

Article

An Integrated Open-Source Software System for the Generation and Analysis of Subject-Specific Blood Flow Simulation Ensembles

Simon Leistikow [1,†], Thomas Miro [1,*,†], Adrian Kummerländer [2], Ali Nahardani [3], Katja Grün [4], Marcus Franz [5,6], Verena Hoerr [3], Mathias J. Krause [2] and Lars Linsen [1]

[1] Computer Science Department, University of Muenster, Einsteinstraße 62, 48149 Münster, Germany
[2] Institute of Applied and Numerical Mathematics, Karlsruhe Institute of Technology, Englerstraße 2, 76131 Karlsruhe, Germany
[3] Heart Center Bonn, Department of Internal Medicine II, University Hospital, Venusberg-Campus 1, 53127 Bonn, Germany
[4] Department of Internal Medicine I, University Hospital Jena, Am Klinikum 1, 07747 Jena, Germany
[5] Department of Cardiology, Angiology and Intensive Care Medicine, Cardiovascular Center Hersfeld-Rotenburg, Heinz-Meise-Straße 100, 36199 Rotenburg an der Fulda, Germany
[6] Department of Cardiothoracic Surgery, University Hospital Jena, Am Klinikum 1, 07747 Jena, Germany
* Correspondence: tmiro@uni-muenster.de
† These authors contributed equally to this work.

Abstract

Hemodynamic analysis of blood flow is critical for diagnosing cardiovascular diseases and investigating cardiovascular parameters, such as aneurysms and wall shear stress. For subject-specific analyses, the anatomy and blood flow of the subject can be captured non-invasively using structural and 4D Magnetic Resonance Imaging (MRI), respectively. Computational fluid dynamics (CFD), on the other hand, can be used to generate blood flow simulations. To generate and analyze subject-specific blood flow simulations, MRI and CFD have to be brought together. We present an interactive, customizable, and user-oriented visual analysis tool that integrates measured data and CFD simulations. Thus, our open-source tool supports both medical and numerical analysis workflows. It enables the creation of simulation ensembles with a high variety of parameters. Furthermore, it allows for visual and analytical examination of simulations and measurements through 2D embeddings. To demonstrate the effectiveness of our tool, we applied it to three real-world use cases, showcasing its ability to configure simulation ensembles and analyze blood flow. We evaluated our example cases together with MRI and CFD experts. By combining the strengths of both CFD and MRI, our tool provides a comprehensive understanding of hemodynamic parameters, facilitating accurate analysis of hemodynamic biomarkers.

Keywords: flow visualization; ensemble visualization; cardiovascular imaging; simulation ensembles; Voreen; OpenLB

Academic Editor: Paolo Bellavista

Received: 16 March 2026
Revised: 23 April 2026
Accepted: 7 May 2026
Published: 9 May 2026

1. Introduction

Numerical simulations are essential for understanding complex phenomena across scientific disciplines, including medicine. In hemodynamics, computational fluid simulations help investigate the blood flow within blood vessels. Configuring these simulations can be tedious, especially when trying to assimilate measured data, as setting boundary conditions and ensuring stability demand a careful setup. Moreover, medical imaging data

must first be transformed into 3D surface models, typically requiring multiple separate software tools.

Integrating these steps into a single application can improve productivity and flexibility. Yet, existing commercial and open-source solutions often target specific use cases with limited adaptability, restricting their applicability. In Section 2, we compare state-of-the-art software tools for (medical) flow simulation and analysis, highlight their strengths and weaknesses, and point out a resulting gap in medical simulation research.

In particular, generating (subject-specific) simulation ensembles remains cumbersome, and integrated support for analyzing such data is often missing. Researchers frequently resort to custom scripts, limiting reproducibility and generalizability. This highlights the need for a unified application that efficiently combines specialized tools for flow simulation and analysis.

We address this by integrating our Lattice Boltzmann simulation framework OpenLB [1] into Voreen [2], our visualization and analysis framework. Our tool combines the high-performance simulation capabilities of OpenLB with Voreen's customizable workflows. An overview of both frameworks is given in Section 4 after providing the background in Section 3.

To demonstrate the tool's capabilities, we define research use cases from CFD and cardiovascular imaging, developed in collaboration with domain experts. Based on these, we build user-friendly and customizable workflows in Voreen to address the use cases (Section 5) and evaluate them through expert feedback (Section 6).

The individual contributions of our work can be summarized as follows:

- We present an extensible, customizable, and integrated system with a user-friendly graphical user interface (GUI) to configure, perform, and analyze three-dimensional Lattice Boltzmann simulations.
- We demonstrate the effectiveness of our system in three research-oriented use cases:
  - (A) The configuration of simulations based on a given geometry stored in a stereolithography (STL) file.
  - (B) The preparation and configuration of subject-specific simulations based on medical imaging data.
  - (C) The visual analysis of the subject-specific simulation ensemble created in (B).
- We evaluate our software with feedback from researchers from the CFD and cardiovascular imaging domain, respectively.

2. Review of Existing Software Solutions

Our tool for the generation of simulation ensembles and their visual data analysis relates to research in computational fluid dynamics, cardiovascular imaging, and visualization. While these fields have specific challenges, they share methods like generating and visualizing velocity vector fields. Consequently, established frameworks face both domain-specific and overlapping issues.

Numerous frameworks and comparative studies exist, e.g., [3,4]. Voreen, for instance, has been compared to similar tools [2], but only regarding visualization. In the following, we review existing software solutions for fluid simulations and their analysis. Since the range of available tools is broad, we focus on the ones most comparable to our system in scope and functionality. To ensure objectivity, we define categories for evaluation. Key factors for existing solutions are availability, usability, and supported features. Availability covers licensing model (open-source vs. commercial) and platform compatibility. Usability includes customization options, required expertise (e.g., CFD or MRI knowledge), documentation, and user support. Comparing supported features is more complex. For our purposes, relevant aspects include 3D model generation from volumetric medical imag-

ing data, simulation and ensemble creation, analysis, and postprocessing functionalities. Table 1 summarizes these categories and criteria.

Table 1. Comparison categories and factors for evaluating biomedical simulation software.

Category	Comparison Factors
Availability	Licensing model (open source vs. commercial)
	Supported operating systems
Usability	Required prior knowledge (e.g., CFD, MRI)
	Customization options
	Availability of documentation
	User support (e.g., community forums)
Supported Features	3D model generation from imaging data (Preprocessing)
	Simulation and ensemble creation
	Simulation and ensemble analysis
	Graphical User Interface (GUI)
	Interactive Visual Data Analysis (Postprocessing)

SimVascular (v2025.08.25) [5,6] is an open-source software designed as a complete pipeline for blood flow modeling, from medical image segmentation to hemodynamic simulations. It offers functionalities for preprocessing, including image segmentation, discrete solid modeling, mesh repair tools, and fluid–solid interaction modeling with variable wall properties. Additionally, it provides a highly configurable simulation setup, supported by well-structured documentation, making it accessible for detailed hemodynamic studies. However, visualization and analysis are outsourced to ParaView, making multi-simulation analysis tedious due to manual export/import across tools. Furthermore, SimVascular is limited to static flow fields, since time-dependent data can not be interpreted.

CRIMSON (v2023.06.03) [7,8] is an open-source software designed as an advanced simulation environment for subject-specific hemodynamic analysis. Developed as a descendant of SimVascular, it shares a common foundation, particularly in medical image segmentation capabilities. However, its development has emphasized computational hemodynamics (CH), incorporating a wide range of adjustable boundary conditions to enhance simulation accuracy. Additionally, unlike SimVascular, CRIMSON integrates visualization and postprocessing within its own graphical user interface, which is available on Linux and Windows. Despite its strengths, CRIMSON targets individual simulations and lacks built-in support for parallel ensemble analysis in its GUI, limiting its use for parameter studies and sensitivity analyses.

