# Adaptive Physics Refinement for Anatomic Adhesive Dynamics Simulations

Aristotle Martin[1][0000−0002−8704−764X], William Ladd[1][0009−0002−1989−8757], Runxin Wu[1][0009−0008−1945−413X], and Amanda Randles[1][0000−0001−6318−3885]

Dept of Biomedical Engineering, Duke University, Durham NC 27705, USA
{aristole.martin, william.ladd, wendy.wu, amanda.randles}@duke.edu

**Abstract.** Explicitly simulating the transport of circulating tumor cells (CTCs) across anatomical scales with submicron precision—necessary for capturing ligand-receptor interactions between CTCs and endothelial walls—remains infeasible even on modern supercomputers. In this work, we extend the hybrid CPU-GPU adaptive physics refinement (APR) method to couple a moving finely resolved region capturing adhesive dynamics between a cancer cell and nearby endothelium to a bulk fluid domain. We present algorithmic advancements that: enable the window to traverse vessel walls, resolve adhesive interactions within the moving window, and accelerate adhesive computations with GPUs. We provide an in-depth analysis of key implementation challenges, including trade-offs in data movement, memory footprint, and algorithmic complexity. Leveraging the advanced APR techniques introduced in this work, we simulate adhesive cancer cell transport within a large microfluidic device at a fraction of the computational cost of fully explicit models. This result highlights our method's ability to significantly expand the accessible problem sizes for adhesive transport simulations, enabling more complex and computationally demanding studies.

**Keywords:** Adaptive physics refinement · Adhesive dynamics · Fluid-structure interaction · Multiscale modeling · Heterogeneous computing.

## 1 Introduction

Understanding the mechanisms driving cancer cell transport through the bloodstream requires models that can capture cellular behavior at the adhesion level while spanning long anatomical length scales. Circulating tumor cells (CTCs) interact with fluid forces and vascular walls through adhesive interactions, processes that are central to cancer metastasis, but remain poorly understood due to their complexity and multiscale nature. Computational modeling offers a unique opportunity to explore these dynamics by integrating cellular adhesion mechanisms with fluid transport across physiologically relevant scales. However, existing *in silico* models are limited in scope and are constrained by the computational cost of resolving submicrometer adhesive binding interactions over meter-length CTC trajectories. Current approaches predominantly focus on idealized microvessels [2, 3, 9, 16, 19], providing valuable but incomplete insights into

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

the localized relationships between fluid flow, cell mechanics, and adhesion. A critical need remains for a unified computational framework of cancer transport capable of maintaining high fidelity at the adhesion scale while efficiently simulating the transport of CTCs across anatomic scales. Addressing this challenge could help uncover new insights into the biophysical mechanisms of cancer progression and inform therapeutic strategies targeting CTC adhesion and vascular interactions. This study (Fig. 1) represents a critical step in this direction by introducing an adaptive physics refinement-based adhesive dynamics (APR-AD) model. The key contributions of this work include:

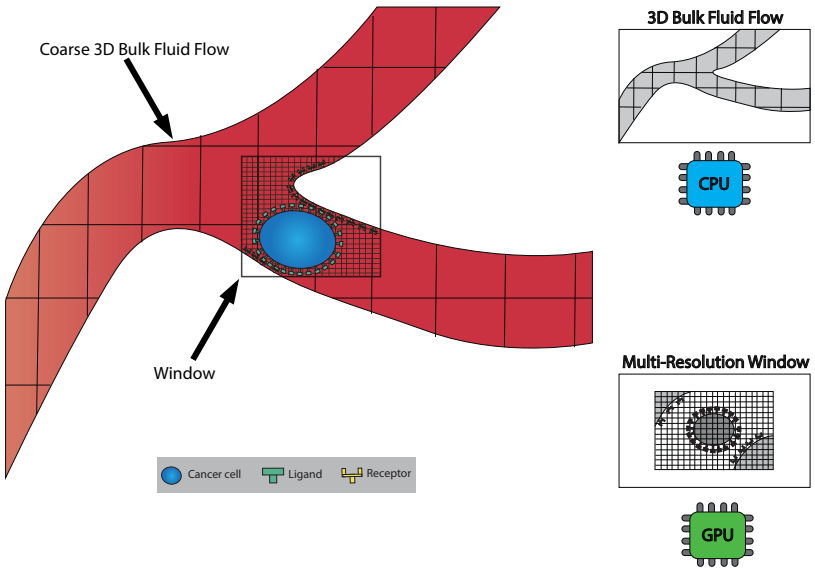


**Fig. 1.** Adaptive physics refinement-based adhesive dynamics (APR-AD) model overview depicting a cancer cell (blue) coated in ligands (green) undergoing adhesive interactions with wall receptors (yellow) within a finely resolved moving window that is coupled to a coarse bulk fluid-only domain. The bulk fluid simulation is performed on the CPUs, whereas the cellular calculations are done by GPUs.

