Machine Learning for Water Movement Simulation in Game Engines: Techniques and Curriculum

1. Introduction: The Promise of Machine Learning for Water Simulation in Game Engines

Traditional methods for simulating water in video games often involve a compromise between visual realism and computational efficiency.[1] Techniques such as mesh deformation, particle systems like Smoothed Particle Hydrodynamics (SPH), and wave equation solvers are commonly employed to create the illusion of fluid movement. Mesh-based methods can effectively represent large bodies of water but may struggle with complex interactions and dynamic changes in topology. Particle systems, on the other hand, excel at simulating splashing and free surface effects but can become computationally expensive with a large number of particles. Wave equations are efficient for generating surface waves but may lack the ability to model intricate fluid dynamics or interactions with objects. These traditional approaches frequently require significant manual tuning and can face limitations in achieving both high visual fidelity and real-time performance, particularly when dealing with complex fluid behaviors.[1]

The field of scientific computing has witnessed a significant surge in the application of machine learning (ML) techniques across various domains, including fluid dynamics.[5] This increased interest is fueled by advancements in high-performance computing (HPC) and the refinement of sophisticated ML algorithms.[5] Data-driven approaches offer a promising avenue to enhance the accuracy of computational fluid dynamics (CFD) simulations, reduce their substantial computational costs, and efficiently extract meaningful patterns from extensive datasets.[5] Machine learning's ability to model intricate nonlinear relationships makes it a compelling solution for the complexities inherent in fluid flow analysis.[6] While traditional models often struggle with the intricacies of separated flows, requiring advanced simulation techniques, ML presents an opportunity to overcome these limitations.[5]

Given the pervasive presence of water in numerous gaming environments, the prospect of employing machine learning to simulate its movement for integration into popular game engines like Unreal Engine and Unity holds considerable appeal.[16] More advanced and efficient water simulation techniques can significantly contribute to a more immersive and engaging player experience. By leveraging the power of ML, it becomes possible to potentially achieve more realistic and interactive water effects within the performance constraints of game engines. This report aims to provide a comprehensive exploration of the best machine learning techniques for this purpose and to outline a structured curriculum that will enable individuals with a strong ML background to acquire the necessary knowledge and skills in fluid mechanics to effectively implement these techniques. The subsequent sections will delve into the foundational concepts of fluid mechanics, explore various ML methodologies applicable to water simulation, present a detailed curriculum for learning this interdisciplinary field, discuss relevant datasets for training ML models, address the challenges associated with real-time simulation in game engines, and finally, consider the integration of ML models within the Unreal Engine and Unity ecosystems.

2. Foundational Concepts in Fluid Mechanics for Machine Learning Practitioners

A fluid is a substance that has no fixed shape and yields easily to external pressure; both liquids and gases are considered fluids.[5] Understanding the fundamental properties of fluids is essential for interpreting simulation data and designing effective machine learning models. Key properties relevant to simulation include density, which is the mass per unit volume of the fluid; viscosity, a measure of a fluid's resistance to flow; pressure, the force exerted per unit area on a surface within the fluid; and surface tension, the tendency of liquid surfaces to shrink into the minimum surface area possible.[5]

The motion of viscous fluid substances is fundamentally described by the Navier-Stokes equations.[7] These equations are a set of partial differential equations that express the conservation of momentum and mass for fluid flow. They consist of terms representing inertia (the resistance of a fluid to changes in its state of motion), pressure gradients (driving forces due to pressure differences), viscous forces (internal friction within the fluid), and any external forces acting on the fluid.[7] While the direct analytical solution of the Navier-Stokes equations is often intractable for complex flows, understanding that machine learning models aim to approximate the solutions to these equations provides a crucial theoretical foundation for this endeavor.

Fluid flow can exhibit vastly different characteristics depending on various factors. Laminar flow is characterized by smooth, orderly movement of fluid layers, with minimal mixing between adjacent layers. In contrast, turbulent flow is chaotic and disordered, featuring significant fluctuations in velocity and pressure, as well as the formation of eddies of varying sizes.[1] Modeling turbulence remains a significant challenge in fluid simulation due to its complex, multi-scale nature, and machine learning is actively being explored as a means to improve the accuracy and efficiency of turbulence models.[6] Boundary conditions play a critical role in defining fluid flow problems by specifying the behavior of the fluid at the boundaries of the computational domain.[24] Common boundary conditions include the no-slip condition at solid walls (where the fluid velocity relative to the wall is zero) and conditions at free surfaces (where the fluid is in contact with a gas, such as air). Machine learning models must learn to respect these constraints to generate physically plausible simulations.[7] Another important distinction is between incompressible and compressible flow. Incompressible flow occurs when the density of the fluid remains relatively constant, while compressible flow involves significant changes in density. For water simulation in games, particularly at typical speeds and pressures, assuming incompressible flow is often a reasonable simplification that can reduce the complexity of both the simulation and the machine learning model.[7]

Computational Fluid Dynamics (CFD) encompasses a range of numerical methods used to simulate fluid flow.[1] These methods, such as finite difference, finite volume, and finite element methods, work by discretizing the governing equations and the flow domain into a grid or mesh, and then numerically solving the equations at discrete points in space and time.[7] Machine learning models can be trained on the vast amounts of data generated by CFD simulations or can be integrated into CFD workflows to accelerate specific tasks or enhance the accuracy of the simulations.[6] Understanding the fundamental principles of CFD provides valuable context for exploring how machine learning can be effectively applied in this domain.

