WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

BioX: A Biological Cell Simulation Sandbox for Mechanistic Modeling of Multicellular Systems

G N Sathvik[1], Dheraj M[2], T Tamakesh[3], B Pavan Kumar[4], Dr. Manoj Kumar M[5]

[1] Dept. of Computer Science and Engineering Jyothy Institute of Technology Bangalore, India
sathvikgn@gmail.com
[4] Dept. of Computer Science and Engineering Jyothy Institute of Technology Bangalore, India
pavankumar202004@gmail.com
[2] Dept. of Computer Science and Engineering Jyothy Institute of Technology Bangalore, India
mdheraj0403@gmail.com
[5] Dept. of Computer Science and Engineering Jyothy Institute of Technology Bangalore, India
manoj.kumar@jyothyit.ac.in
[3] Dept. of Computer Science and Engineering Jyothy Institute of Technology Bangalore, India
ttamakesh@gmail.com

ABSTRACT—

The behavior of biological tissues is governed by local cell-to-cell interactions. Predicting this behavior is a central goal of systems biology. We built BioX (Biological Cell Simulation Sandbox) to address this; it is an agent-based modeling platform for studying tissue-scale biology. BioX uses the PhysiCell C++ engine for its core simulation and the BioFVM solver for substrate diffusion. The platform's main feature is the Biox-Studio, a graphical user interface (GUI) that allows researchers to set up and run complex 3D simulations without writing C++ code. To show a typical workflow, we simulate the 3D growth of a tumor spheroid and its response to a diffusing drug. Ultimately, BioX is a publicly available tool designed to make mechanistic modeling more accessible for both research and education.

Introduction

Individual cells form biological tissues, but the complex functions of those tissues—from organ development to immune response—arise from collective action. It is really hard to understand how one cell doing its job suddenly becomes a whole group of cells working together. We use computer programs to recreate these situations so we can see how it works.
Computer simulations give us a solid way to move forward. Within this field, agent-based models (ABMs) have proven particularly useful; they can track thousands of individual agents to see how local interactions produce large-scale patterns. However, high-fidelity 3D simulation typically requires advanced C++ programming and High-Performance Computing (HPC) expertise, a skill set that creates a significant barrier to entry for experimental biologists and clinicians.

We developed BioX specifically to bridge this gap. BioX is a tiered simulation platform that encapsulates a high-performance C++ computational backend within a multi-modal graphical interface. Unlike traditional tools that offer a monolithic interface, BioX introduces a central "Launcher Hub" that categorizes workflows into three distinct levels of complexity: a Simple Mode for educational templates, a Complex Mode (BioX-Studio) for granular research control, and a Drug Tester Mode integrated with external pharmacological databases (ChEMBL and DrugBank). This paper details the platform's modular architecture and demonstrates its utility in streamlining the workflow from pharmacological data retrieval to mechanistic simulation.

Literature Survey

Computational cell modeling has diversified rapidly, evolving from rule-based simulators to hybrid AI–mechanistic frameworks operating across molecular, cellular, and multicellular scales. For instance, Bunne et al. [1] proposed the AI Virtual Cell (AIVC), a multimodal neural-network framework trained on omics and imaging datasets, concluding that such models can overcome the limitations of traditional approaches by enabling predictive digital experiments and accelerating biological discovery. In a complementary direction, Francis et al. [2] introduced the SMART toolkit, which integrates finite-element solvers with realistic cell geometries; Their study showed that you really need to simulate exactly *where* and *when* things happen in a cell. If you just assume everything is mixed together evenly (like a soup), you miss the details—like specific little pockets of Calcium or how cell parts connect.
In cancer research, experts found that these computer models are the only way to really figure out the best time to give treatment or predict when a drug might stop working, and incorporating tumor–microenvironment dynamics into treatment design. Extending this concept, Ham et al. [4] argued that

hybrid mechanistic–AI approaches are essential for predictive, testable insights into cell-fate decision making, thereby shifting experimental focus toward discriminatory data collection.