1. **Algorithmic advancements** that extend APR to span vessel walls.
2. **Integration of a detailed adhesive dynamics model**, enabling high-fidelity simulation of ligand-receptor binding events.
3. **Optimization for GPU architectures,** ensuring computational efficiency for simulations across large-scale domains.

By enabling high-resolution simulations of adhesive transport at anatomic scales, the APR-AD framework represents a significant step forward in multiscale modeling. This approach not only bridges existing gaps in computational cancer

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

research, but also provides a scalable platform for addressing critical questions about the interplay between fluid mechanics, adhesion, and cancer cell transport that can probe fundamental drives of metastasis across physiologically relevant scales.

## 2 Application Overview

The present work employs HARVEY [13, 12], a massively parallel, multiphysics code. The fluid flow is resolved by solving the Lattice Boltzmann Bhatnagar-Gross-Krook (LBGK) equation on a standard D3Q19 lattice [11]. Fluid-structure interactions between finite element (FEM) meshed cells and the background fluid are computed using the Immersed Boundary method (IBM) [7, 1]. Adhesive interactions between cell surface ligands and receptor-lined endothelial walls are resolved using the stochastic adhesive dynamics (AD) model from [5, 4].

An adaptive physics refinement (APR) algorithm developed in [10, 14] is used by HARVEY to resolve tumor cell transport over large spatial distances. Other multiphysics approaches to modeling cancer transport processes include nanoparticle-based simulations [15]. Within the context of APR, high-resolution grids are used in the window to resolve fluid-structure interactions of the tracked cancer cell, while a coarser grid is employed outside the window to resolve the background fluid dynamics. The information exchange at the bulk-window boundary is handled through a multi-block approach detailed in [10]. An important quantity in the APR scheme is the ratio between the coarse and fine grid spacings, or the multi-resolution ratio $n$. Due to the cubic dependence of the total number of lattice sites $N_s$ on grid spacing $\Delta x$ ($N_s \sim \frac{1}{\Delta x^3}$, for three spatial dimensions), significant memory savings can result from higher values of $n$ [6]. The APR method is optimized for heterogeneous workloads, with the bulk computations performed by the CPUs, and the fine window calculations handled by the GPUs (Fig. 1).

## 3 Algorithmic Advances to Capture Adhesive Interactions with APR

In this section, we describe the methodology for extending APR to include AD. Sections 3.1 and 3.2 detail the incorporation of walls and endothelial receptors, respectively, in APR. An overview of the window move algorithm is provided in Section 3.3. Finally, the development of a GPU-accelerated AD implementation is documented in section 3.4.

### 3.1 Incorporation of Walls into APR

A major advancement made in the present work is the additional ability of the moving window to traverse vessel walls. In previous versions of the APR algorithm, the window was assumed to be entirely submerged in fluid [10]. Allowing

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

the APR window to span vessel walls was a necessary prerequisite to accurately resolving adhesive phenomena within the window. Several changes were made to the existing framework to accommodate walls.

When the window is first created, the identities of the grid points (i.e., fluid or wall) must be determined from the surrounding bulk. Previously, all window interior points were assumed to be fluid. In the updated setup, bulk tasks classify grid points as either fluid or wall points and then communicate the subset of points that intersect with the window (Fig. 2(A)). Within the window, categorization of misaligned points into either fluid or wall type relies on interpolation from nearby bulk points (Fig. 2(B)). Specifically, if a fine-scale point is completely surrounded by coarse fluid points, it is classified as fluid; otherwise, if a single neighboring coarse point is a wall, the corresponding window point is designated as a wall.

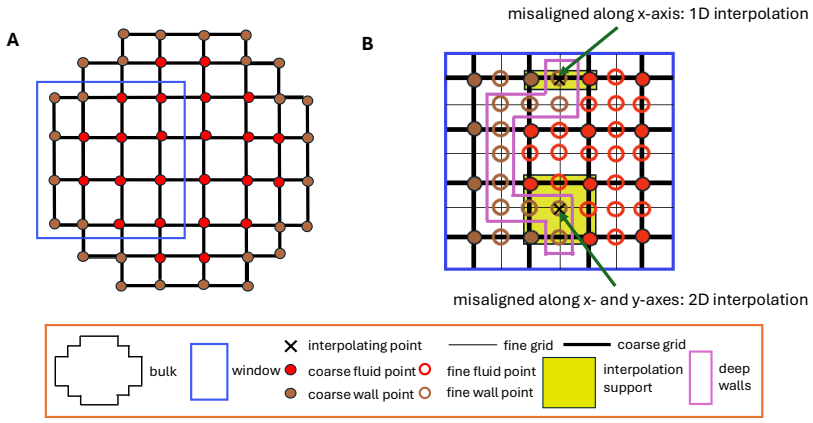


**Fig. 2.** Schematic illustration of the process for setting up the window points. (A): the bulk tasks communicate the subset of points intersecting with the window box. (B): the window tasks interpolate misaligned points from the bulk. For this example, the multi-resolution ratio $n = 2$. Two fine points are denoted with the "x" mark with their interpolation boxes shown in yellow to illustrate how their types are determined.