3. Machine Learning Techniques for Fluid Simulation: A Detailed Exploration

Graph Neural Networks (GNNs) have emerged as a powerful class of machine learning models particularly well-suited for representing and learning the dynamics of particle-based fluid simulations.[23] In this paradigm, a fluid is represented as a graph where individual particles are treated as nodes, and the interactions between these particles, such as their proximity or the forces they exert on each other, are represented as edges.[31] This graph-based representation naturally aligns with the Lagrangian perspective of fluid dynamics, where the fluid's motion is tracked by following the individual particles.[31]

Various GNN architectures have been successfully applied to fluid dynamics problems. Graph Networks (GNs) provide a general framework that typically consists of three main components: an encoder, a processor, and a decoder.[36] The encoder transforms the input graph (representing the current state of the fluid) into a latent representation. The processor then performs message passing, where information is exchanged between connected nodes, allowing the network to learn the interactions and dependencies within the fluid system. Finally, the decoder maps the processed latent representation back to the desired output, such as the predicted accelerations or velocities of the particles.[36] MeshGraphNets (MGNs) are a specific type of GNN architecture explicitly designed for mesh-based simulations, making them highly relevant for learning complex fluid dynamics from data generated by traditional CFD methods.[38] MGNs also employ an encode-process-decode structure and are capable of handling irregular meshes, which are common in CFD. A key advantage of MGNs is their ability to learn mesh adaptation during the simulation, allowing for variable resolution and scale at runtime.[42] Graph Attention Networks (GATs) utilize attention mechanisms to assign weights to neighboring nodes, enabling the model to focus on the most relevant interactions for predicting the flow, which can be particularly useful for capturing non-equilibrium features in the flow field.[23] To address the challenge of modeling long-range interactions in fluids, multi-scale GNNs have been developed, which incorporate graph representations at different resolutions to capture both local and global dynamics.[26]

Training GNNs for fluid simulation typically involves providing the model with input features that describe the current state of the fluid, such as the positions, velocities, and types of particles.[38] The model is then trained to predict the next state of the fluid, often represented as the accelerations of the particles or their velocities at the subsequent time step.[38] The choice of these input and output features is critical for the model's ability to accurately learn and generalize the underlying fluid dynamics.[36]

Physics-Informed Neural Networks (PINNs) offer an alternative approach to fluid simulation by directly integrating physical laws, such as the Navier-Stokes equations, into the neural network's loss function.[6] This physics-informed learning paradigm allows for training models even with limited amounts of labeled data and helps to ensure that the resulting simulations adhere to fundamental physical principles.[10] A typical PINN architecture involves a neural network that takes the spatial and temporal coordinates as input and outputs the fluid properties, such as velocity, pressure, and density.[24] The loss function for training PINNs consists of two main components: a data fitting term that measures the discrepancy between the network's predictions and any available observational data, and a physics-based residual term that penalizes the network for not satisfying the governing fluid dynamics equations.[24] PINNs have been successfully applied to various problems in fluid mechanics, including flow reconstruction from sparse sensor measurements and learning unknown parameters in physical models.[24] They can be particularly valuable for augmenting experimental data or tackling problems where traditional numerical methods are computationally prohibitive.[24] However, training PINNs for highly complex turbulent flows and ensuring their generalization to unseen scenarios can still present significant challenges.[54]

Surrogate models provide another powerful application of machine learning in fluid simulation.[6] These models are trained to mimic the behavior of computationally expensive traditional CFD simulations.[7] Once trained, a surrogate model can provide rapid predictions for tasks such as design optimization or exploring vast parameter spaces, significantly reducing the computational burden associated with repeated CFD runs.[7] Various machine learning models can be employed for surrogate modeling, including neural networks, Gaussian processes, and other regression techniques.[59] The accuracy of a surrogate model heavily relies on the quality and quantity of the CFD simulation data used for training.[6] Surrogate models can be particularly useful for water simulation in games, allowing for fast approximations of complex water flow scenarios based on pre-computed CFD data.

Reinforcement Learning (RL) is a machine learning paradigm where an agent learns to make optimal decisions by interacting with an environment and receiving feedback in the form of rewards or penalties.[66] While not directly used for simulating the physics of water itself, RL can have potential applications in water-related simulations within game engines. For example, RL could be used to train intelligent agents to control underwater vehicles in a realistic fluid environment or to develop optimal policies for managing water resources in simulated ecosystems.

4. A Structured Curriculum for ML-Based Water Simulation

This curriculum provides a structured learning path for individuals with a strong machine learning background to acquire the necessary knowledge and skills for implementing ML-based water simulation, with a focus on eventual integration into Unreal Engine or Unity.

- **Chapter 1: Introduction to Fluid Mechanics**
    - **Theory:** This chapter lays the groundwork by introducing the fundamental concepts of fluid mechanics. It begins with the definition of a fluid, differentiating between liquids and gases, and establishes the continuum hypothesis, which allows us to treat fluids as continuous media rather than collections of discrete particles. Basic fluid properties such as density (mass per unit volume), viscosity (resistance to flow, including dynamic and kinematic viscosity), pressure (force per unit area), temperature (a measure of thermal energy), and surface tension (the cohesive forces at the liquid surface) are defined and explained.[5] The importance of units and dimensions in fluid mechanics is emphasized, along with

    a general introduction to fluid statics (fluids at rest) and fluid dynamics (fluids in motion).
  - **Practice:** To solidify the theoretical understanding, practical exercises in this chapter involve solving simple problems related to fluid properties, such as calculating the hydrostatic pressure at a specific depth in water. Learners will also be encouraged to identify different types of fluid flow (e.g., laminar flow from a tap, turbulent flow in a river rapids) in everyday scenarios. Exploring online resources and interactive simulations that visually demonstrate fluid properties will further enhance comprehension.
- **Chapter 2: Numerical Methods for Basic Fluid Simulation**
  - **Theory:** This chapter provides an overview of the numerical techniques that underpin traditional fluid simulation. It introduces the concept of discretization, which involves approximating continuous equations with discrete numerical schemes. Finite difference and finite volume methods are presented conceptually as ways to solve partial differential equations on a grid.[7] Basic ideas of numerical stability (the ability of a numerical method to avoid amplifying errors) and convergence (the property of a numerical solution approaching the true solution as the discretization becomes finer) are discussed. The chapter then introduces particle-based methods, with a focus on Smoothed Particle Hydrodynamics (SPH).[1] The fundamental idea of representing a fluid as a collection of interacting particles, where each particle carries fluid properties, is explained. Finally, the chapter provides a brief overview of common software used in CFD, such as OpenFOAM and Ansys Fluent.[40] The goal here is to familiarize learners with the tools and techniques used to generate the data that machine learning models often learn from, rather than to provide in-depth training on these software packages.
  - **Practice:** Practical exercises in this chapter may involve running simple SPH simulations using available open-source tools or programming libraries (if suitable for beginners with strong ML skills). Learners will visualize the output of these basic numerical simulations, observing how the fluid particles move and interact. They will also gain an understanding of the key parameters involved in SPH, such as the smoothing length (which determines the range of interaction between particles) and particle mass.
- **Chapter 3: Machine Learning Essentials for Physics**
  - **Theory:** This chapter reviews the essential machine learning concepts that are crucial for tackling physics-related problems. It begins with a recap of fundamental machine learning paradigms, including supervised learning (learning from labeled data), unsupervised learning (discovering patterns in unlabeled data), and reinforcement learning (learning through interaction and feedback).[15] The basics of deep learning are then covered, including the structure of neural networks, common activation functions (e.g., ReLU, sigmoid), the concept of loss functions (measuring the error of the model's predictions), and optimization

algorithms used to train neural networks (e.g., Adam).[25] The chapter introduces graph neural networks (GNNs), explaining how they represent data as graphs and perform computations through message passing between connected nodes. Different types of GNN layers, such as Graph Convolutional Networks (GCNs), are briefly introduced.[36] Finally, the chapter discusses the concept of inductive bias in machine learning, which refers to incorporating prior knowledge or assumptions into the model architecture to improve learning and generalization in physics applications.[23]