SimScale (accessed in April 2026) [9,10] is a cloud-based simulation platform designed for a range of engineering applications. It is a commercial, web-based solution that provides a user-friendly interface. In the context of (multiphase) flow simulations, SimScale allows users to perform geometric modeling, configure flow properties, and define boundary conditions. Additionally, it offers postprocessing tools for visualizing individual simulations. The platform enhances usability through tutorials and professional customer support. However, it lacks support for simulation ensembles with varying parameters and built-in comparative analysis of multiple simulations on the same object. Additionally, it does not preprocess medical imaging data (DICOM, NIfTI). While external tools can be used to convert medical data into Computer-Aided Design (CAD)-compatible formats, this requires additional expertise, making the workflow less accessible to users from the medical domain.

Comparable commercial software such as Ansys [11] also provides fluid flow simulation capabilities. Since its advantages and limitations closely align with those of SimScale in the context of our comparison, we only mention it briefly here.

FabSim3 (v3.79) [12,13] is an automation toolkit designed for verified simulations using high-performance computing. At its core, it is a Python-based framework that allows users to write custom scripts for running simulation ensembles and analyzing their outputs with various metrics. Automating key steps in pre- and postprocessing, it minimizes manual errors and reduces the workload associated with managing large-scale simulations. Additionally, it is supported by a documentation that details its functionalities. However, it does not include a GUI, making it less accessible to users without a background in computer science or related fields. Furthermore, while it offers a high degree of automation, its parameter configuration options are more constrained compared to other solutions, which may limit flexibility in the simulation setup.

OpenFOAM (v13) [14–16] is an open-source C++ library and one of the most widely adopted computational mechanics software solutions. Compared to FabSim3, it provides a more modular approach, allowing users to fine-tune a wide range of settings for both pre- and postprocessing. It also supports the computation of simulation ensembles with varying boundary conditions through user-written scripts. Additionally, OpenFOAM has gained significant popularity, leading to an active community of contributors and users, which ensures continuous development and improvement. However, as a library, OpenFOAM lacks a GUI, reducing accessibility for non-experts in computer science (or related fields). Unlike FabSim3, it requires extensive CFD knowledge due to minimal automation and manual configuration. Visualization relies on external tools like ParaView.

Voreen (v5.3.0) [2,17] is an open-source framework for the interactive visualization and analysis of volumetric data. It offers highly configurable analysis and visualization workflows. Within these workflows, volume processing pipelines can be built and executed. The framework supports medical image processing and visualization as well as the connection to computing clusters for expensive tasks such as flow simulations. Moreover, it provides a GUI, allowing users to visualize processing and simulation results in various ways and, thus, enhances their visual analysis. A more rigorous description is provided in Section 4. However, Voreen offers limited support for geometric modeling. Although it includes basic smoothing and hole-filling algorithms, most modeling tasks are delegated to external CAD software. Moreover, boundary conditions are auto-detected via vessel graph extraction and are only slightly adjustable. Further limitations are discussed in Sections 6 and 7.

Table 2 summarizes supported features. Availability and usability are excluded, focusing solely on technical capabilities for simulation and visualization. Each reviewed software has strengths and limitations for different user groups. Our system proposed in this paper combines Voreen with OpenLB (see Section 4) and is, therefore, labeled "Voreen + OpenLB" in Table 2. Our solution enables ensemble simulations and analysis in a user-friendly environment requiring minimal prior knowledge. Integrating all steps into one system removes the need for external software and manual data transfers, filling a gap in medical simulation research.

Table 2. Overview of supported features in different simulation and visualization software solutions. The feature classification into "fully supported", "partially supported", and "not supported" is a simplification to aid comparison. Actual capabilities may vary depending on use cases, configurations, or ongoing developments.

Feature	Voreen + OpenLB	SimVascular	Crimson	SimScale	FabSim3	OpenFOAM
Medical Image Segmentation	●	●	●	○	◑	○
Geometric Modeling	◑	●	◑	●	○	●
Configure Boundary Conditions	◑	●	●	●	◑	●
Running Simulations	●	●	●	●	●	●
Running Simulation Ensembles	●	●	◑	◑	●	●
In-Situ Analysis of Simulations	●	○	◑	●	◑	◑
Postprocessing and Visualization	●	◑	●	●	◑	◑
Open Source	●	●	●	○	●	●

Legend: ● = fully supported, ◑ = partially supported, ○ = not supported.

3. Background on Flow Simulation Ensembles

Visualization, simulation, and analysis often depend on multiple parameters. Instead of examining them separately, simulation ensembles allow for analyzing combined parameter settings [18,19]. In our work, we define a flow simulation ensemble as a set of simulated time-resolved 3D flow vector fields as in [20]. Each individual simulation is called an ensemble member and has its own simulation-specific parameter configuration. Such configurations are often stored in the file format's metadata.

The macroscopic motion of fluids is commonly described using the Navier–Stokes equations (NSE). We use the force-free incompressible NSE

$$\begin{cases} \partial_t u + (u \cdot \nabla)u = -\frac{1}{\rho}\nabla p + \nu \Delta u & \text{in } \Omega \times I \\ \nabla \cdot u = 0 & \text{in } \Omega \times I \end{cases} \tag{1}$$

for velocity u, pressure p, density $\rho > 0$, and kinematic viscosity $\nu > 0$ on spatial domain $\Omega \subseteq \mathbb{R}^3$ and time $I \subseteq \mathbb{R}_{>0}$. This macroscopic model is sufficient for hemodynamic simulations, as blood can be assumed to be incompressible under all physiological conditions. Depending on the vessel size, it can even be assumed as a Newtonian fluid with constant viscosity. For smaller vessels or low shear rates, the constant viscosity ν can simply be replaced by a suitable shear-thinning viscosity law. The corresponding simplifications and assumptions are typical for this purpose, cf. [21,22].

The Lattice Boltzmann method (LBM) is used to approximate the NSE (1) target equation on a regular space–time grid with the $D3Q19$ velocity stencil (cf. Figure 1).

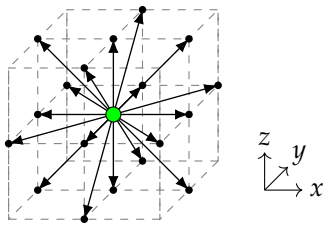

Figure 1. The discrete velocity set $D3Q19$.

Specifically, we are using the Lattice Boltzmann equation (LBE) with the Bhatnagar–Gross–Krook (BGK) collision operator. This BGK-LBE is given by

$$\begin{aligned} f_i^{\text{post}}(x,t) &:= f_i(x,t) - \frac{1}{\tau}\Big(f_i(x,t) - f_i^{\text{eq}}(x,t)\Big), \\ f_i(x + \xi_i \triangle t, t + \triangle t) &= f_i^{\text{post}}(x,t) \end{aligned} \tag{2}$$

for distribution functions f_i along q discrete velocities ξ_i on a regular lattice $\Omega_{\triangle x} \subset \Omega \subseteq \mathbb{R}^3$ with cell size $\triangle x$ at discrete times $I_{\triangle t} \subset I \subseteq \mathbb{R}_{\geq 0}$ separated by step size $\triangle t$. The equilibrium distribution f_i^{eq} is relaxed towards with relaxation time $\tau > 0.5$ and defined as

$$f_i^{\text{eq}}(\rho, u) := \rho\omega_i\left(1 + \frac{\xi_i \cdot u}{c_s^2} + \frac{(\xi_i \cdot u)^2}{2c_s^4} - \frac{u \cdot u}{2x_s^2}\right)$$

for lattice weights ω_i, lattice speed of sound c_s, and macroscopic density and velocity moments ρ and u, respectively. These moments are obtained from the distribution functions f_i using

$$\rho(x,t) = \sum_{i=0}^{q-1} f_i(x,t),$$
$$u(x,t) = \frac{1}{\rho(x,r)} \sum_{i=0}^{q-1} f_i(x,t)\xi_i.$$

In some cases, we additionally apply a Smagorinsky-type large eddy simulation (LES) model in order not to resolve smaller eddies. For the BGK-LBE, this is realized by replacing the constant relaxation time τ with an effective relaxation time τ_{eff}, locally computed by

$$\tau_{\text{eff}}(x,t) = \frac{\nu_{\text{mol}} + \nu_{\text{turb}}(x,t)}{c_s^2}\frac{\Delta t}{(\Delta x)^2} + \frac{1}{2},$$
$$\nu_{\text{turb}}(x,t) = (C_S \Delta x)^2 \overline{\Pi}(x,t)$$

for molecular viscosity ν_{mol}, Smagorinsky constant $C_S \geq 0$, and shear rate $\overline{\Pi}$ (i.e., the second order non-equilibrium moment).