As a result, the fine-scale fluid point boundary remains aligned with the bulk fluid boundary, while additional "deep" wall points fill the gap between the last fine fluid point and neighboring coarse wall points. The thickness of this inner layer of wall points will depend on the resolution ratio $n$, but does not exceed one coarse lattice unit. These deep wall points are subsequently pruned from the vessel geometry to ensure consistency in the final representation.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

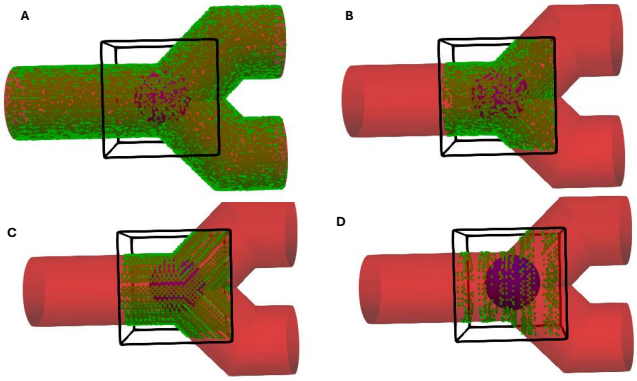


**Fig. 3.** Overview of different methods for placing wall receptors in the APR framework. (A) Method 1: bulk-driven placement, with random selection of receptors. (B) Method 2: window placement, with random selection of receptors. (C) Method 3: window placement with spatial patterning function $\Phi$ from Equation 1, and constant threshold $\Gamma$. (D) Method 3, with variable threshold $\Gamma$ from Equation 2. The black outline of the APR window is depicted in each case, with the cancer cell in blue, and wall receptors in green.

### 3.2 Wall Receptor Placement

The initialization of endothelial wall receptors within the APR window requires careful consideration. To address this, we developed three distinct methods (Fig. 3). The first method initializes wall receptors globally (Fig. 3(A)). Bulk tasks generate receptor distributions over the full geometry and communicate their positions to the window tasks. While this approach ensures precise mapping of wall receptor distributions (e.g., based on mesh coloring, as in [9]) to the window, it comes with increased memory demands and communication overhead due to a large number of receptors.

In contrast, the second method (Fig. 3(B)) localizes receptor initialization to the window itself. During window creation, receptors are spawned randomly by the window root rank and then broadcast to the rest of the window tasks. By delegating receptor selection to a single root process, this approach guarantees deterministic receptor placement irrespective of window process count. The downside of this approach is that receptor spawning is serialized, causing other window tasks to remain idle during initialization. Nevertheless, this approach significantly reduces memory usage, as only the wall receptors that are needed by the window tasks are stored at any given time. It is particularly well-suited for scenarios where a relatively uniform wall receptor distribution is assumed.

The third method balances the flexibility of the first method with the memory efficiency of the second. Here, window tasks independently generate receptors in parallel using a spatial patterning function, $\Phi$, which determines receptor placement based on the global position of the wall point. A receptor is placed if

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

the function output exceeds a threshold $\Gamma$. A spatial patterning function yielding a checkerboard pattern is shown in Equation (1).

$$\Phi(x, y, z) = \sin(2\pi f x)\sin(2\pi f y)\sin(2\pi f z) \tag{1}$$

The frequency $f$ in Equation (1) is set according to the desired wall receptor density. A simple stripe pattern (Fig. 3(D)) of wall receptors along the $z$ direction can be obtained through specifying $\Gamma$ as in Equation (2).

$$\Gamma(z) = \frac{\mathrm{erf}(k_1 \sin(z) + k_2) + 1}{2} \tag{2}$$

where erf is the error function, and $k_1$ and $k_2$ are constants selected to control the stripe spacing.

Ultimately, the choice of receptor placement method depends on the application. The first method offers maximal flexibility with arbitrary receptor distributions based on colored meshes. The second method (Fig. 3(B)) serves as an effective first pass approximation when uniform receptor distributions are sufficient. When a specific pattern is required that can be described mathematically, the third method (Fig. 3(C-D)) provides a flexible, function-driven approach.

### 3.3 Moving the Window

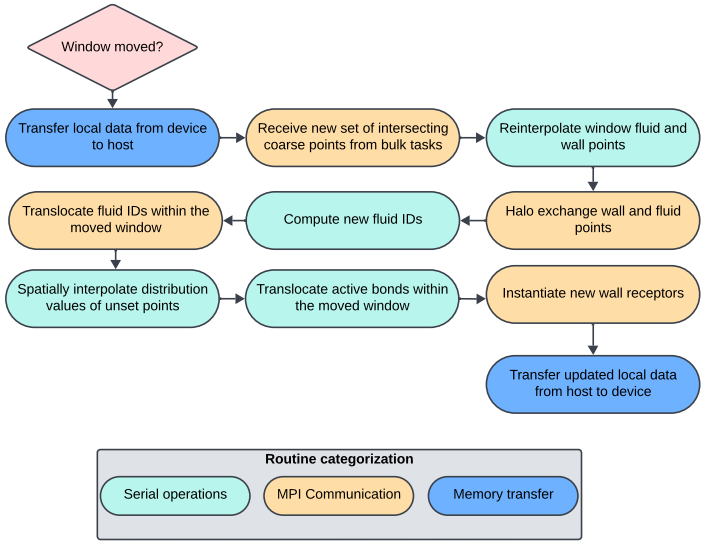


**Fig. 4.** Overview of the window move algorithm with incorporation of walls, from the window task perspective.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