  - **Practice:** Practical exercises in this chapter involve implementing basic neural networks for regression tasks (predicting continuous values) using popular deep learning libraries like TensorFlow or PyTorch. Learners will also practice working with graph data structures using libraries such as NetworkX or PyTorch Geometric, and experiment with simple GNN models on synthetic or toy graph datasets to understand the fundamentals of graph-based learning.
- **Chapter 4: Deep Dive into Graph Neural Networks for Fluid Dynamics**
  - **Theory:** This chapter delves into the specifics of applying Graph Neural Networks to the domain of fluid dynamics. It provides a detailed explanation of the Graph Network (GN) architecture, highlighting its suitability for learning the complex interactions in physical systems.[36] A significant portion of the chapter is dedicated to an in-depth study of the MeshGraphNet (MGN) architecture.[38] The encoder, processor (with its message-passing mechanism), and decoder components of MGNs are thoroughly explained, along with how MGNs handle mesh data and incorporate world-space edges to account for interactions beyond the mesh connectivity. Other relevant GNN architectures for fluid simulation, such as Graph Attention Networks (GATs) and multi-scale GNNs, are also discussed, emphasizing their unique capabilities.[23] Finally, the chapter clarifies the distinction between Lagrangian (particle-based) and Eulerian (grid-based) representations of fluids in the context of how GNNs are applied to them.[31]
  - **Practice:** Practical exercises in this chapter involve implementing a basic layer of a GN or MGN using a deep learning framework like TensorFlow or PyTorch. Learners will also analyze the structure of publicly available fluid simulation datasets that are specifically designed for training GNNs, such as the DeepMind "Learning to Simulate" dataset [36], to understand the format and types of data used to train these models.
- **Chapter 5: Implementing and Training GNNs for Water Simulation**
  - **Theory:** This chapter focuses on the practical aspects of implementing and training GNNs specifically for water simulation. It guides the learner through the process of choosing an appropriate GNN architecture, with MeshGraphNets being a strong candidate for particle-based water simulation.[38] The crucial step of preparing the training data is covered, emphasizing the need to understand the input and output features provided by datasets like DeepMind's. As illustrated in Table 1, input features often include particle positions, velocities, and potentially

other properties, while the model learns to predict accelerations or the next state of velocities.[38] The chapter explains how to define an appropriate loss function for training, such as Mean Squared Error (MSE) between the predicted and ground truth accelerations or velocities.[65] Important training considerations like batch size, learning rate, the number of training epochs, and the importance of data normalization are discussed.[38] Finally, the chapter explores strategies for addressing the challenges of long-term simulation stability and generalization to new scenarios, such as adding a small amount of noise to the training data or utilizing recurrent network architectures if temporal dependencies need to be explicitly modeled.[7]

- **Practice:** Practical exercises in this chapter involve downloading and exploring a relevant dataset like the DeepMind "Learning to Simulate" dataset, which includes simulations of water.[36] Learners will then implement a basic MGN model using a deep learning framework like TensorFlow or PyTorch. The next step involves training this model on a subset of the water simulation data. Finally, the model's performance will be evaluated on a separate validation dataset to assess its ability to generalize.

Table 1: Example Input and Output Features for Training a GNN for Water Simulation (Based on DeepMind Dataset)

Feature Type	**Feature Name(s)**	**Description**	**Input/Output**
Node Features	Position (x, y, z)	Current position of each water particle in 3D space	Input
Node Features	Velocity (vx, vy, vz)	Current velocity of each water particle in 3D space	Input
Node Features	Node Type (e.g., fluid, boundary)	Categorical identifier for the type of particle	Input
Node Features	Historical Velocity (vx_hist, vy_hist, vz_hist)	Previous time steps' velocities (number of steps can vary)	Input
Edge Features	Displacement (dx, dy, dz)	Vector pointing from one particle to another within the interaction radius	Input
Edge Features	Distance	Euclidean distance between two interacting particles	Input
Node Target	Acceleration (ax, ay, az)	Predicted acceleration of each water particle in 3D space for the next step	Output

- **Chapter 6: Integrating ML Models into Unity and Unreal Engine**
    - **Theory:** This chapter focuses on the practical steps involved in taking a trained machine learning model and integrating it into the Unity and Unreal Engine game development environments. It begins with an overview of the inference process, which is the act of running a trained ML model to generate predictions in real-time. The chapter then details the process of exporting trained models to the ONNX (Open Neural Network Exchange) format.[74] ONNX is a crucial standard as it provides cross-platform compatibility, allowing models trained in frameworks like TensorFlow or PyTorch to be used by different inference engines.[74] The chapter provides a thorough explanation of how to utilize Unity Barracuda, Unity's built-in neural network inference library, for executing these ONNX models within Unity.[74] It also introduces Unity Sentis as a newer alternative to Barracuda for neural network inference in Unity, highlighting its capabilities and potential advantages.[19] Finally, the chapter provides a brief overview of the TensorFlow plugin available for Unreal Engine, offering a starting point for integrating TensorFlow models into that engine.[66]
    - **Practice:** The practical exercises in this chapter involve exporting a simple machine learning model that has been previously trained (either from Chapter 5 or a readily available pre-trained model) into the ONNX format. Learners will then import this ONNX model into a Unity project using either the Barracuda or Sentis package. They will write basic C# scripts within Unity to load the imported model and prepare the necessary input data, such as the positions of water particles. Finally, they will implement the inference process using the loaded model and access the resulting output, such as the predicted accelerations of the water particles.
- **Chapter 7: Rendering and User Interaction with ML-Simulated Water**
    - **Theory:** This chapter focuses on how to visually represent the ML-simulated water within the game engine and how to enable user interaction with it. It explores various techniques for rendering particle-based fluids in both Unity and Unreal Engine.[16] These techniques include using Unity's built-in particle system to visualize individual water particles, employing impostor splatting for efficient rendering of large numbers of particles, and utilizing surface meshing techniques (such as the Obi Fluid Surface Mesher in Unity) to create a continuous fluid surface.[16] The chapter also discusses the use of shaders to enhance the visual realism of the water, covering aspects like transparency, reflections, refractions, and the creation of dynamic wave effects.[16] Finally, it explores methods for implementing basic user interaction with the simulated water, such as allowing players to apply forces to the water, introduce obstacles into the flow, or otherwise influence its behavior.[16]
    - **Practice:** Practical exercises in this chapter involve taking the output from the ML model (predicted particle positions) and visualizing it using Unity's particle system. Learners will experiment with different rendering techniques for