Drug discovery research has also benefited from computational methods: DiNuzzo [5] concluded that AI-enabled network modeling can accelerate hypothesis generation and improve target validation, ultimately reducing downstream clinical failures. At the molecular scale, Stevens et al. [6] demonstrated the feasibility of whole-cell molecular dynamics simulations of the JCVI-syn3A minimal cell, concluding that integrative atomistic modeling provides unprecedented insights into spatiotemporal cellular organization.

Building on these advances, our work introduces BioX, a simulation sandbox built on the open-source PhysiCell framework [7], supported by its BioFVM transport solver [9]. Unlike prior models that are either highly specialized or technically inaccessible, BioX integrates an agent-based 3D engine with an intuitive GUI, enabling large-scale multicellular simulations without requiring advanced programming expertise [8]. BioX takes powerful, pro-level tools and makes them actually usable. It gives researchers a reliable place to test out their theories on how cells work together, without the headache of complex software.

System Architecture and Methods

BioX is built on a modular design that combines two distinct solvers: one for tracking individual agents (cells) and another for the environment around them. The system is split into two main parts: the **Computational Core** (the engine) and the **Interface Layer** (what the user sees).

A. The BioX Computational Core

The simulation engine uses a hybrid approach:

- Agent-Based Mechanics: We treat every single cell as its own independent agent. Each one tracks its own size, life stage, and death. To keep things running fast, the engine uses a 3D grid system to calculate physical forces—like cells sticking together or pushing apart. This allows us to simulate massive populations of cells in 3D without slowing down.
- Transport Solver: The biochemical microenvironment is simulated using a finite volume method (FVM). This solver calculates the diffusion, decay, uptake, and secretion of chemical substrates (e.g., oxygen, drugs, signaling factors) by solving reaction-diffusion Partial Differential Equations (PDEs). This enables dynamic coupling where cell behavior is driven by local environmental gradients.

B. Microenvironment Simulation with BioFVM

The biochemical environment is simulated using the integrated BioFVM (Biological Finite Volume Method) engine [2]. BioFVM solves systems of reaction-diffusion partial differential equations (PDEs) to model the concentration of chemical substrates. This is critical for capturing spatial effects, which are often neglected in well-mixed models that rely on ordinary differential equations (ODEs) [3]. This setup creates a real-time link between the cells and their environment, allowing us to simulate two key things:

- **Chemical Dynamics:** How substances (like oxygen, glucose, or drugs) spread out, get absorbed by cells, get released by cells, or naturally break down over time.
- **Feedback Loops:** Cells can actually 'sense' what is around them. If the local environment changes, the cell can react and change its behavior.

C. Customizable Cell Phenotype and Cycle Models

Every cell has its own detailed profile, including its size, life stage, and health status. The platform comes with several pre-made biological models that users can tweak to fit their needs:

- **Cell Cycles:** You can choose simple models (is the cell active or resting?) or complex ones that track the full transition from living to dying to dead (necrosis)."
- Cell Death: The framework can model both apoptosis and necrosis, with transition rates that can be dependent on environmental factors.

D. Data Integration Layer

A standout feature of BioX is that it pulls in real-world data instantly. We keep local copies of major databases like ChEMBL and DrugBank right inside the system. This allows BioX to automatically look up specific drug details—like molecular weight or how potent a drug is—so your simulation is always based on real facts, not just guesses.