When the tracked cell nears the edges of the APR region, a window move is triggered, initiating a cascade of operations outlined in Fig. 4. Some steps mirror those taken for updating a fully submerged window: data are transferred from the device to the host, which then re-orients fluid points within the window following the move and spatially interpolates distribution values for incoming fluid points near the leading edge of the window [10]. However, to accommodate vessel walls, several routines are significantly modified, and additional steps are introduced. When the window relocates, new fluid and wall point positions are determined through re-interpolation of intersecting bulk points (Fig. 2). This operation is followed by a communication exchange to update wall and fluid points at the window task boundaries. In the case of a fully submerged window, this step is unnecessary, as the coordinates of halo fluid points remain unchanged, preserving the communication structure. However, for a partially submerged window, refreshing the fluid communication tables is essential, as the number of fluid points can change due to the presence of passing wall points. Consequently, the indices of fluid points, referred to as "fluid IDs", must then be re-computed causing the new fluid IDs to become out of sync with the original data. To efficiently reconcile these discrepancies, a map data structure is used to translate fluid IDs and ensure accurate data movement.

After a window move, newly uncovered window points are updated through spatial interpolation from the surrounding bulk. In contrast to a stationary window, where interpolation occurs in two dimensions along interfacial planes, these new points require full 3D interpolation. For a fully submerged window, tricubic interpolation using the Catmull-Rom spline [17] is applied over a cubic support region spanning four coarse points in each spatial dimension, with the interpolating point positioned near the center of the box. However, volumetric interpolation becomes more complex in a partially submerged window, where the presence of walls disrupts uniform support regions. In this case, identifying the appropriate support points requires determining the largest rectangular subprism of coarse fluid points that fully encloses the interpolating point. The algorithm we devised for this task is illustrated in Fig. 5.

The procedure begins by constructing an auxiliary array that records the maximum prism height values based on the point classifications. The algorithm then identifies all possible subprisms that contain the interpolating point $P$. Candidate subprism formation starts at an anchor column, extending downward until the maximal height drops below the height at the anchor value. Once a base face is established, the prism is extruded along the depth direction as far as allowed by the height values. A final check ensures that the resulting subprism contains the point $P$, and its volume $V_R$ is compared against the running maximum value. The resulting subprism becomes the 3D support used for spatial interpolation.

Once the window moves to encompass new wall points, new wall receptors are instantiated in accordance with the desired receptor placement scheme (Fig. 3). The shuffling of wall receptor indices that the window move introduces requires the re-mapping of preexisting ligand-receptor bonds to their new index locations.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

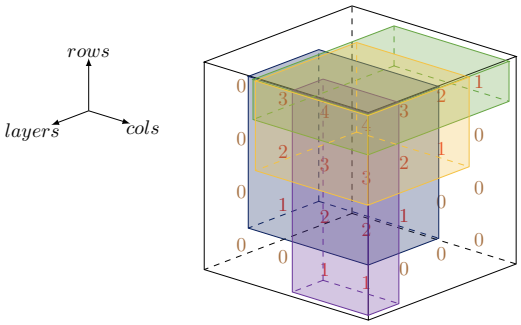


**Fig. 5.** Schematic illustration of subprism finder algorithm. The support matrix is shown, with nonzero values denoting the maximal height values of coarse fluid points, and zeros indicating the presence of coarse walls within the support. In this example, the algorithm is evaluating subprisms from the upper right position of the box. Four candidate rectangles are identified, with the interpolating location assumed to be near the center of the cube. The algorithm extrudes each of the rectangles identified as far as allowed based on the height values of deeper points to form a set of potential subprisms that contain the interpolating point.

### 3.4 GPU Acceleration of Adhesive Dynamics Calculations

The AD routines, originally designed to execute on multi-core CPUs, were redesigned as GPU kernels to be executed by the window tasks. Biological cells can express multiple species of ligand, that we collectively refer to as ligand sets. Each ligand set has a user-specified density that determines the number of ligands of that set placed on the cell surface. To model microvilli, ligands are permitted to interact with multiple wall receptors, though only one wall receptor can be engaged in a bond at any given time. The GPU parallelization strategy involves threading over ligands. To optimize memory access, data structures for adhesive cells, such as wall receptor positions, were reformatted from an array-of-structures layout to a structure-of-arrays format. At a high level, the AD process consists of three main components: (1) random number generation (RNG), (2) bond formation, and (3) bond rupture.