particle-based fluids to understand their visual and performance characteristics. They will also write Unity scripts to enable basic forms of user interaction with the simulated water, such as allowing the user to click and drag to apply forces to the fluid or to spawn simple objects that interact with it.

- **Chapter 8: Advanced Topics, Datasets, and Performance Optimization**
  - **Theory:** This final chapter delves into more advanced concepts and practical considerations. It introduces more sophisticated GNN architectures for fluid simulation, such as X-MeshGraphNet, which is designed for improved scalability in large-scale simulations.[43] The chapter also discusses the use of physics-informed loss functions during the training of GNNs, which can help to enforce fundamental physical constraints like conservation of mass and momentum, leading to more realistic and stable simulations.[26] A significant focus is placed on techniques for improving the performance of machine learning inference within game engines.[82] These techniques include model quantization (reducing the precision of the model's weights and activations), optimizing tensor operations for the specific hardware, and leveraging GPU acceleration whenever possible. Finally, the chapter touches upon considerations for scaling the water simulation to handle a larger number of particles or more complex interaction scenarios while maintaining real-time performance.
  - **Practice:** The practical exercises in this chapter are more exploratory. Learners will investigate and conceptually understand more advanced GNN architectures for fluid simulation. They will also explore the documentation and available tools within Unity (Barracuda or Sentis) and Unreal Engine for optimizing the performance of integrated neural network models. This might involve experimenting with different inference backends (CPU vs. GPU) or exploring options for model quantization provided by the respective platforms.

5. Relevant Datasets for Training Machine Learning Models for Water Simulation

Training effective machine learning models for water simulation necessitates access to large and high-quality datasets of fluid flow. Several datasets have been made publicly available by the research community, offering valuable resources for this purpose.

The DeepMind "Learning to Simulate" dataset is a prominent example, providing a collection of particle-based simulations across various physical domains, including water.[36] This dataset includes simulations of water splashing and bouncing within containers, generated using Smoothed Particle Hydrodynamics (SPH) and Material Point Method (MPM) simulators.[70] The dataset contains a significant number of particles and time steps, making it suitable for training Graph Neural Networks to learn the underlying physics.[70] Access to this dataset is typically provided through the DeepMind research repository.[70]

The EAGLE dataset is a large-scale collection of approximately 1.1 million 2D meshes resulting from simulations of unsteady fluid dynamics, specifically focusing on airflow produced by a moving unmanned aerial vehicle interacting with various scene structures.[71] While primarily focused on airflow, the underlying principles and the dataset's scale could potentially offer insights or be adaptable for certain 2D water simulation scenarios, particularly those involving

complex boundary interactions.[71]
The Fluid Cube Dataset contains 100 unique fluid simulations of a fluid block moving within a unit cube.[69] These simulations were generated using the SPH method, with variations in initial shape, position, velocity, and viscosity of the fluid block.[69] This dataset is well-suited for researchers and engineers looking to understand fluid behavior under different physical conditions and for testing machine learning algorithms designed to predict fluid flow.[69]
BLASTNet is presented as the first large machine learning dataset specifically tailored for fundamental fluid dynamics.[115] This dataset aims to address the high dimensionality of scientific data and comprises a substantial number of samples across diverse flow configurations, including data focused on carbon-free hydrogen fuels.[115] Its scale and diversity make it a potentially valuable resource for more advanced research in ML-driven fluid simulation.
Other datasets mentioned in the research material include a fluid flow dataset parameterized by the Reynolds number, containing a wide spectrum of laminar and turbulent flow regimes.[116] Additionally, the CellDJBench dataset focuses on biological fluid simulation, containing various datasets encompassing biological dynamics, which might be relevant for specific niche applications.[117]
When choosing a dataset for training, several factors should be considered. These include the dimensionality of the simulation (2D vs. 3D), the complexity of the simulated scenarios and interactions, the total number of data points available, the format in which the data is provided, and the specific type of fluid behavior that the model is intended to learn.[70]

6. Challenges and Considerations for Real-Time Simulation in Game Engines

Integrating machine learning models for water simulation into game engines to achieve real-time performance presents several significant challenges and considerations.[1] One primary concern is the computational cost associated with running ML inference, especially for complex models like deep neural networks and when simulating a large number of particles.[1] While ML models can potentially offer speedups compared to traditional CFD methods, executing a neural network for every frame of a game can still be computationally demanding and may impact the overall frame rate.[2]
Another crucial aspect is ensuring that ML models trained on specific datasets can generalize effectively to the diverse and often unseen scenarios encountered in different game environments and during player interactions.[7] A model trained solely on simulations of simple water splashes might not accurately predict the behavior of water interacting with complex in-game objects or under varying environmental conditions.[118] The diversity and quality of the training data are therefore paramount for achieving good generalization.[15]
Maintaining physical plausibility and adhering to fundamental conservation laws, such as the conservation of mass and momentum, is also a critical consideration.[7] A purely data-driven ML model might learn to produce visually appealing results that do not necessarily respect these physical laws, potentially leading to unrealistic and unpredictable behavior within the game.[7] Physics-informed learning approaches, where physical laws are incorporated into the training process, can help to mitigate this issue.[10]

Training effective ML models typically requires large amounts of high-quality simulation data.[6] Generating this data using traditional CFD methods can be a time-consuming and computationally expensive process.[115] Furthermore, integrating ML models into existing game engine pipelines and ensuring their seamless interaction with other game systems, such as physics engines and rendering pipelines, can present technical complexities.[119] Finally, for interactive water simulations, the latency introduced by the ML inference process needs to be minimized to provide a responsive and engaging user experience.[83] Any significant delay between player input or game events and the visual response of the water simulation can negatively impact gameplay.