Connecting the BGK-LBE (2) to the target equation, it can be shown to yield a second-order approximation of the NSE (1) in space [23]. Considering the BGK-LBE (2) from an algorithmic perspective, we can see a clear separation into a local collision and a non-local propagation part (cf. Figure 2). This is the property that renders LBMs uniquely suited to massively parallel execution on modern high-performance computers (HPCs).

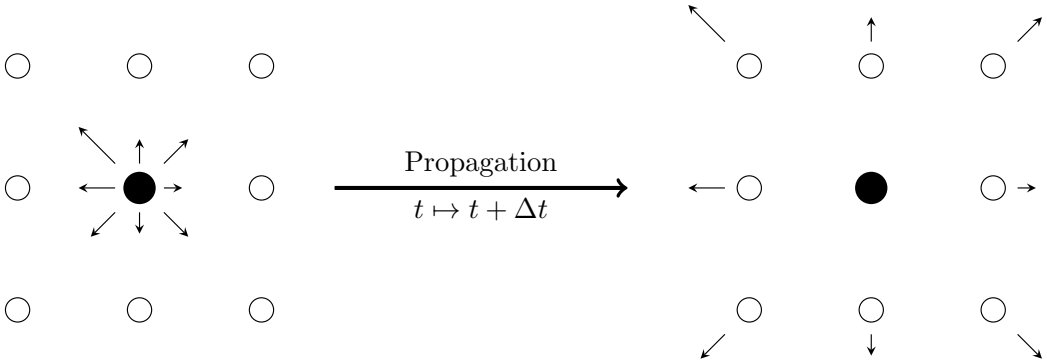

Figure 2. Neighborhood-local propagation of distribution functions for the D2Q9 stencil.

Boundary conditions depend on the type of domain boundary. At inlets, velocity Dirichlet conditions are imposed using prescribed profiles, e.g., Poiseuille. At outlets, constant pressure boundary conditions are applied. Vessel walls are modeled using no-slip boundary conditions, realized via local or interpolated bounce-back schemes.

The spatial and temporal resolutions are determined by the lattice spacing Δx and time step Δt, which are chosen based on the geometry resolution and flow regime. To ensure numerical stability, the condition

$$\frac{1}{3}\frac{\rho}{\nu}u(\tau - 0.5)\Delta x < 0.4$$

must be satisfied [23].

For more information about LBM and solution approaches, we refer to the literature [23,24].

4. Our Integrated Software System

As discussed in Section 2, existing solutions address parts of the challenges in creating, visualizing, and analyzing simulation ensembles, but lack an interactive and customizable tool tailored for CFD and cardiovascular imaging experts. We address this by integrating the high-performance Lattice Boltzmann framework OpenLB into the interactive visual analysis framework Voreen.

4.1. Voreen

Voreen [2,17] is a C++ open-source framework for interactive visualization and analysis of multi-modal volumetric data. It provides volume rendering, interactive data analysis, and configurable analysis workflows. It is maintained in a public repository on GitHub (https://github.com/voreen-project/voreen accessed on 15 April 2026) and available for Windows and Linux. In the following, we first present existing relevant features of Voreen and afterwards the newly added or significantly revised features relevant to our use cases in Section 5. A more detailed feature overview is given by Drees et al. [2].

Voreen already provided volume processing tools such as iso-surface extraction, connected component analysis, centerline extraction, and semi-automatic random walker segmentation. Scalar volume data can be visualized using, for example, slice-based or direct volume rendering by (out-of-core) raycasting. We added and revised

- Support for export of common geometric file formats, i.e., Wavefront Object Files (.obj) and Stereolithography Files (.stl);
- Streamline and pathline rendering of 3D vector fields;
- A hole-filling algorithm for the generation of watertight meshes;
- Voxelization of geometries via inside–outside tests;
- Support for SLURM-based high-performance computation (HPC) clusters using SSH;
- Integration of OpenLB;
- Support for import and export of VTK [25] file formats, i.e., .vti, .vtm, .pvd;
- Documentation on GitHub;
- Customizable workspaces that combine the above-mentioned points with the already existing features; see Section 5.

4.2. OpenLB

OpenLB [1,26] is an open-source C++ framework for multi-physics simulations using LBM. It provides a broad suite of models enabling transparent execution on both CPU and GPU architectures [27–29].

LBM's computational efficiency extends to automated meshing due to its regular lattice structure [30]. OpenLB uses constructive solid geometry via indicator arithmetic to describe complex simulation geometries [1], simplifying meshing, domain decomposition, and load balancing (cf. Figure 3).

The meshing process produces voxelized, decomposed material geometries, assigning local cell models and boundary conditions. Dynamics group moment systems, equilibrium

distributions, and collision operators into local models, while post-processors handle non-local boundary treatments. Parameters and data are declared via fields, with per-cell values defined through functors, which are also used for postprocessing outputs like VTK [25].

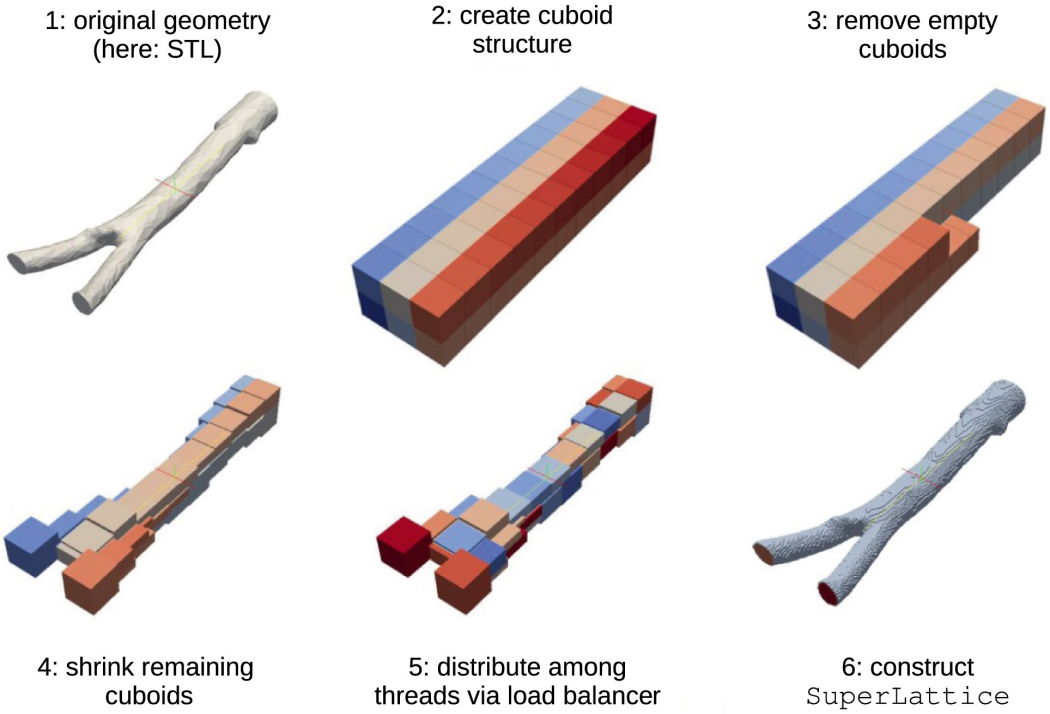

Figure 3. Automated meshing and decomposition [1].