RNG is a prerequisite for both (2) and (3), as these routines rely on probabilistic interactions. We implemented RNG directly on the GPU using software-emulated linear feedback shift registers (LFSRs). An LFSR is a deterministic shift register that generates new values based on a linear function of its previous state. With an appropriate tap configuration, an LSFR of a given width can produce uniformly distributed values over a predefined period. For this work, we selected a 64-bit LFSR with the following feedback polynomial:

$$x^{63} + x^{61} + x^{60} + 1 \tag{3}$$

where the exponent terms represent the tapped bits. This tap configuration ensures a maximal cycle length, as documented in [18]. The LFSR was chosen for

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

its efficient representation on the GPU and low computational overhead. To integrate LFSRs into the AD process, we allocated an array of LFSRs to store random values for each ligand. Each ligand set has what we denote as its "root" LFSR, which is seeded based on a combination of parameters that uniquely identify the cell and particular ligand set for the cell. The binary sequences of subsequent LFSRs within the ligand set are derived by shifting the root a number of times corresponding to the relative position of the ligand within the set, guaranteeing that no two ligands draw identical values at any given time. Beyond its simplicity, this approach also minimizes communication overhead. When the CTC transfers ownership between window tasks, only the root LFSR of each ligand set needs to be communicated. The receiving task can then locally reconstruct the full set of LFSR states, maintaining determinism in the simulation. The LFSR kernel is written to leverage shared memory and gives an approximate 20% time reduction over an RNG block-based approach.

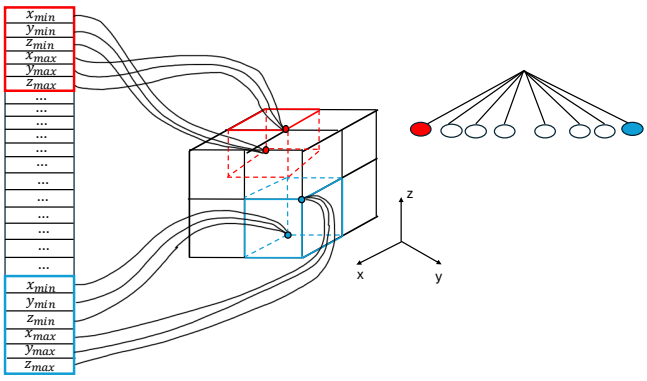


**Fig. 6.** Schematic representation of an octree in array form on the GPU. An octree with a single refinement level is shown for clarity. The octant array stores the minimum and maximum bounds of each octant box in an array-of-structures format.

Following the RNG phase, the bond formation step is executed. During each time step, adhesive ligands must identify the subset of wall receptors within range of bond formation. A naïve search through all wall receptors, for every cell ligand, leads to $O(N^2)$ time, making it computationally infeasible. To accelerate this process, we adopted an octree-based spatial partitioning scheme inspired by the CPU implementation [8] and optimized it for GPU execution (Fig. 6). For simplicity, a complete octree is used, ensuring a structured hierarchy. To guarantee a one-to-one mapping between cell ligands and octant indices, octant boxes are extended with a halo depth equal to the reactive distance, which is an intrinsic property of the ligand type. During bond formation, each ligand determines the index of its corresponding leaf node, which serves as a key to efficiently look up the start and end locations defining the segment of the wall

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

receptors that fall into the corresponding octant (the wall receptors are sorted by octant index during tree creation, further improving lookup efficiency).

One of the challenges of parallelizing the adhesive bond formation step on the GPU is the constraint that each wall receptor can only engage in a bond with one ligand at a time. To resolve thread contention over shared receptors, a simple arbitration mechanism is implemented: only the thread with the smallest ID is permitted to form a bond with a given wall receptor, ensuring deterministic and conflict-free assignments.

The final phase of AD involves evaluating whether bonds are broken. This operation is the simplest to parallelize and is completed in a single kernel invocation.

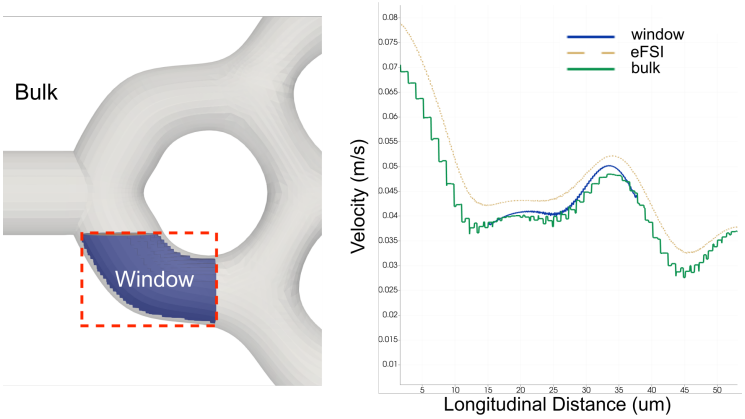


**Fig. 7.** APR walls validation in a stationary window. Left: region of the simulated microfluidic device, with the window location denoted by the red box and window fluid points indicated in dark blue. Right: Longitudinal velocity profiles for the bulk (green), window (blue), and eFSI (dotted yellow).