7. Conclusion: The Future of ML-Driven Water Simulation in Interactive Environments

In conclusion, machine learning presents a compelling and promising avenue for advancing water simulation techniques within game engines. By leveraging the power of Graph Neural Networks, Physics-Informed Neural Networks, and surrogate models, it becomes possible to potentially achieve more realistic, interactive, and computationally efficient water effects compared to traditional methods. Graph Neural Networks, particularly MeshGraphNets, offer a robust framework for learning complex fluid dynamics from particle-based simulation data, demonstrating strong generalization capabilities.[36] Physics-Informed Neural Networks provide a way to incorporate fundamental physical laws into the learning process, enabling training with limited data and ensuring physical plausibility.[10] Surrogate models offer the potential to significantly accelerate traditional CFD simulations, allowing for fast approximations of complex water flow scenarios.[7]

The field of machine learning for fluid dynamics is rapidly evolving, with emerging trends pointing towards more sophisticated GNN architectures capable of handling larger scales and more complex interactions.[43] Further advancements in Physics-Informed Machine Learning (PIML) methods promise to enhance the accuracy and stability of these models, particularly for challenging turbulent flows.[6] The development of more efficient inference techniques specifically tailored for the hardware constraints of game engines will be crucial for enabling real-time performance.[82]

The journey towards seamlessly integrating advanced ML-driven water simulation into interactive experiences is an exciting one. While challenges remain in terms of computational cost, generalization, and ensuring physical accuracy, the potential benefits for enhancing the visual fidelity and interactivity of games are substantial.[13] By following a structured curriculum that bridges the gap between fluid mechanics and machine learning, and by leveraging the growing availability of relevant datasets and tools, developers and researchers can continue to push the boundaries of what is possible in creating truly immersive and dynamic virtual worlds.

Works cited

1. A COMPREHENSIVE REVIEW OF FLUID DYNAMICS SIMULATION TECHNIQUES FOR REAL-TIME AND COMPUTATIONAL EFFICIENCY - (https://www.irjmets.com)., accessed May 14, 2025,

https://www.irjmets.com/uploadedfiles/paper//issue_1_january_2025/66079/final/fin_irjmets1738685871.pdf
2. Advancements in Realistic Physics Simulation for Games - Argentics, accessed May 14, 2025, https://www.argentics.io/advancements-in-realistic-physics-simulation-for-games
3. How is in-game real time fluid flow simulated? : r/howdidtheycodeit - Reddit, accessed May 14, 2025, https://www.reddit.com/r/howdidtheycodeit/comments/ta4zy3/how_is_ingame_real_time_fluid_flow_simulated/
4. Why are there still no real time fluid simulations in games today? The hardware can easily handle at least small simulations easily, but I don't know of any non-indie game that has them? Why is it all still shaders and displacements? : r/truegaming - Reddit, accessed May 14, 2025, https://www.reddit.com/r/truegaming/comments/k7wbad/why_are_there_still_no_real_time_fluid/
5. A Review of Simulations and Machine Learning Approaches for Flow Separation Analysis, accessed May 14, 2025, https://www.mdpi.com/2226-4310/12/3/238
6. Advancing Computational Fluid Dynamics through Machine Learning: A Review of Data-Driven Innovations and Applications - ResearchGate, accessed May 14, 2025, https://www.researchgate.net/publication/385253160_Advancing_Computational_Fluid_Dynamics_through_Machine_Learning_A_Review_of_Data-Driven_Innovations_and_Applications
7. Machine learning–accelerated computational fluid dynamics - PNAS, accessed May 14, 2025, https://www.pnas.org/doi/10.1073/pnas.2101784118
8. Recent Advances on Machine Learning for Computational Fluid Dynamics: A Survey - arXiv, accessed May 14, 2025, https://arxiv.org/abs/2408.12171
9. Machine Learning for Fluid Mechanics - Annual Reviews, accessed May 14, 2025, https://www.annualreviews.org/doi/10.1146/annurev-fluid-010719-060214
10. A Review of Physics-Informed Machine Learning in Fluid Mechanics - MDPI, accessed May 14, 2025, https://www.mdpi.com/1996-1073/16/5/2343
11. Towards a new paradigm in intelligence-driven computational fluid dynamics simulations, accessed May 14, 2025, https://www.tandfonline.com/doi/full/10.1080/19942060.2024.2407005
12. Current and emerging deep-learning methods for the simulation of fluid dynamics - Journals, accessed May 14, 2025, https://royalsocietypublishing.org/doi/10.1098/rspa.2023.0058
13. Machine Learning for Computational Fluid Dynamics - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=IXMSOSEj14Q
14. Machine Learning for Fluid Mechanics - University of Washington, accessed May 14, 2025, https://faculty.washington.edu/sbrunton/ARFM2020.pdf
15. Challenges and Opportunities for Machine Learning in Fluid Mechanics - ResearchGate, accessed May 14, 2025, https://www.researchgate.net/publication/358898582_Challenges_and_Opportuni