4.3. Integration

We integrate OpenLB into Voreen as a flexible backend module, reading simulation parameters from Voreen's XML parameter file. This case is compiled once with a suitable choice of parallelization modes and then executed via the Voreen UI. As long as the XML interface is not modified, this allows for independent updates to both the Voreen front-end and the OpenLB simulation back-end. Recompilation of both front- and back-end is only required if the interface changes, i.e., when new simulation models, boundary conditions, or parameters are implemented.

This integration extends Voreen's feature set with fluid flow simulation capabilities. Together, Voreen and OpenLB provide a comprehensive tool for volumetric data processing. Figure 4 illustrates workflows emphasizing Voreen's unique selling point: all pipeline steps—from segmentation to analysis—are performed within a single software, allowing iterative adjustments of any pipeline step with small effort.

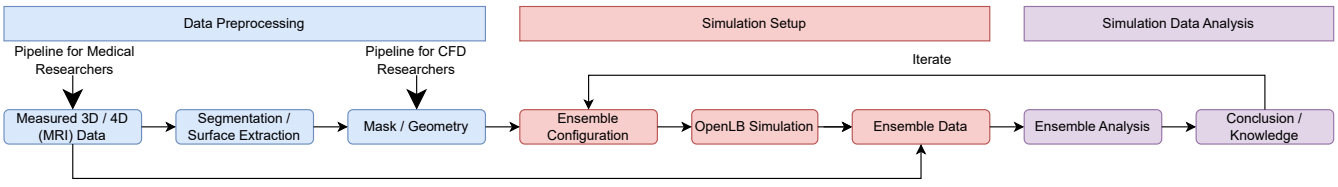

Figure 4. Ensemble Analysis Pipeline: Processing and configuration of simulation ensembles as well as their analysis.

5. Results

This section introduces three use cases for typical CFD and MRI tasks, demonstrating the setup and analysis of simulation ensembles with interactive visualization

techniques [31]. Unlike Leistikow et al., who emphasized technical parameters, we focus on providing a practical user guide. The use cases cover (A) configuring ensembles from STL files, (B) generating 3D models and ensembles from MRI data, and (C) analyzing ensembles from (B) to derive insights. For reproducibility, the three use cases have been added to the open-source GitHub repository of Voreen. Detailed guides are available in the GitHub wiki for OpenLB Use Case A https://github.com/voreen-project/voreen/wiki/OpenLB-UseCase-A, OpenLB Use Case B https://github.com/voreen-project/voreen/wiki/OpenLB-UseCase-B and OpenLB Use Case C https://github.com/voreen-project/voreen/wiki/OpenLB-UseCase-C (accessed on 15 April 2026). Furthermore, in the Supplementary Material, we provide tutorial videos for all use cases. All reported computation times were measured on a workstation equipped with an Intel i7-6850K CPU with 12 cores, 32 GB RAM, and an NVIDIA GeForce GTX 1080 GPU. The simulation output has been written to a HDD with 100 MB s^{-1} average linear write speed.

5.1. Use Case A: Simulation for Given Geometry

CFD researchers often work with geometry or volume files, such as blood vessels, stored in STL format [32]. Challenges such as boundary condition setup and processing are typically handled in a trial-and-error iteration of parameter selection and result analysis. In situ visualization during runtime can accelerate this process by enabling early aborts and reconfiguration. Use Case A implements this workflow, with user interaction reduced to the following steps:

- Open the predefined workspace use_case_A.vws and select a geometry file from the Input Geometry panel.
- Configure and adjust simulation parameters using the Inlet/Outlet and Simulation Setup panel. More specifically, inlets and outlets are automatically identified by a vessel network analysis [33]. The method relies on a centerline extraction, and the accuracy of inlet and outlet detection depends on the quality of the extracted centerline. However, for tubular structures, the centerline extraction generally works well. The underlying algorithm provides parameters (e.g., bulge size) that can be tuned to improve robustness for specific datasets. Moreover, the framework is interactive, i.e., detected inlets and outlets can be manually adjusted, e.g., by prescribing velocity profiles, position, target velocity, and radius. Also, additional boundary conditions can be added or removed as needed. This ensures reliable simulation setups. In our case studies, the automatic detection yielded satisfactory results without requiring manual correction.
- Initiate the calculation process by using the Run Simulation function, and save the results in a preferred format and folder using the Results function.

Figure 5 shows a sample result for this simulation, describes the parameter setup, and illustrates the configurable parameters available in the Simulation Setup panel. Multiple linked views visualize the geometry and streamlines (top) and the axial velocity magnitude map (bottom left). The user can sketch a line in the latter view for which the velocity magnitude profile is rendered in the bottom right view. Tabs in the bottom left view allow switching between visualizing the material volume used by OpenLB for setting boundary conditions, the velocity magnitude, and an alternative output, such as the pressure field. Plots are updated in real time as the simulation is running, i.e., as soon as a new time step has been written.