## 4 Results

### 4.1 Validation of APR Walls

To validate the multi-block scheme with the addition of walls, we simulated a stationary window placed within a complex vessel representing a microfluidic device (Fig. 7). The bulk fluid resolution was set to 0.6 $\mu$m, while the window resolution was refined to 0.15 $\mu$m ($n = 4$). As a reference, we used the flow profile from an explicit fluid structure interaction (eFSI) model with a consistent resolution across the entire domain that matched the finer resolution of the window domain. The velocity profile error between the window and eFSI cases remained within 5% with minor discrepancies attributed to interpolation errors and fluid convergence effects.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

### 4.2 Timing Composition Analysis

To evaluate the performance of the APR scheme, a timing composition analysis was conducted, as shown in Fig. 8. Conceptually, the simulation can be decomposed into two distinct phases, representing periods where the cell is advecting through a stationary window, and when the window itself is being moved to follow the cell. The timing diagram reflects this distinction by showing the total simulation loop time in Fig. 8(A), and the portion of the loop time spent in moving the window (denoted by "Window Move" in grey) is depicted in Fig. 8(B). Notably, the bulk spends a considerable amount of time in "Window Move." Fig. 8(B) indicates that the "Window Test" routine accounts for most of the window movement time. During the "Window Test", the root window rank checks whether the window needs to be moved, and broadcasts the Boolean result to the remaining window tasks, as well as the bulk. Altogether, the results suggest that the bulk tasks spend a significant portion of their time waiting on the window tasks. This result is expected since the bulk has less work assigned to it than the window (bulk fluid compared with IBM+AD), coupled together with the need for the window tasks to perform $n$ window time steps' worth of work for each corresponding bulk time step.

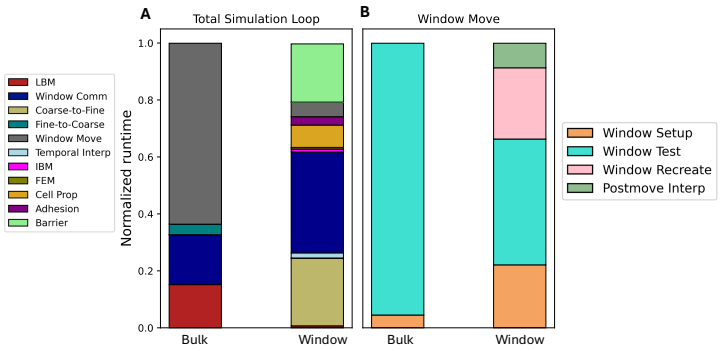


**Fig. 8.** Timing composition of APR simulation from Fig. 10. A: Total runtime breakdown for bulk (left) and window (right). B: Window move breakdown for bulk (left) and window (right). Subroutines with negligible runtime were excluded from the timing diagram.

The dominant runtime factor in the APR window is MPI communication (dark blue) (Fig. 8(A)). This result reflects a shift in the primary bottleneck from computation to communication, as core computational kernels (LBM, IBM, FEM) have been offloaded onto the GPU. Unlike the bulk, which only handles fluid data exchange, the window tasks must also manage cellular data communication, further increasing overhead. A significant portion of simulation time is spent in the "Coarse-to-Fine" routine, which involves receiving fluid data from bulk tasks at the multi-resolution interface and performing spatial interpolation.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

Since this routine remains CPU-bound, it is considerably more time-consuming than purely GPU-executed kernels such as IBM. The large "barrier" time indicates the load imbalance within the window, a consequence of the naive domain decomposition scheme used for window tasks. In contrast, the bulk benefits from a bisection load balancer, resulting in more even workload distribution. As shown in Fig. 8(B), window tasks spend most of their time performing the "Window Test" routine, where the root process determines whether a window move is required and broadcasts the result. This finding highlights the cost of global communication in the window task workflow. Outside of "Window Test", most of the window movement time is spent in "Window Recreate", which involves interpolating new wall points and reconstructing window data, and "Window Setup", which updates bulk-window communication tables.

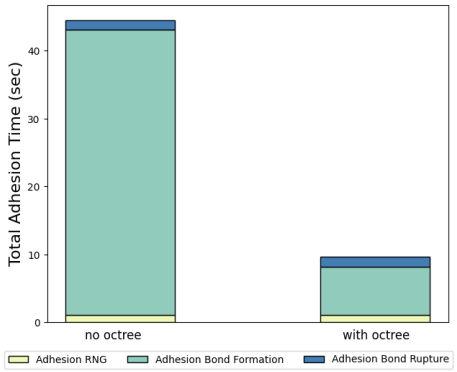


**Fig. 9.** Impact of GPU octree on adhesion time, during a 40,000 time step simulation in the microfluidic device (Fig. 10) on Aurora. Left: Total adhesion time without octree. Right: Total adhesion time with octree enabled.