ties_for_Machine_Learning_in_Fluid_Mechanics
16. Unity Fluid Simulation Tutorial: CPU & GPU Methods - Daily.dev, accessed May 14, 2025, https://daily.dev/blog/unity-fluid-simulation-tutorial-cpu-and-gpu-methods
17. I'm developing a GPU Fluid Sim that can flow over terrain creating rivers : r/Unity3D - Reddit, accessed May 14, 2025, https://www.reddit.com/r/Unity3D/comments/18oew5d/im_developing_a_gpu_fluid_sim_that_can_flow_over/
18. #UE5 Series: Intro to Liquid Simulation in Unreal Engine - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=fyMhOi1eQ_s
19. Unity Sentis: Use AI models in Unity Runtime, accessed May 14, 2025, https://unity.com/products/sentis
20. Simple Liquid Simulation in Unity! - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=_8v4DRhHu2g&pp=0gcJCdgAo7VqN5tD
21. Coding Adventure: Ocean Simulation and Buoyancy - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=e0gKrx5JEn8
22. Unity 6 Preview is now available, accessed May 14, 2025, https://unity.com/blog/engine-platform/unity-6-preview-release
23. Graph attention network-based fluid simulation model | AIP Advances, accessed May 14, 2025, https://pubs.aip.org/aip/adv/article/12/9/095114/2819821/Graph-attention-network-based-fluid-simulation
24. New insights into experimental stratified flows obtained through physics-informed neural networks | Journal of Fluid Mechanics | Cambridge Core, accessed May 14, 2025, https://www.cambridge.org/core/journals/journal-of-fluid-mechanics/article/new-insights-into-experimental-stratified-flows-obtained-through-physicsinformed-neural-networks/C70BDF645A700AE3AA9C91A028FCFB68
25. A Review of Physics-Informed Machine Learning in Fluid Mechanics - Stanford University, accessed May 14, 2025, http://web.stanford.edu/group/ihmegroup/cgi-bin/MatthiasIhme/wp-content/papercite-data/pdf/sharma2023physics.pdf
26. Multi-scale graph neural network for physics-informed fluid simulation - ResearchGate, accessed May 14, 2025, https://www.researchgate.net/publication/380663224_Multi-scale_graph_neural_network_for_physics-informed_fluid_simulation
27. A Pioneering Neural Network Method for Efficient and Robust Fluid Simulation - arXiv, accessed May 14, 2025, https://arxiv.org/html/2412.10748v3
28. Rapid spatio-temporal flood modelling via hydraulics-based graph neural networks, accessed May 14, 2025, https://hess.copernicus.org/articles/27/4227/2023/
29. [2202.12619] Fluid Simulation System Based on Graph Neural Network - arXiv, accessed May 14, 2025, https://arxiv.org/abs/2202.12619
30. Inverse Design for Fluid-Structure Interactions using Graph Network Simulators, accessed May 14, 2025, https://proceedings.neurips.cc/paper_files/paper/2022/file/59593615e358d522955

78e0d8e94ec4a-Paper-Conference.pdf
31. ACCELERATING LAGRANGIAN FLUID SIMULATION WITH GRAPH NEURAL NETWORKS, accessed May 14, 2025, https://simdl.github.io/files/34.pdf
32. Combining Differentiable PDE Solvers and Graph Neural Networks for Fluid Flow Prediction - Proceedings of Machine Learning Research, accessed May 14, 2025, http://proceedings.mlr.press/v119/de-avila-belbute-peres20a/de-avila-belbute-peres20a.pdf
33. Learning Lagrangian Fluid Dynamics with Graph Neural Networks - OpenReview, accessed May 14, 2025, https://openreview.net/forum?id=7WwYBADS3E_
34. Fluid Simulation System Based on Graph Neural Network - arXiv, accessed May 14, 2025, https://arxiv.org/pdf/2202.12619
35. Develop Physics-Informed Machine Learning Models with Graph Neural Networks | NVIDIA Technical Blog, accessed May 14, 2025, https://developer.nvidia.com/blog/develop-physics-informed-machine-learning-models-with-graph-neural-networks/
36. Learning to Simulate Complex Physics with Graph Networks - CS Stanford, accessed May 14, 2025, https://cs.stanford.edu/people/jure/pubs/learning_to_simulate-icml20.pdf
37. Learning to Simulate Complex Physics with Graph Networks, accessed May 14, 2025, https://proceedings.mlr.press/v119/sanchez-gonzalez20a/sanchez-gonzalez20a.pdf
38. MeshGraphNet with Lagrangian mesh - NVIDIA Docs, accessed May 14, 2025, https://docs.nvidia.com/deeplearning/physicsnemo/physicsnemo-core/examples/cfd/lagrangian_mgn/readme.html
39. MeshGraphNet with Lagrangian mesh - NVIDIA Docs Hub, accessed May 14, 2025, https://docs.nvidia.com/deeplearning/modulus/modulus-core/examples/cfd/lagrangian_mgn/readme.html
40. Accelerating Computational Fluid Dynamics Simulation of Post-combustion Carbon Capture Modeling with MeshGraphNets | Journal Article | PNNL, accessed May 14, 2025, https://www.pnnl.gov/publications/accelerating-computational-fluid-dynamics-simulation-post-combustion-carbon-capture
41. Generalization capabilities of MeshGraphNets to unseen geometries for fluid dynamics, accessed May 14, 2025, https://arxiv.org/html/2408.06101v1
42. MeshGraphNet Explained - Papers With Code, accessed May 14, 2025, https://paperswithcode.com/method/meshgraphnet
43. X-MeshGraphNet: NVIDIA's Scalable Solution for Physics Simulation Using Graph Neural Networks, accessed May 14, 2025, https://neurohive.io/en/state-of-the-art/x-meshgraphnet-nvidia-s-scalable-solution-for-physics-simulation-using-graph-neural-networks/
44. Learning Mesh-Based Simulation with Graph Networks - Tobias Pfaff (DeepMind) - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=fLo39PSLvsw

45. [2408.06101] Generalization capabilities of MeshGraphNets to unseen geometries for fluid dynamics - arXiv, accessed May 14, 2025, https://arxiv.org/abs/2408.06101
46. MeshGraphNet for transient vortex shedding - NVIDIA Docs, accessed May 14, 2025, https://docs.nvidia.com/deeplearning/physicsnemo/physicsnemo-core/examples/cfd/vortex_shedding_mgn/readme.html
47. Learning Mesh-Based Simulation with Graph Networks - ML4Eng @ NeurIPS 2020, accessed May 14, 2025, https://ml4eng.github.io/camera_readys/14.pdf
48. (PDF) MeshGraphNet: An Effective 3D Polygon Mesh Recognition With Topology Reconstruction - ResearchGate, accessed May 14, 2025, https://www.researchgate.net/publication/347477125_MeshGraphNet_An_Effective_3D_Polygon_Mesh_Recognition_With_Topology_Reconstruction
49. Learning Mesh-Based Simulation with Graph Networks - OpenReview, accessed May 14, 2025, https://openreview.net/forum?id=roNqYL0_XP
50. [2010.03409] Learning Mesh-Based Simulation with Graph Networks - arXiv, accessed May 14, 2025, https://arxiv.org/abs/2010.03409
51. Learning mesh-based simulations - Google Sites, accessed May 14, 2025, https://sites.google.com/view/meshgraphnets
52. Physics-informed neural networks for fluid mechanics - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=EHrgSPHZG3Y
53. Adopting Computational Fluid Dynamics Concepts for Physics-Informed Neural Networks | AIAA SciTech Forum, accessed May 14, 2025, https://arc.aiaa.org/doi/10.2514/6.2025-0269
54. Physics Informed Neural Networks (PINNs) [Physics Informed Machine Learning] - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=-zrY7P2dVC4
55. [2205.14249] Experience report of physics-informed neural networks in fluid simulations: pitfalls and frustration - arXiv, accessed May 14, 2025, https://arxiv.org/abs/2205.14249
56. Raocp/PINN-laminar-flow: Physics-informed neural network for solving fluid dynamics problems - GitHub, accessed May 14, 2025, https://github.com/Raocp/PINN-laminar-flow
57. Physics-informed neural networks (PINNs) for fluid mechanics: A review - arXiv, accessed May 14, 2025, https://arxiv.org/abs/2105.09506
58. Thoughts on NeuralVDB/AI Simulation Tech | Forums - SideFX, accessed May 14, 2025, https://www.sidefx.com/forum/post/389766/
59. Shallow neural networks for fluid flow reconstruction with limited sensors | Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences - Journals, accessed May 14, 2025, https://royalsocietypublishing.org/doi/10.1098/rspa.2020.0097
60. CFD using Neural Networks - Reddit, accessed May 14, 2025, https://www.reddit.com/r/CFD/comments/gmhyat/cfd_using_neural_networks/
61. New method uses machine learning for more robust fluid dynamics simulations, accessed May 14, 2025,