E. User Interface and Operational Modes

We designed BioX to work for beginners and experts alike. The 'Launcher Hub' acts as a menu where you can choose between three ways to work:

1. **Complex Mode (BioX-Studio):** Think of this as 'Expert Mode.' It gives you full access to everything the engine can do. It serves as a visual editor where advanced researchers can define domain geometry, create multiple cell types with distinct phenotypes, and program custom logic rules (e.g., "if oxygen drops below X, secrete factor Y"). It allows for full parameterization without direct C++ coding.
2. **Simple Mode:** Designed for education and rapid prototyping, this mode abstracts the underlying complexity. Users select from a library of pre-configured templates (e.g., "Tumor Growth," "Immune Defense"). The system automatically handles the engine configuration and launches an embedded visualization dashboard, allowing users to observe emergent behaviors immediately.
3. **Drug Tester Mode:** This mode is a specialized interface for in silico pharmacology. It features:
    - **Compound Search:** A direct interface to search the integrated drug databases.
    - **Auto-Fill Data:** You don't need to enter numbers manually. The system grabs the real data (like drug sensitivity) and plugs it straight

into your simulation.

- **Scenario Switching:** You can easily toggle between Cancer Mode (watching a drug soak into a tumor) and Virus Mode (tracking an infection and the immune response). The interface automatically changes to show the right tools for the job.

Application: Putting it to the Test: A Virtual Experiment

To demonstrate the platform's integrative workflow, we utilized the Drug Tester Mode to simulate the effects of a chemotherapeutic agent on a cancerous tumor cell cluster.

A. The Setup Using the Drug Tester interface, we chose the **Cancer Therapy** mode. Instead of doing the complex math ourselves to estimate how the drug moves, we simply queried the system for "Paclitaxel." BioX instantly pulled the key data—specifically how potent the drug is (IC50) and its target (Tubulin)—directly from the ChEMBL database. The system then automatically updated the simulation settings, adjusting the physics of how the drug spreads based on its actual molecular weight.

B. How it Played Out We set up a small 3D environment containing a cluster of cancer cells in the center, surrounded by a ring of healthy tissue. We then placed a virtual "Drug Capsule" next to the tumor. Once the simulation started, the capsule released the Paclitaxel, and the system calculated exactly how the drug diffused through the tissue over time, creating a realistic concentration gradient.

C. What We Saw We watched the simulation unfold in real-time using the built-in Studio Viewer. As the drug levels rose high enough to be effective, the cancer cells (shown in red) began to die and turn black. Crucially, the healthy cells (shown in blue)—which are naturally more resistant—survived. This experiment confirms that BioX can successfully take raw data from a database and turn it into an accurate, visual simulation for testing hypotheses.

Key parameters for the representative simulation.

Parameter	Value	Units
Domain Size	2000×2000×20	Mm
Simulation Duration	64,800	minutes
Oxygen Diffusion Coeff.	100,000	µm2/min
Oxygen Decay Rate	0.1	1/min
Oxygen Boundary Cond.	38	mmHg
Initial Cell Count	100	Cells
Cell Proliferation Rate	0.00072	1/min

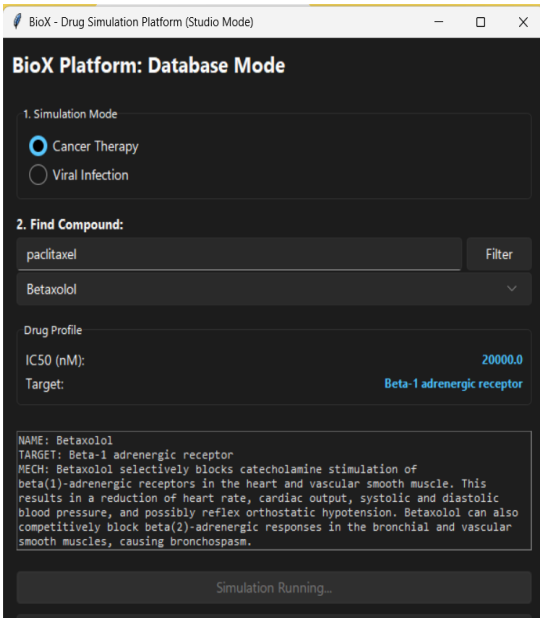

Fig. 1 The Biox Drug Tester mode demonstrates paclitaxel drug simualation on a cancerous cell