### 4.3 Performance Optimizations

To better leverage the parallelism offered by the CPUs, a subset of bulk task functions were re-written using SIMD-style programming (i.e., SYCL ND-range kernels) analogous to the GPU. SIMD efficiency was enhanced by converting bulk data structures to structure-of-arrays (SoA). These code changes resulted in an approximate 25% reduction in overall runtime. Furthermore, implementing the search for wall receptors based on octrees for bond formation kernels significantly accelerated adhesive bond formation, which was previously the primary computational bottleneck in adhesive dynamics, as shown in Fig. 9. This optimization greatly reduced the runtime of bond formation calculations, improving overall simulation performance.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

### 4.4 APR-AD Simulation in Complex Microvessel

The capabilities of our APR-AD framework are demonstrated using the microfluidic vessel geometry shown in Fig. 10. In this simulation, the APR window tracked an adhesive cancer cell as it flowed through the lower branch of the geometry. A bulk grid spacing of 0.6 $\mu$m was selected along with a resolution ratio $n = 4$, yielding a window grid spacing of 0.15 $\mu$m. The simulation was performed on three nodes of the Aurora supercomputer (Argonne National Laboratory), yielding significant resource savings, as summarized in Table 1. To validate the APR framework, we also conducted an explicit adhesive transport simulation at the window resolution using 32 nodes of Aurora. A sinusoidal wall receptor patterning function (Equation (1)) was used in both the eFSI and APR simulations for consistency. Both qualitative (Fig. 10(A)) and quantitative (Fig. 10(B)) comparisons between the explicit (eFSI) and APR simulations indicate strong agreement. Minor discrepancies observed in receptor-ligand binding stemmed primarily from differences in the RNG schemes between the CPU and GPU. Despite this, the overall cell trajectories (Fig. 10(B)) and CTC morphologies (Fig. 10(A)) remained consistent across both cases.

| Model | $\Delta x$ ($\mu$m) | fluid points | wall receptors | memory usage |
|---|---|---|---|---|
| APR-AD (window) | 0.15 | $7.83 \times 10^5$ | $1.41 \times 10^5$ | 1.15 GB |
| APR-AD (bulk) | 0.6 | $1.41 \times 10^6$ | 0 | 0.59 GB |
| eFSI | 0.15 | $1.09 \times 10^8$ | $6.60 \times 10^5$ | 42.6 GB |

**Table 1.** Comparison of resource requirements between APR-AD and explicit FSI (eFSI) simulations within the microfluidic device shown in Fig. 10.

## 5 Conclusion

In this work, we demonstrated the capability of our novel APR-AD framework to resolve submicrometer ligand-receptor interactions of a cancer cell in a large region of practical utility. The APR-AD scheme offers significant computational savings compared to an explicit FSI approach (Table 1). By allowing the bulk to be simulated at a coarser resolution than the window, the number of fluid points and endothelial wall receptors are substantially reduced. In particular, the total memory footprint is reduced by more than an order of magnitude ($25X$, greatly expanding the feasible simulation volume for studies of adhesive dynamics.

The presented work provided a detailed examination of the underlying multiscale modeling techniques and key implementation considerations. The presented study serves as a proof-of-concept for a novel approach to investigating the adhesion cascade of circulating tumor cells at anatomical scales, paving the way for experimental validation studies and future applications in physiological environments. Future efforts will seek to address the limitations of the presented

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

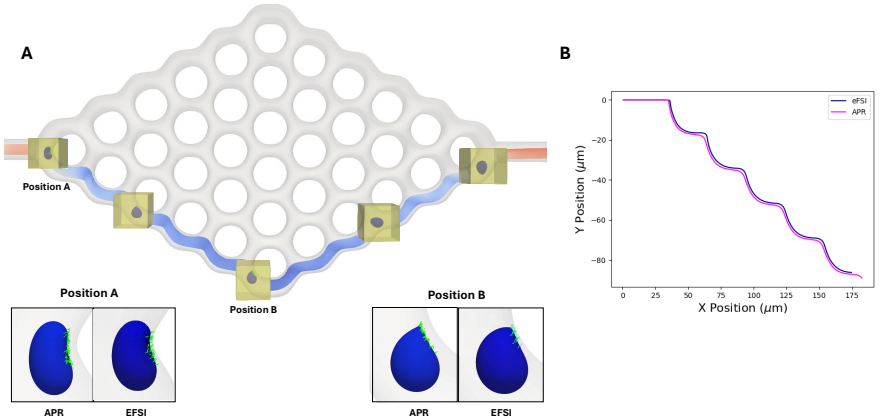


**Fig. 10.** APR simulation of a ligand-coated cancer cell (blue) traversing a microvessel geometry. (A): The APR window (yellow) tracking the cancer cell is shown at different time points during the simulation, with bonds formed between the cell and the endothelial receptors shown in green. The overall trajectory taken by the cell is indicated by its velocity pathline. The cell is shown at select positions for both the APR and corresponding eFSI simulations. (B): Quantitative comparison of CTC trajectory between eFSI (blue) and APR (magenta) in the left half of the geometry.

model by incorporating heterogeneous cell populations into the APR window and modeling the stochasticity of the adhesive receptor parameters.