https://www.imperial.ac.uk/news/244054/new-method-uses-machine-learning-more/
62. [R] How machine learning will revolutionise physics simulations in games? - Reddit, accessed May 14, 2025, https://www.reddit.com/r/MachineLearning/comments/phvgzb/r_how_machine_learning_will_revolutionise_physics/
63. Challenges and Opportunities for Machine Learning in Fluid Mechanics - arXiv, accessed May 14, 2025, https://arxiv.org/html/2202.12577v3
64. Probabilistic neural networks for fluid flow surrogate modeling and data recovery, accessed May 14, 2025, https://link.aps.org/doi/10.1103/PhysRevFluids.5.104401
65. Full article: Airvox: efficient computational fluid dynamics prediction using 3D convolutional neural networks for building design - Taylor & Francis Online, accessed May 14, 2025, https://www.tandfonline.com/doi/full/10.1080/19401493.2024.2410744
66. Use of Machine learning and Deep Reinforcement Learning for Game Development, accessed May 14, 2025, https://forums.unrealengine.com/t/use-of-machine-learning-and-deep-reinforcement-learning-for-game-development/99044
67. How to use Machine Learning AI in Unity! (ML-Agents) - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=zPFU30tbyKs&pp=0gcJCdgAo7VqN5tD
68. FleX ML Agents Simulation Environment | neuroailab.github.io, accessed May 14, 2025, https://neuroailab.github.io/flex-ml-agents/
69. Fluid Cube Dataset (0.1) - Inductiva.AI, accessed May 14, 2025, https://inductiva.ai/blog/article/fluid-cube-dataset
70. Learning to simulate - Google Sites, accessed May 14, 2025, https://sites.google.com/view/learning-to-simulate
71. EAGLE Dataset, accessed May 14, 2025, https://eagle-dataset.github.io/
72. MeshGraphNet with Lagrangian mesh - NVIDIA Docs, accessed May 14, 2025, https://docs.nvidia.com/deeplearning/physicsnemo/physicsnemo-core/examples/cfd/lagrangian_mgn/README.html
73. PyTorch Implementation of Learning-to-Simulate (ICML2020). - GitHub, accessed May 14, 2025, https://github.com/Emiyalzn/Learn-to-Simulate
74. Getting started with Barracuda | Barracuda | 1.0.4 - Unity - Manual, accessed May 14, 2025, https://docs.unity3d.com/Packages/com.unity.barracuda@1.0/manual/GettingStarted.html
75. Getting started with Barracuda | Barracuda | 1.0.4 - Unity - Manual, accessed May 14, 2025, https://docs.unity3d.com/Packages/com.unity.barracuda%401.0/manual/GettingStarted.html
76. Run PyTorch inside Unity, accessed May 14, 2025, https://discuss.pytorch.org/t/run-pytorch-inside-unity/203079
77. Create next-gen features with AI models using Unity Sentis, accessed May 14, 2025, https://unity.com/blog/games/create-next-gen-ai-models-with-unity-sentis

78. how to deploy pytorch neural network for production code in Unity - Stack Overflow, accessed May 14, 2025, https://stackoverflow.com/questions/56316823/how-to-deploy-pytorch-neural-network-for-production-code-in-unity
79. Tutorials - Christian Mills, accessed May 14, 2025, https://christianjmills.com/series/tutorials/
80. What are the relevant steps to deploy a model as a Unity usable version? - Stack Overflow, accessed May 14, 2025, https://stackoverflow.com/questions/61281090/what-are-the-relevant-steps-to-deploy-a-model-as-a-unity-usable-version
81. Introduction to Barracuda | Barracuda | 1.0.4 - Unity - Manual, accessed May 14, 2025, https://docs.unity3d.com/Packages/com.unity.barracuda@1.0/manual/index.html
82. Configuring your Unity project for stronger performance, accessed May 14, 2025, https://unity.com/how-to/project-configuration-and-assets
83. Frequently Asked Questions (FAQ) | Barracuda | 1.0.4 - Unity - Manual, accessed May 14, 2025, https://docs.unity3d.com/Packages/com.unity.barracuda@1.0/manual/FAQ.html
84. DATA 320 Final - Library - PackageCache - com.unity.barracuda@3.0.0 - CHANGELOG.md - GitLab, accessed May 14, 2025, https://code.wm.edu/dkwon02/data-320/-/blob/1ffcfff9b14bd48c27592c69056f3474b27cfc2e/DATA%20320%20Final/Library/PackageCache/com.unity.barracuda@3.0.0/CHANGELOG.md
85. Class ModelOptimizer | Barracuda | 1.0.4 - Unity - Manual, accessed May 14, 2025, https://docs.unity3d.com/Packages/com.unity.barracuda@1.0/api/Unity.Barracuda.ModelOptimizer.html
86. Getting Started With Deep Learning in Unity - Christian Mills, accessed May 14, 2025, https://christianjmills.com/posts/deep-learning-unity-intro/
87. Help with Barracuda Object Detection Latency [Unity 2021] : r/Unity3D - Reddit, accessed May 14, 2025, https://www.reddit.com/r/Unity3D/comments/wvbl4y/help_with_barracuda_object_detection_latency/
88. Performance difference between Version 1.0.4 and 1.2.1 · Issue #130 · Unity-Technologies/barracuda-release - GitHub, accessed May 14, 2025, https://github.com/Unity-Technologies/barracuda-release/issues/130
89. (PDF) Real-time object detection and augmentation - ResearchGate, accessed May 14, 2025, https://www.researchgate.net/publication/383681914_Real-time_object_detection_and_augmentation
90. Syn-McJ/TFClassify-Unity-Barracuda: An example of using Tensorflow and ONNX models with Unity Barracuda inference engine for image classification and object detection. - GitHub, accessed May 14, 2025, https://github.com/Syn-McJ/TFClassify-Unity-Barracuda
91. How to use tensorflow model in Unity inference engine? #1825 - GitHub, accessed May 14, 2025,