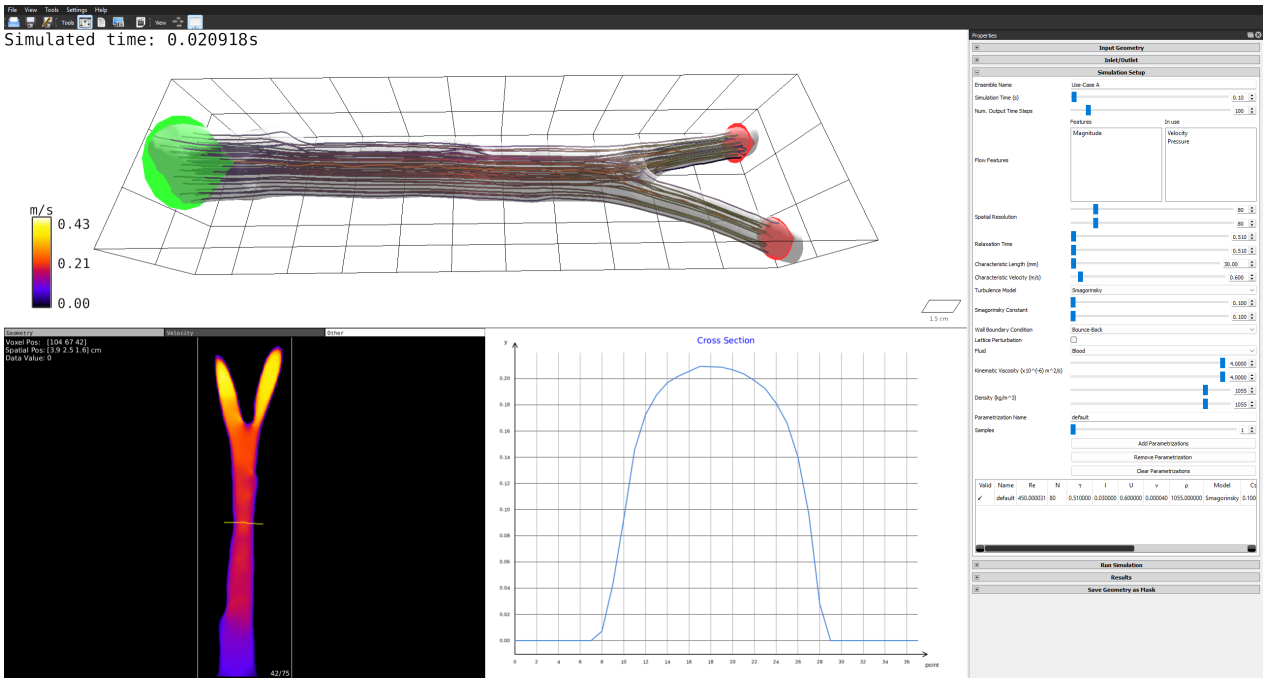

Figure 5. In situ simulation result for Use Case A. The multiple views visualize the geometry, inlets (green cones), outlets (red cones) and streamlines (**top**), the axial velocity magnitude (**bottom left**) and the velocity magnitude profile for a sketched line in the axial view (**bottom right**). The simulation was performed over a physical time of 0.10 s with 100 equidistant output steps. The spatial resolution corresponds to 80 lattice nodes per characteristic length. The simulation employed a relaxation time of $\tau = 0.51$, a characteristic length of $l = 30\,\text{mm}$, and a characteristic velocity of $u = 0.6\,\text{m/s}$. The kinematic viscosity was set to $\nu = 4 \cdot 10^{-6}\,\text{m}^2/\text{s}$ and the density to $\rho = 1055\,\text{kg/m}^3$. The simulation was carried out using a Smagorinsky turbulence model and bounce-back boundary conditions at the vessel walls. The computation for this one simulation took about 25 min.

5.2. Use Case B: Subject-Specific Simulation for Given Medical Imaging Data

Cardiovascular imaging researchers working with 4D-PC-MRI data obtain time series of 3D velocity fields. Here, 12 velocity maps of a rat's pulmonary artery, sampled at 10 ms intervals, were stored in VTI file format. The goal is to reproduce these measurements with simulations: while MRI data typically suffers from noise and low spatial and temporal resolution, simulations can provide noise-free, high-resolution fields but require modeling assumptions. To account for these, we generate simulation ensembles with varying parameters and compare results in Use Case C. In Use Case B, the measured velocity is used as the inlet boundary condition, and the simulation evolves from there. Two workflows were developed to preprocess such data and perform simulations, involving the following steps:

- For preprocessing, open the workspace use_case_B_segmentation.vws and select the measured data using the Input panel. The Data Settings panel allows the user to adjust the data layout, if needed, and to enable aggregation in the time domain. Since the segmentation works best with high-contrast images, a single time step might not be suited to obtain the segmentation of the desired vessel's lumen. Depending on the imaging method used, parts of the vessel's lumen may only be visible in a specific time step, while other parts are visible in different time steps. Hence, we allow the user to aggregate along the temporal axis, e.g., using the maximum, mean voxel value, and the sum. The 2D Rendering panel allows adjusting the transfer function.
- Using the mouse, foreground and background seeds can be placed interactively by the user, which are fed into the random-walker segmentation algorithm [34]. In our example, we choose to only segment the pulmonary trunk contained in the measurement.

- In the Output panel, the segmentation mask can be stored in various file formats, such as VTI.
- Next, open the workspace use_case_B_simulation.vws providing a similar workflow to Use Case A, and load both the created segmentation and measured velocity data. For each voxel of the meshed lattice that happens to be inside the inlet boundary condition region, the measurement is sampled trilinear in space and linear in time. For those outside the inlet boundary condition, they are set to 0. Physical consistency is ensured by the simulation itself.
- Proceed with configuring parameters, running the simulation, and saving the results, similar to Use Case A in the Inlet/Outlet, Simulation Setup, Run Simulations, and Results panels. In contrast to Use Case A, we use the measurement as initialization for the velocity inlet boundary condition and run an ensemble with three members with different inlet velocities.

Figure 6 shows a sample result for the segmentation. The upper-left view shows the 3D segmentation result, which is complemented by further views that display axial, coronal, and sagittal planes. Figure 7 describes the parameter configuration and displays one of the calculated simulations, where the workspace is analogous to the one for Use Case A, cf. Figure 5.

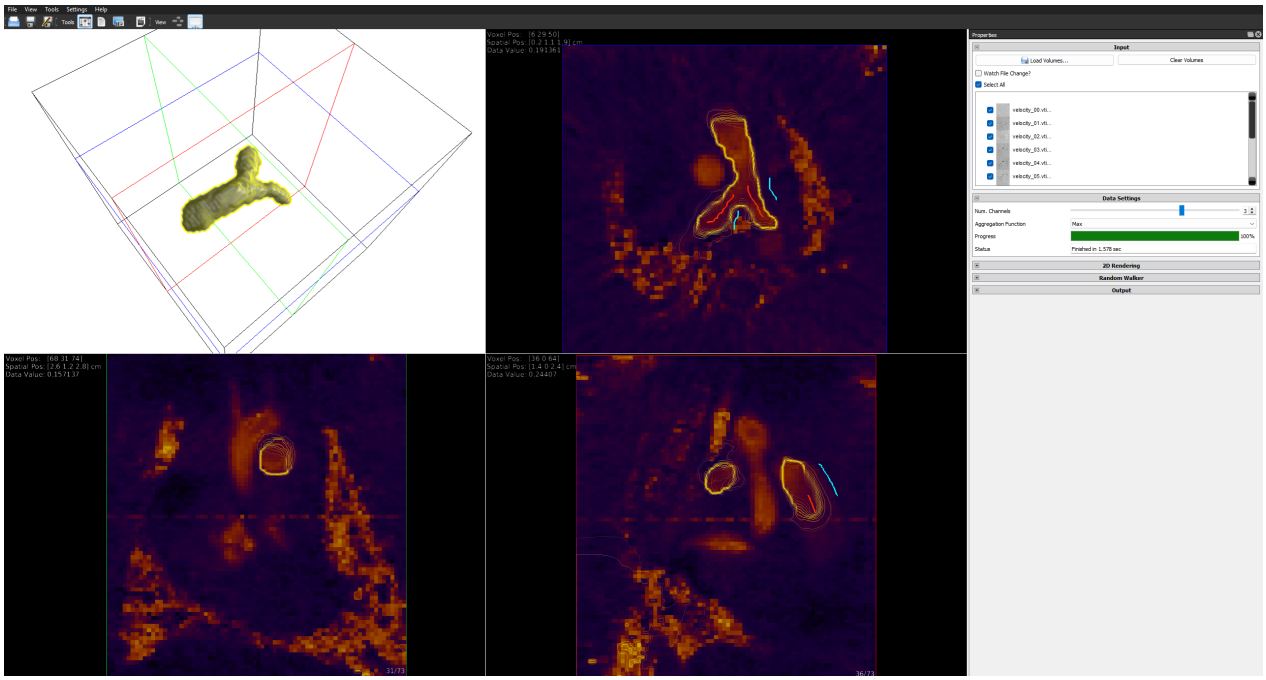

Figure 6. Segmentation result for Use Case B. The 3D rendering (**upper left**) of the segmentation result (yellow) is complemented by further views that display axial (**upper right**), coronal (**bottom left**), and sagittal (**bottom right**) planes. Foreground and background labels (red and cyan, respectively) can be placed interactively by the user.

5.3. Use Case C: Analysis of Simulation Ensembles

In both domains, analysis depends on the task at hand. Typical questions include assessing the effect of specific parameters (e.g., velocity magnitude) or comparing simulations with real-world measurements to identify the best-fitting parameters. Similarity measures for this purpose have been presented in [31]. To apply them, researchers can perform the following steps:

- Open the workspace use_case_C.vws and select the simulation ensemble that has been created with either Use Case A or B using the Input panel.