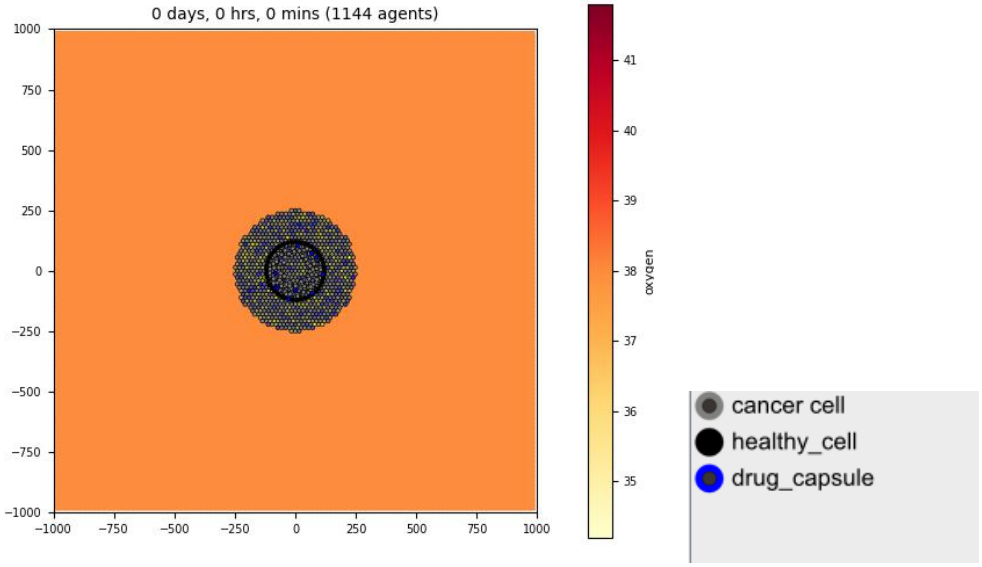

Fig. 2 Demonstrating the simulation with drug as a substrate 0 hrs initial plot

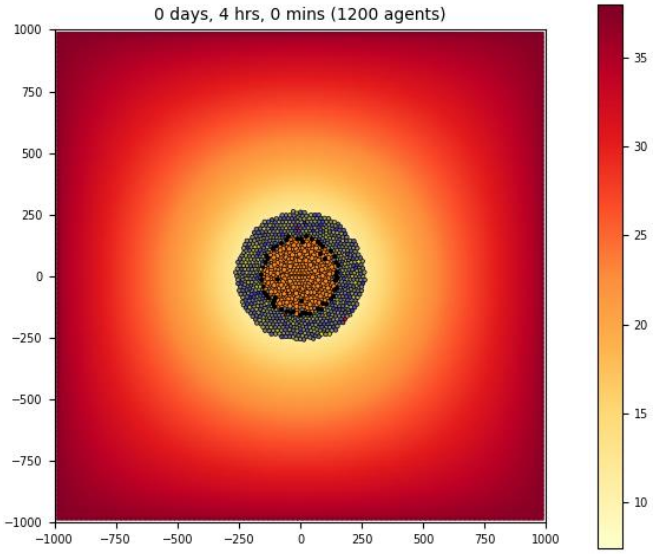

Fig. 3

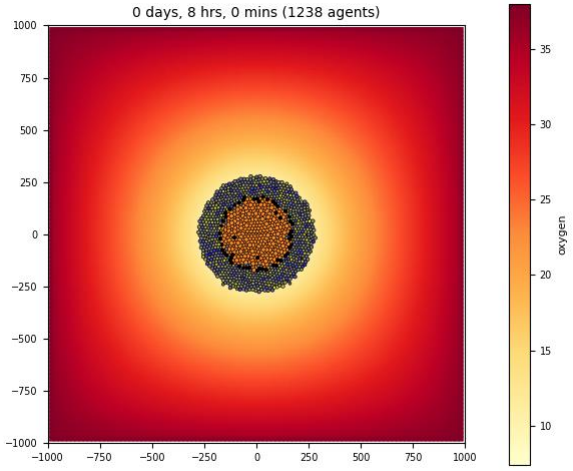

Fig. 4

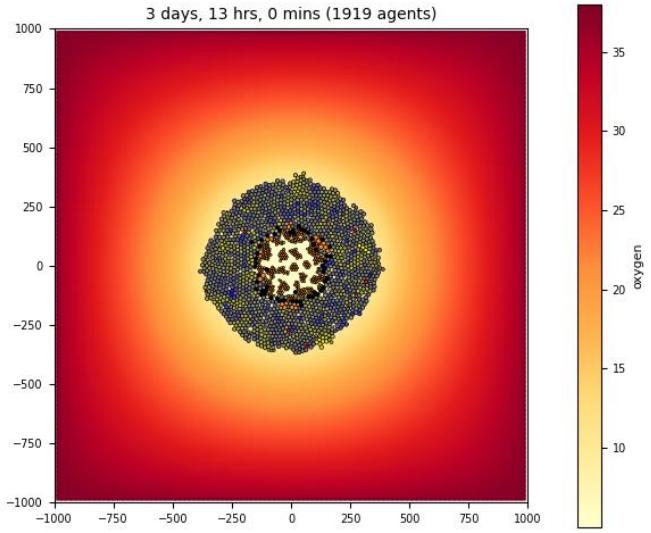

Fig. 5 Observing Most cancerous cell are dead

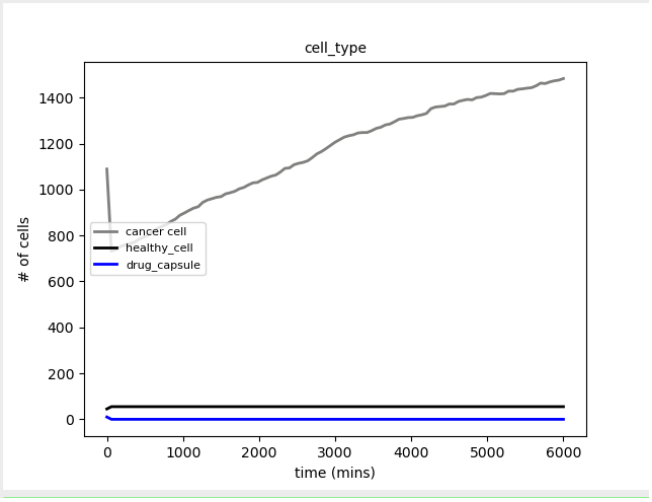

Fig. 6 The Population plot spanning from 0 hr to 3 days 13 hrs

Fig. 2/3/4/5. Time-series progression of a simulated multicellular spheroid. The images show the state of the cell population at (2) 0 hours, (3) 4 hours, and (4) 8 hours.(5)3 days 12 hours The total agent count increases from 1144 to 1919 over this period, demonstrating initial proliferation (Blue=proliferating, Yellow=quiescent, Red=necrotic) of a cancer cell while Black represents healthy cell the blue drug capsule enfuses with the cancer cell secreting drug substrate as seen in opservation at fig 3.The substrate drug kills the cancer cell over the span of time as observed.

Discussion

The central challenge we aimed to address with BioX is one of accessibility. High-fidelity tools in computational biology often remain underutilized in experimental settings because they require specialized C++ programming skills and High-Performance Computing (HPC) knowledge. Our primary contribution is the development of a platform that lowers this barrier by encapsulating a high-performance C++ backend within a tiered, intuitive graphical user interface.

The BioX Approach: Flexibility and Usability Many existing computational tools force every user into the same rigid workflow, regardless of their goal. BioX takes a different approach by acknowledging that a student visualizing cell division needs a different interface than a pharmacologist screening compounds. By separating the platform into three distinct modes—Simple for education, Complex for granular control, and Drug Tester for specific experiments—we have created a system that supports both initial exploration and rigorous hypothesis testing.