https://github.com/Unity-Technologies/ml-agents/issues/1825
92. TensorFlow Lite VS Barracuda Object Detection In Unity's Augmented Reality, accessed May 14, 2025, https://stackoverflow.com/questions/66758219/tensorflow-lite-vs-barracuda-object-detection-in-unitys-augmented-reality
93. Tensorflow lite body recognition vs Unity Barracuda - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=PXaqYM7qY4E
94. TFClassify Unity -- UWA问答| 开源库| 帮助开发者发现更好的解决方案| 侑虎科技, accessed May 14, 2025, https://lab.uwa4d.com/lab/5b442a05d7f10a201faf6639
95. Machine Learning Models in Unity with Barracuda: Image Classification - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=LhzKfx2kuDs
96. sentis-samples/StarSimulationSample/README.md at main - GitHub, accessed May 14, 2025, https://github.com/Unity-Technologies/sentis-samples/blob/main/StarSimulationSample/README.md
97. Sentis overview | Sentis | 2.1.2 - Unity - Manual, accessed May 14, 2025, https://docs.unity3d.com/Packages/com.unity.sentis@latest/
98. Ray Traced Pong starting to looking pretty good on integrated graphics (neural rendering using Sentis) : r/Unity3D - Reddit, accessed May 14, 2025, https://www.reddit.com/r/Unity3D/comments/17vo3a6/ray_traced_pong_starting_to_looking_pretty_good/
99. Unity 6 is here: See what's new, accessed May 14, 2025, https://unity.com/blog/unity-6-features-announcement
100. Fluid Rendering - Obi Physics for Unity, accessed May 14, 2025, https://obi.virtualmethodstudio.com/manual/7.0/fluidrendering.html
101. Particle Rendering - Obi Physics for Unity, accessed May 14, 2025, https://obi.virtualmethodstudio.com/manual/6.3/particlerendering.html
102. Coding a Realtime Fluid Simulation in Unity [Pt. 1] - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=zbBwKMRyavE&pp=0gcJCdgAo7VqN5tD
103. Making Water in Unity (Unity 2D SPH Fluid Simulation) - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=NlqqDcAK5rM&pp=0gcJCdgAo7VqN5tD
104. Made an Opensource, Realtime, Particle-based Fluid Simulation Sandbox Game / Engine for Unity! : r/Unity3D - Reddit, accessed May 14, 2025, https://www.reddit.com/r/Unity3D/comments/1ki8opd/made_an_opensource_realtime_particlebased_fluid/
105. Coding Adventure: Rendering Fluids - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=kOkfC5fLfgE
106. A PBD fluid in unity running on the GPU - GitHub, accessed May 14, 2025, https://github.com/Scrawk/PBD-Fluid-in-Unity
107. Physics - Unity - Manual, accessed May 14, 2025, https://docs.unity3d.com/6000.1/Documentation/Manual/class-PhysicsManager.html
108. Physics - Unity - Manual, accessed May 14, 2025,

https://docs.unity3d.com/6000.1/Documentation/Manual/PhysicsSection.html
109. Physics solutions for game development - Unity, accessed May 14, 2025, https://unity.com/solutions/programming-physics
110. Unity + Deep Learning Support | Legacy MuJoCo Forum - Roboti LLC, accessed May 14, 2025, https://www.roboti.us/forum/index.php?threads/unity-deep-learning-support.3489/
111. Accurate physics simulation of small objects · Issue #424 · Unity-Technologies/ml-agents, accessed May 14, 2025, https://github.com/Unity-Technologies/ml-agents/issues/424
112. can I use Unity for simulations? : r/Unity3D - Reddit, accessed May 14, 2025, https://www.reddit.com/r/Unity3D/comments/zqscnw/can_i_use_unity_for_simulations/
113. X-MeshGraphNet: Scalable Multi-Scale Graph Neural Networks for Physics Simulation, accessed May 14, 2025, https://arxiv.org/html/2411.17164v2
114. Setting up Unity* Barracuda to Enable AI Style Transfer - Intel, accessed May 14, 2025, https://www.intel.com/content/www/us/en/developer/articles/technical/in-game-style-transfer-tutorial-leveraging-unity.html
115. BLASTNet – The First Large Machine Learning Dataset for Fundamental Fluid Dynamics, accessed May 14, 2025, https://hai.stanford.edu/news/blastnet-first-large-machine-learning-dataset-fundamental-fluid-dynamics
116. A Fluid Flow Data Set for Machine Learning - Research Collection, accessed May 14, 2025, https://www.research-collection.ethz.ch/handle/20.500.11850/515488
117. CellDJBench: Benchmark Datasets for Data-Driven Biological Fluid Simulation, accessed May 14, 2025, https://openreview.net/forum?id=KTHUTtEX5F
118. Is AI/ML just hype or do you think it will actually speed up solving time? : r/CFD - Reddit, accessed May 14, 2025, https://www.reddit.com/r/CFD/comments/12dsate/is_aiml_just_hype_or_do_you_think_it_will/
119. Unreal-Engine-Based General Platform for Multi-Agent Reinforcement Learning - arXiv, accessed May 14, 2025, https://arxiv.org/html/2503.15947v1
120. Can the UE be used efficiently for pure physics simulations? - Unreal Engine Forums, accessed May 14, 2025, https://forums.unrealengine.com/t/can-the-ue-be-used-efficiently-for-pure-physics-simulations/44639
121. Machine Learning for Computational Fluid Dynamics - YouTube, accessed May 14, 2025, https://www.youtube.com/watch?v=lXMSOSEj14Q&pp=0gcJCdgAo7VqN5tD