- Next, the Similarity Plot and Ensemble Variance may be used to obtain an overview of the ensemble. Two members, e.g., the measurement and a simulation similar to it, can be selected from the Similarity Plot and compared directly by visualizing their voxel-wise difference. It is represented both as raycasting and as slice-rendering for the purpose of reading off exact voxel values. The Similarity Plot is obtained through a vector-field similarity measure proposed by Leistikow et al. [31]. In this approach, the dissimilarity between two vector fields is computed by combining differences in magnitude and direction. For the difference in direction, the angular deviation between corresponding velocity vectors is evaluated and averaged over the spatial domain, yielding a normalized dissimilarity measure in the range $[0, 1]$. It is evaluated for each pair of time steps of each simulation or measurement and stored in a similarity matrix. This matrix is then projected into a single dimension using multi-dimensional scaling (MDS) and plotted over time to obtain a Similarity Plot where the vertical distance between runs represents their dissimilarity for a selected point in time. For more details, we refer to Leistikow et al. [31].

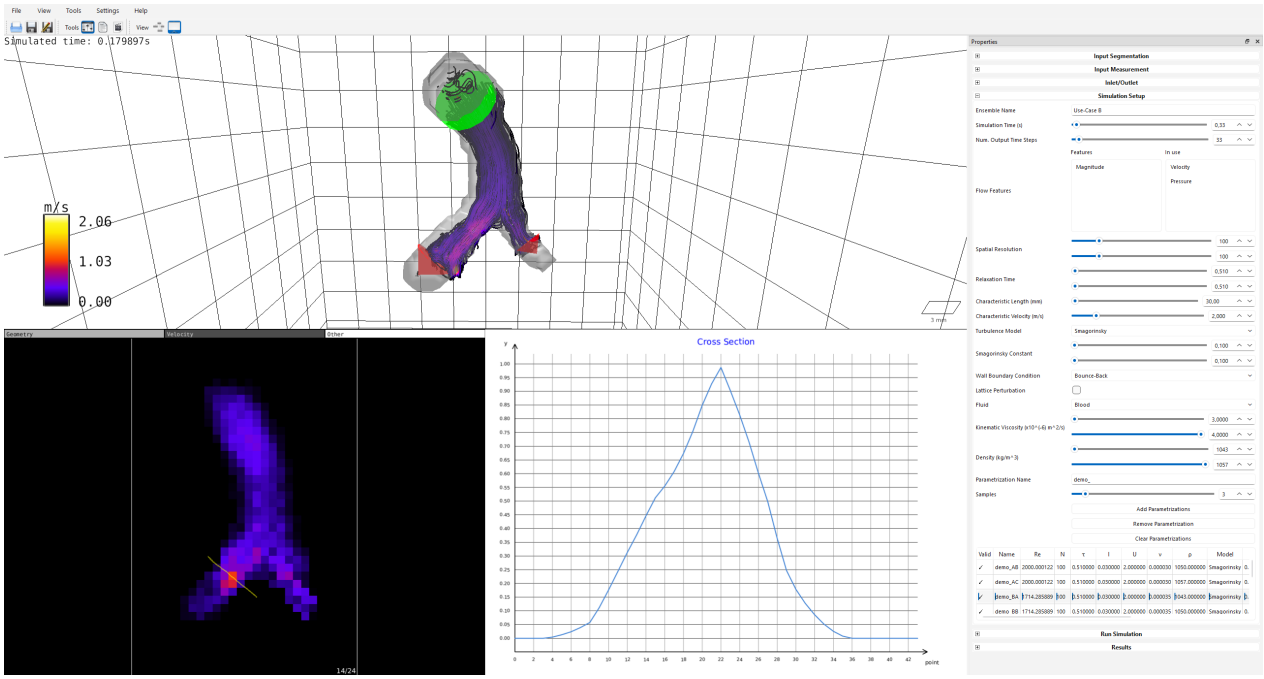

Figure 7. Simulation result for Use Case B. The views are structured as before in Figure 5. The simulations were performed over a physical time of 0.33 s with 33 equidistant output steps. The spatial resolution corresponds to 100 lattice nodes per characteristic length. The simulations employed a relaxation time of $\tau = 0.51$, a characteristic length of $l = 30\,\text{mm}$, and a characteristic velocity of $u = 2\,\text{m/s}$. The kinematic viscosity was equidistantly sampled 3 times between as $\nu = 3 \cdot 10^{-6}$ and $\nu = 4 \cdot 10^{-6}\,\text{m}^2/\text{s}$, while the density was between $\rho = 1043\,\text{kg/m}^3$ and $\rho = 1057\,\text{kg/m}^3$. The simulation was carried out using a Smagorinsky turbulence model and bounce-back boundary conditions at the vessel walls. The computation for an ensemble of 9 simulation runs took about 3 min.

Figure 8 shows the ensemble visualization with our configuration settings. Multiple linked views visualize the Similarity Plot (top left) of all members visualized in different colors, the deviation of all members raycasted (top right), and the deviation of the two selected members raycasted (bottom left) as well as slice-rendered voxel-wise difference (bottom right). The user can change the selected members in the Similarity Plot. Tabs in the top-right view switch between member deviation, connected components, and streamlines.

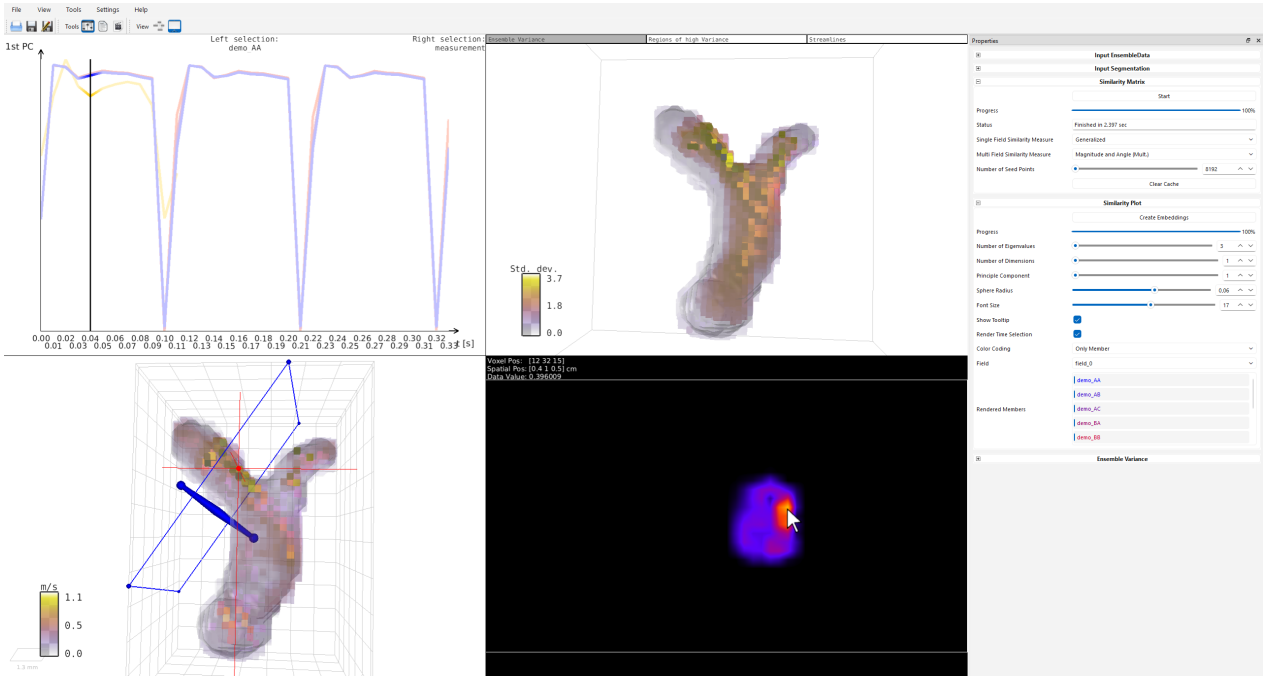

Figure 8. Ensemble visualization with multiple linked views for Use Case C. They visualize the 2D Similarity Plot of all members (**top left**), the deviation of all members raycasted (**top right**), and the deviation of the two selected members raycasted (**bottom left**) and the slice-rendered voxel-wise difference (**bottom right**). In the Similarity Plot, the measurement is yellow, while the remaining colors represent the different simulation members. Due to overlap, not all members are fully visible. Nevertheless, the plot still reveals where the simulations match the measurement and where they deviate.