## 6 Acknowledgements

The authors thank Wentao Ma, Daniel Puleri, and Jorik Stoop for fruitful discussions. This work was supported by the NCI of the NIH under Award Number 5R01EB024989. Research reported in this publication was supported by the NIH under Award Number T32GM144291. Computing support for this work came from the Argonne National Laboratory (ANL) Aurora Early Science program. An award of computer time was provided by the INCITE program. This research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357.

## References

1. Ames, J., Puleri, D.F., Balogh, P., Gounley, J., Draeger, E.W., Randles, A.: Multi-gpu immersed boundary method hemodynamics simulations. Journal of computational science **44**, 101153 (2020)
2. Cui, J., Liu, Y., Xiao, L., Chen, S., Fu, B.M.: Numerical study on the adhesion of a circulating tumor cell in a curved microvessel. Biomechanics and Modeling in Mechanobiology **20**, 243–254 (2021)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19

3. Dabagh, M., Gounley, J., Randles, A.: Localization of rolling and firm-adhesive interactions between circulating tumor cells and the microvasculature wall. Cellular and Molecular Bioengineering **13**(2), 141–154 (2020)
4. Fedosov, D., Caswell, B., Suresh, S., Karniadakis, G.: Quantifying the biophysical characteristics of plasmodium-falciparum-parasitized red blood cells in microcirculation. Proceedings of the National Academy of Sciences **108**(1), 35–39 (2011)
5. Hammer, D.A., Apte, S.M.: Simulation of cell rolling and adhesion on surfaces in shear flow: general results and analysis of selectin-mediated neutrophil adhesion. Biophysical journal **63**(1), 35–57 (1992)
6. Krüger, T., Kusumaatmaja, H., Kuzmin, A., Shardt, O., Silva, G., Viggen, E.M.: The Lattice Boltzmann Method - Principles and Practice (10 2016). https://doi.org/10.1007/978-3-319-44649-3
7. Peskin, C.S.: Numerical analysis of blood flow in the heart. Journal of Computational Physics **25**(3), 220–252 (1977)
8. Puleri, D.F., Martin, A.X., Randles, A.: Distributed acceleration of adhesive dynamics simulations. In: Proceedings of the 29th European MPI Users' Group Meeting. pp. 37–45 (2022)
9. Puleri, D.F., Randles, A.: The role of adhesive receptor patterns on cell transport in complex microvessels. Biomechanics and modeling in mechanobiology **21**(4), 1079–1098 (2022)
10. Puleri, D.F., Roychowdhury, S., Balogh, P., Gounley, J., Draeger, E.W., Ames, J., Adebiyi, A., Chidyagwai, S., Hernández, B., Lee, S., et al.: High performance adaptive physics refinement to enable large-scale tracking of cancer cell trajectory. In: 2022 IEEE International Conference on Cluster Computing (CLUSTER). pp. 230–242. IEEE (2022)
11. Qian, Y.H., d'Humières, D., Lallemand, P.: Lattice bgk models for navier-stokes equation. Europhysics letters **17**(6), 479 (1992)
12. Randles, A., Draeger, E.W., Bailey, P.E.: Massively parallel simulations of hemodynamics in the primary large arteries of the human vasculature. Journal of Computational Science **9**, 70–75 (2015)
13. Randles, A.P., Kale, V., Hammond, J., Gropp, W., Kaxiras, E.: Performance analysis of the lattice boltzmann model beyond navier-stokes. In: 2013 IEEE 27th International Symposium on Parallel and Distributed Processing. pp. 1063–1074. IEEE (2013)
14. Roychowdhury, S., Mahmud, S.T., Martin, A., Balogh, P., Puleri, D.F., Gounley, J., Draeger, E.W., Randles, A.: Enhancing adaptive physics refinement simulations through the addition of realistic red blood cell counts. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. pp. 1–13 (2023)
15. Shabbir, F., Mujeeb, A.A., Jawed, S.F., Khan, A.H., Shakeel, C.S.: Simulation of transvascular transport of nanoparticles in tumor microenvironments for drug delivery applications. Scientific Reports **14**(1), 1764 (2024)
16. Takeishi, N., Imai, Y., Ishida, S., Omori, T., Kamm, R.D., Ishikawa, T.: Cell adhesion during bullet motion in capillaries. American Journal of Physiology-Heart and Circulatory Physiology **311**(2), H395–H403 (2016)
17. Twigg, C.: Catmull-rom splines. Computer **41**(6), 4–6 (2003)
18. Ward, R., Molteno, C.: Table of linear feedback shift registers. Tech. Rep. 2012-1, University of Otago, https://www.physics.otago.ac.nz/reports/electronics/ETR2012-1.pdf
19. Ye, H., Shen, Z., Li, Y.: Cell stiffness governs its adhesion dynamics on substrate under shear flow. IEEE Transactions on Nanotechnology **17**(3), 407–411 (2017)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97626-1_19