Validation of the Workflow The tumor experiment described earlier served as the primary proof-of-concept for this design. It demonstrated that we can replicate complex biological scenarios—specifically, the way a drug diffuses through tissue and causes tumor death—without requiring the user to write code or manually calculate physics. By successfully driving the simulation using data pulled directly from ChEMBL and DrugBank, we proved that it is possible to run a complete, data-backed experiment entirely through the graphical interface.

Future Directions: merging AI with Simulation Moving forward, the logical evolution of BioX is to reduce the need for manual setup by integrating artificial intelligence. We aim to build workflows where machine learning algorithms analyze real-world microscopy data and automatically tune the

simulation parameters to match. This transition would transform BioX from a theoretical research platform into a predictive tool capable of modeling specific patient scenarios—a critical step toward creating true "digital twins" for personalized cancer treatment.

Conclusion

BioX was built with a specific goal: to take powerful, complex modeling tools and make them accessible to people who aren't software engineers. By combining a high-speed C++ engine with a user-friendly Python interface, we allow researchers to simulate difficult biological concepts—like how cell populations grow or how drugs move through tissue—without needing to write a single line of code.

What makes the platform truly practical is its connection to the real world. Because it can pull actual chemical data from databases and visualize the simulation in real-time, BioX serves as a dual-purpose tool: it is simple enough for teaching students, but accurate enough for scientists to test new theories on a computer before heading to the lab. Ultimately, it offers a solid foundation for understanding the messy, complex reality of biological systems.

Acknowledgment

We want to thank the Department of Computer Science and Engineering at Jyothy Institute of Technology, Bangalore. They provided the hardware and the support environment that made this research possible.

We also owe a debt to the open-source community—specifically the developers of the PhysiCell framework and the teams maintaining the ChEMBL and DrugBank databases. Their hard work provided the essential building blocks for the BioX platform.

REFERENCES

[1] C. Bunne, Y. Roohani, Y. Rosen, et al., “How to build the virtual cell with artificial intelligence: Priorities and opportunities,” Cell, vol. 187, no. 12, pp. 7045–7056, Dec. 2024.

[2] E. A. Francis, J. G. Laughlin, J. S. Dokken, et al., “Spatial modeling algorithms for reactions and transport in biological cells,” Nature Computational Science, vol. 5, no. 1, pp. 76–89, November. 2024.

[3] C. Ma and E. Gurkan-Cavusoglu, “A comprehensive review of computational cell cycle models in guiding cancer treatment strategies,” npj Systems Biology and Applications, vol. 10, no. 71, pp. 1–15, 2024.

[4] L. Ham, T. E. Woodward, M. A. Coomer, and M. P. H. Stumpf, “Mapping, modeling, and reprogramming cell-fate decision making systems,” arXiv preprint arXiv:2412.00667, Dec. 2024.

[5] M. DiNuzzo, “How artificial intelligence enables modeling and simulation of biological networks to accelerate drug discovery,” Frontiers in Drug Discovery, vol. 2, pp. 1–9, Oct. 2022.

[6] J. A. Stevens, F. Grünewald, P. A. M. van Tilburg, et al., “Molecular dynamics simulation of an entire cell,” Frontiers in Chemistry, vol. 11, no. 1106495, pp. 1–12, Jan. 2023.

[7] A. Ghaffarizadeh, R. Heiland, S. H. Friedman, S. M. Mumenthaler, and P. Macklin, “PhysiCell: An open source physics-based cell simulator for 3-D multicellular systems,” PLOS Computational Biology, vol. 14, no. 2, pp. 1–31, Feb. 2018.

[8] G. N. Sathvik, D. M., Tamakesh, and Pavan, “BioX: A Biological Cell Simulation Sandbox for Mechanistic Modeling of Multicellular Systems,” Unpublished manuscript (current sdraft), 2025.

[9] P. Macklin, “BioFVM: An efficient, parallelized diffusive transport solver for 3-D biological simulations,” Bioinformatics, vol. 30, no. 22, pp. 3356–3358, Nov. 2014.