6. Evaluation and Discussion

To evaluate our software, we conducted two workshop sessions with three researchers. Session S1 involved a master's student in Fluid–Structure Interaction (P1) and a doctoral student in CFD (P2), while Session S2 included a postdoctoral researcher in cardiovascular imaging (P3). Each session covered the full data pipeline (Figure 4), with S1 focusing on Use Cases A (Section 5.1) and C (Section 5.3), and S2 on Use Cases B (Section 5.2) and C. Each session ended with discussions on the comparison factors (Table 1), the system's utility for the participants' work, and potential improvements. We summarize their feedback below.

6.1. CFD Domain

Participants P1 and P2 saw a high potential in Use Case A, especially in the rapid configuration of inlet and outlet boundary conditions via Voreen's GUI. It accelerates the simulation setup and supports collaboration with non-coding researchers. They also valued the intermediate visualization of computational steps for analyzing results and quality.

For experienced users, automatic boundary placement at vessel endpoints was seen as too restrictive. Both suggested enabling variable boundary positioning within the GUI, with P2 stressing this would be a major improvement given the time-consuming nature of such tasks.

Regarding Use Case C, both participants considered it irrelevant for their workflows. CFD research typically relies on standard benchmarks, and Use Case A is aligned with these.

6.2. Cardiovascular Imaging Domain

P3 recognized the value of the integrated data pipeline, highlighting the performance of image processing, simulation configuration, and result analysis within a single software environment. Similar to P1 and P2, P3 also noted the advantages of the GUI-based bound-

ary condition configuration, which facilitates efficient creation of simulation ensembles. Moreover, he evaluated the presented segmentation results as being of good quality.

From P3's perspective, Use Case C contributed to the simulation evaluation by enabling direct visual comparison between measured and simulated data. For instance, we created a simulation intended to replicate flow measurements in a rat's pulmonary artery. Initial results appeared promising. However, a critical vortex structure observed in the measurement was absent in the simulation, leading to a classification of the simulation as inadequate.

P3, an experienced SimVascular and Voreen user, compared our system to it during Session S2. He noted that both platforms support volume rendering, but SimVascular provides different simulation configurations, such as local mesh-size adaptation and boundary condition refinement. While Use Case C offered useful visualizations, P3 found them limited to specific tasks. For the variety of fluid mechanics conditions in the human body, such as bifurcations, branching, curvatures, and disease-specific flow profiles, we need more predefined workspaces for users to interact with. Adapting these workflows requires learning Voreen, which he considered more challenging than SimVascular and ParaView. In addition, he perceived SimVascular and ParaView as offering a broader range of usage.

7. Conclusions and Future Work

We presented a novel open-source software system for generating and analyzing ensembles of flow simulations. We demonstrated its applicability through representative use cases that cover common simulation and analysis tasks. The system provides a GUI for image preprocessing, simulation configuration, and simulation analysis, enabling users to interactively set up, refine, and visualize measured data, geometric volume files, and simulation results. By integrating the complete workflow into a single solution, our system offers a comprehensive tool particularly suited to researchers without coding experience and to newcomers in the field.

Although the presented use cases were designed to highlight features that we found valuable in collaboration with domain experts, they are not limited to these exact applications. Each domain and project brings its own requirements, and additional steps can be incorporated by composing functionality from Voreen's extensive feature set, even without programming knowledge. Overall, our system supports researchers in medicine, biology, and numerical modeling who are interested in exploratory analysis of medical and volumetric flow data.

For future work, we plan to add more flexible boundary-condition placement, allowing users to configure x, y, and z positions along the mesh and to shift boundary elements via drag-and-drop (Section 6.1). Feedback from Session S2 (Section 6.2) further suggested MRI-specific extensions, such as deformable vessel walls (e.g., as supported in SimVascular), as well as incorporating patient data into predefined workflows to strengthen clinical relevance. Finally, MRI is not the only imaging modality used to construct geometric models of cardiovascular systems. Alternatives include computed tomography angiography (CTA) and 3D digital subtraction angiography (3D-DSA). While no such data are currently available, we plan to extend the system with additional use cases based on CTA and 3D-DSA.

Supplementary Materials: For each use case in Section 5, there are example videos uploaded on YouTube: Use-Case A https://www.youtube.com/watch?v=Prorm9gwx4E&t=35s, Use-Case B https://www.youtube.com/watch?v=UaVKYyxxqb8 and Use-Case C https://www.youtube.com/watch?v=b9jiQCNSxWg (accessed on 5 April 2026).

https://doi.org/10.3390/computers15050300

Author Contributions: Conceptualization, T.M. and S.L.; methodology, S.L.; software, S.L.; validation, T.M., S.L., A.K. and A.N.; formal analysis, T.M.; resources, A.N., K.G., M.F. and V.H.; writing—original draft preparation, T.M.; writing—review and editing, T.M., S.L., A.K., A.N., K.G., M.F., V.H., M.J.K. and L.L.; visualization, T.M.; supervision, V.H., M.J.K. and L.L.; project administration, V.H., M.J.K. and L.L.; funding acquisition, V.H., M.J.K. and L.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Deutsche Forschungsgemeinschaft (DFG) under project number 468824876.

Institutional Review Board Statement: All the experiments were conducted in accordance with the National Institute of Health Guidelines for the Care and Use of Laboratory Animals (8th edition) and the European Community Council Directive for the Care and Use of Laboratory Animals of 22 September 2010 (2010/63/EU). The study protocol was approved by the competent State Office of Food Safety and Consumer Protection (TLLV, Bad Langensalza, Germany; local registration number: UKJ-17-003).

Data Availability Statement: The data presented in this study are available in our GitHub repository and source code.

Acknowledgments: During the preparation of this work the authors used ChatGPT-4o and a generative AI hosted by the University of Münster in order to improve the readability and language of the manuscript. After using these tools, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

MRI	Magnetic Resonance Imaging
CFD	Computational Fluid Dynamics
LBM	Lattice Boltzmann Method
LBE	Lattice Boltzmann Equation
STL	Stereo Lithography
GUI	Graphical User Interface
NSE	Navier–Stokes equations
CH	Computational Hemodynamics
BGK	Bhatnagar–Gross–Krook
HPC	High-Performance Computing
MDS	Multidimensional Scaling
CTA	Computed Tomography Angiography
3D-DSA	3D Digital Subtraction Angiography

References

1. Krause, M.J.; Kummerländer, A.; Avis, S.J.; Kusumaatmaja, H.; Dapelo, D.; Klemens, F.; Gaedtke, M.; Hafen, N.; Mink, A.; Trunk, R.; et al. OpenLB—Open source lattice Boltzmann code. *Comput. Math. Appl.* **2021**, *81*, 258–288. [CrossRef]
2. Drees, D.; Leistikow, S.; Jiang, X.; Linsen, L. Voreen–An Open-source Framework for Interactive Visualization and Processing of Large Volume Data. *arXiv* **2022**, arXiv:2207.12746. [CrossRef]
3. Kodman, J.; Singh, B.; Murugaiah, M. A Comprehensive Survey of Open-Source Tools for Computational Fluid Dynamics Analyses. *J. Adv. Res. Fluid Mech. Therm. Sci.* **2024**, *119*, 123–148. [CrossRef]
4. Winter, M. Benchmark and Validation of Open Source CFD Codes, with Focus on Compressible and Rotating Capabilities, for Integration on the SimScale Platform. Master's Thesis, Chalmers University of Technology, Gothenburg, Sweden, 2014.
5. Updegrove, A.; Wilson, N.M.; Merkow, J.; Lan, H.; Marsden, A.L.; Shadden, S.C. SimVascular: An open source pipeline for cardiovascular simulation. *Ann. Biomed. Eng.* **2017**, *45*, 525–541. [CrossRef] [PubMed]

6. SimVascular. 2023. Available online: https://simvascular.github.io/ (accessed on 15 April 2026).
7. Arthurs, C.J.; Khlebnikov, R.; Melville, A.; Marčan, M.; Gomez, A.; Dillon-Murphy, D.; Cuomo, F.; Silva Vieira, M.; Schollenberger, J.; Lynch, S.R.; et al. CRIMSON: An open-source software framework for cardiovascular integrated modelling and simulation. *PLoS Comput. Biol.* **2021**, *17*, e1008881. [CrossRef] [PubMed]
8. Crimson: An Advanced Simulation Environment for Subject-Specific Hemodynamic Analysis. 2024. Available online: https://crimson.software/index.html (accessed on 15 April 2026).
9. SimScale—Cloud-based Simulation Platform. Available online: https://www.simscale.com/ (accessed on 15 April 2026).
10. SimScale Documentation. Available online: https://www.simscale.com/docs/ (accessed on 15 April 2026).
11. Ansys. 2025. Available online: https://www.ansys.com/ (accessed on 15 April 2026).
12. Groen, D.; Arabnejad, H.; Suleimenova, D.; Edeling, W.; Raffin, E.; Xue, Y.; Bronik, K.; Monnier, N.; Coveney, P.V. FabSim3: An automation toolkit for verified simulations using high performance computing. *Comput. Phys. Commun.* **2023**, *283*, 108596. [CrossRef]
13. Groen, D. FabSim3: An Automation Toolkit for Complex Simulation Tasks. 2023. Available online: https://fabsim3.readthedocs.io/en/latest/ (accessed on 15 April 2026).
14. OpenFOAM. 2025. Available online: https://www.openfoam.com/ (accessed on 15 April 2026).
15. Chen, G.; Xiong, Q.; Morris, P.J.; Paterson, E.G.; Sergeev, A.; Wang, Y. OpenFOAM for computational fluid dynamics. *Not. AMS* **2014**, *61*, 354–363. [CrossRef]
16. Jasak, H. OpenFOAM: Open source CFD in research and industry. *Int. J. Nav. Archit. Ocean Eng.* **2009**, *1*, 89–94. [CrossRef]
17. Voreen Project. Voreen. 2025. Available online: https://github.com/voreen-project/voreen (accessed on 15 April 2026).
18. Wilson, A.T.; Potter, K.C. Toward visual analysis of ensemble data sets. In *Proceedings of the 2009 Workshop on Ultrascale Visualization, UltraVis '09*; ACM : New York, NY, USA, 2009; pp. 48–53. [CrossRef]
19. Evers, M.; Linsen, L. Multi-dimensional parameter-space partitioning of spatio-temporal simulation ensembles. *Comput. Graph.* **2022**, *104*, 140–151. [CrossRef]
20. Hummel, M.; Obermaier, H.; Garth, C.; Joy, K.I. Comparative visual analysis of Lagrangian transport in CFD ensembles. *IEEE Trans. Vis. Comput. Graph.* **2013**, *19*, 2743–2752. [CrossRef] [PubMed]
21. Mirzaee, H.; Henn, T.; Krause, M.J.; Goubergrits, L.; Schumann, C.; Neugebauer, M.; Kuehne, T.; Preusser, T.; Hennemuth, A. MRI-based computational hemodynamics in patients with aortic coarctation using the lattice Boltzmann methods: Clinical validation study. *J. Magn. Reson. Imaging* **2017**, *45*, 139–146. [CrossRef] [PubMed]
22. Qureshi, M.U.; Colebank, M.J.; Paun, L.M.; Ellwein Fix, L.; Chesler, N.; Haider, M.A.; Hill, N.A.; Husmeier, D.; Olufsen, M.S. Hemodynamic assessment of pulmonary hypertension in mice: A model-based analysis of the disease mechanism. *Biomech. Model. Mechanobiol.* **2019**, *18*, 219–243. [CrossRef] [PubMed]
23. Krüger, T.; Kusumaatmaja, H.; Kuzmin, A.; Shardt, O.; Silva, G.; Viggen, E.M. *The Lattice Boltzmann Method: Principles and Practice*; Graduate Texts in Physics; Springer International Publishing: Cham, Switzerland, 2017. [CrossRef]
24. Simonis, S. Lattice Boltzmann Methods for Partial Differential Equations. Doctoral Thesis, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany, 2023. [CrossRef]
25. Schroeder, W.; Martin, K.; Lorensen, B. *The Visualization Toolkit*, 4th ed.; Kitware: Clifton Park, NY, USA, 2006.
26. Kummerländer, A.; Bingert, T.; Bukreev, F.; Czelusniak, L.E.; Dapelo, D.; Gaul, C.; Hafen, N.; Ito, S.; Jeßberger, J.; Khazaeipoul, D.; et al. *Open Source Lattice Boltzmann Code*; OpenLB Release 1.8.1.; Mathias J. Krause: Karlsruhe, Germany, 2025. [CrossRef]
27. Kummerländer, A.; Dorn, M.; Frank, M.; Krause, M.J. Implicit propagation of directly addressed grids in lattice Boltzmann methods. *Concurr. Comput. Pract. Exp.* **2023**, *35*, e7509. [CrossRef]
28. Kummerländer, A.; Bukreev, F.; Berg, S.F.R.; Dorn, M.; Krause, M.J. Advances in Computational Process Engineering using Lattice Boltzmann Methods on High Performance Computers. In *High Performance Computing in Science and Engineering '22*; Nagel, W.E., Kröner, D.H., Resch, M.M., Eds.; Springer Nature: Cham, Switzerland, 2024; pp. 233–247. [CrossRef]
29. Kummerländer, A.; Bukreev, F.; Teutscher, D.; Dorn, M.; Krause, M.J. Optimization of Single Node Load Balancing for Lattice Boltzmann Methods on Heterogeneous High Performance Computers. *J. Parallel Distrib. Comput.* **2025**, *206*, 105169. [CrossRef]
30. Haussmann, M.; Ries, F.; Jeppener-Haltenhoff, J.B.; Li, Y.; Schmidt, M.; Welch, C.; Illmann, L.; Böhm, B.; Nirschl, H.; Krause, M.J.; et al. Evaluation of a Near-Wall-Modeled Large Eddy Lattice Boltzmann Method for the Analysis of Complex Flows Relevant to IC Engines. *Computation* **2020**, *8*, 43. [CrossRef]
31. Leistikow, S.; Nahardani, A.; Hoerr, V.; Linsen, L. Interactive Visual Similarity Analysis of Measured and Simulated Multi-field Tubular Flow Ensembles. In *Eurographics Workshop on Visual Computing for Biology and Medicine (2020)*; Eurographics Association: Lower Saxony, Germany, 2020; pp. 139–150. [CrossRef]
32. Szilvśi-Nagy, M.; Matyasi, G. Analysis of STL files. *Math. Comput. Model.* **2003**, *38*, 945–960. [CrossRef]

33. Drees, D.; Scherzinger, A.; Hägerling, R.; Kiefer, F.; Jiang, X. Scalable robust graph and feature extraction for arbitrary vessel networks in large volumetric datasets. *BMC Bioinform.* **2021**, *22*, 346. [CrossRef] [PubMed]
34. Grady, L. Random walks for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2006**, *28*, 1768–1783. [CrossRef] [PubMed]